A New Systematic Grammar of TLI Loglan

Randall Holmes

June 13, 2020

Chapter 1

Introduction

This is a new grammar of TLI Loglan. The intention is to present a coherent picture of my provisional adjustments of the language. The organizational principle is that I follow the structure of the Parsing Expression Grammar (PEG) which is used to generate the computer grammar.

There will be observations on points of difference with earlier versions of the language as necessary.

This document speaks authoritatively, but not all these proposals have been approved by the TLI Loglan community. I am writing this in hope of providing support for a consensus that this is the way to proceed. The membership is welcome to offer criticisms, whether general or of particular points.

The reader can safely ignore comments in footnotes unless already proficient in Loglan or interested in the history of the language.

Chapter 2

Phonetics and Orthography

In this chapter, we discuss the rules for writing and pronouncing Loglan, and the way in which a stream of speech sounds or letters is formally resolved into Loglan words (with the proviso that grammar rather than phonetics dictates word boundaries between structure words).

2.1 Phonetic and orthographic components

2.1.1 Loglan letters and punctuation

The letters of the Loglan alphabet are the 23 letters of the Roman alphabet excluding **q, w, x**.

The foreign letters **q, w, x** can only occur in "alien text" embedded in Loglan.[1]

The vowels are **a, e, i, o, u, y**. The first five are the regular vowels.

The consonants are **b, c, d, f, g, h, j, k, l, m, n, p, r, s, t, v, z**

The Loglan name of a vowel V has two forms (legacy and modern[2]): legacy uppercase is V**ma** and lowercase is V**fi**, and modern uppercase is **zi**V**ma** and lowercase **zi**V. The legacy forms are fully supported, but they

[1]These letters were originally not included in Loglan, then they were added with strange pronunciations in the 1980's and 90's, then they were largely eliminated from the dictionary in the late 90's; after 2013, we proceeded to eliminate them completely again. Names for these letters (usable as pronouns) will be presented later.

[2]The modern forms were suggested by us after 2013, but we have fully accommodated the phonetics to the original forms.

are phonetically irregular as Loglan words, and there are contexts where the modern forms must be used.

The Loglan name of a consonant C is C**ai** (uppercase) or C**ei** (lowercase).

There are some other series of letter names to be introduced below. The primary function of these words is not phonetic, but as variables (pronouns), as will be explained later.

The junctures, indicating syllable breaks or stress are **-**, **'**, *****. The hyphen **-** is a simple syllable break.[3] **'** is a syllable stress marker, which may appear in place of (not in addition to) a hyphen after a stressed syllable, or in final position in a word after a stressed syllable. ***** is a symbol for emphatic syllable stress, with the same grammar as **'**. A juncture is never followed by another juncture; a hyphen can be followed only by a letter (the hyphen, unlike the stress marks, never appears in final position).

We define a phonetic block as a sequence of letters and junctures (referred to collectively as "characters"), with the junctures giving information about syllable breaks and stress. A phonetic block is always intended to be pronounced without pause.

The terminal punctuation marks of Loglan are **.:?!;#**.

The comma **,** is an especially important punctuation mark, with the phonetic meaning of a pause in the flow of speech. A comma is always followed by whitespace followed by a phonetic block or alien text (ignoring any initial parentheses or double quotes appearing before the block or alien text). A phonetic pause is always denoted either by whitespace or a comma followed by whitespace; in some contexts the comma-marked pause is mandatory. Whitespace may or may not denote a pause; there are some contexts where whitespace must denote a pause.[4]

The double hyphen – is an independent punctuation mark, not a syllable break: it represents a pause, probably longer than the pause represented by a comma, and in some cases may be used in place of a comma where a pause is required. The ellipsis ... is an independent punctuation mark, not terminal punctuation, similarly representing a pause and occasionally usable in place of a comma.

Parentheses and double quotes may enclose Loglan or alien text under some circumstances described in the grammar. These are generally ignored

[3]The use of the hyphen to abbreviate the phonetic hyphen **y** found in earlier sources is not accepted here; in general, we do not pronounce punctuation.

[4]Uses of a close comma without a following space in earlier versions of Loglan are entirely replaced with uses of the hyphen as a syllable break.

for phonetic purposes. They are not pronounced: punctuation marks are never intended to be pronounced in the version of Loglan described here, though they may dictate pauses.

The letters have lowercase forms and uppercase forms and there is a capitalization rule applying to phonetic blocks. The formal capitalization rule is quite complex: the basic idea is that an uppercase letter will not appear immediately following a lowercase letter unless it is the first letter of a phonetic copy of a letter name (class `TAIO` to be discussed below; the letter names given above are words of this class), and a vowel may appear capitalized after **z** (for malicious reasons to be explained later). There is no restriction on capitalization resuming after a juncture. This allows the usual sort of capitalization, and also allows all-caps, and where junctures are present individual blocks of letters may be capitalized in different styles independently. The special treatment of letter names will be motivated in examples when these words appear.[5]

2.1.2 Pronunciation of Loglan letters

The regular vowels have typical continental European (not English!) pronunciation. The irregular vowel **y** may be pronounced with the indistinct schwa sound. Unstressed regular vowels do **not** become schwa (English and Russian speakers note!) A more distinct pronunciation for **y** (the vowel in English "look"[6] or the vowel written bI in Cyrillic) might be preferred.

The pronunciations of **b, d, f, g, k, l, m, p, r, s, t, v, z** require no special comment (except to note that **g** is always hard).

The letters **c, j** have unusual pronunciations, **c** being English "sh" in "ship", and **j** being the voiced version of the same sound, the "z" in "azure". English "ch" and "j" are the diphthongs **tc** and **dj** respectively.

The letter **h** has the typical English pronunciation, except in syllable final position, where it has the sound of "ch" in Scottish "loch". The latter pronunciation is actually permitted in other positions as well.[7]

[5]The approach to punctuation which we have taken has been driven partially by our own design decisions and partly by the punctuation and capitalization practices in the Visit to Loglandia.

[6]A suggestion of John Cowan.

[7]Syllable-final **h** did not exist in previous versions of Loglan. The pronunciation of **h** which is mandatory in final position might be preferred by speakers of some languages in other positions.

The letter **n** is pronounced as usual in English, and (as usual in English as well!) is pronounced as the "ng" in "sing" when it occurs before **g**, **k**, or the alternative pronunciation of **h**.

The vowels **i** and **u** are sometimes pronounced as English "y" and "w". as will be explained below.

The consonants **l, m, n, r** are sometimes syllabic ("vocalic"). When they are used syllabically, they are always doubled.[8]

We neither reject nor support complex alternative schemes of pronunciation of the language outlined in older sources; the alternatives we propose here seem sufficient.

2.1.3 Alien text

We describe the rules for embedding alien text in Loglan.[9] For such text, rules of pronunciation are not supplied by Loglan. It is required that alien text (however it is pronounced) be preceded by a pause (regarded as part of the alien text) and followed by a pause or end of text or speech (not regarded as part of the alien text): a pause may be expressed either by whitespace or by a comma or terminal punctuation (terminal punctuation only after the alien text, of course) followed by whitespace, and end of text or speech simply by end of text or by terminal punctuation. The body of the alien text between the initial pause and the final pause or end may be text not containing double quotes enclosed in double quotes, or it may consist of one or more blocks of text excluding commas, spaces and terminal punctuation marks, separated by the word **y**, which must be preceded and followed by pauses in speech, which are independently expressible by whitespace or by a comma-marked pause. Examples are **"War and Peace"** and **War y and y Peace**. When alien text is enclosed in quotes, occurrences of **y** between pause-separated components of the alien text may be omitted in writing but

[8]The rule that syllabic continuants must be doubled forces changes of spelling in names in legacy Loglan text in many cases. Syllabic consonants in borrowed predicates were already doubled, and Brown suggested in Loglan 1 that this might be a good rule to adopt in general.

[9]The model for these rules is actually the final state of the rule for Linnaean names with **lao** (now foreign names in general) given in the late 90's. We require that the occurrences of **y** there suggested merely for speech also be expressed in writing. The use of double quotes is a novelty but seems natural. The strong quotation scheme of 1989 Loglan is abandoned, essentially by giving **lie** the same phonetic grammar as **lao**.

must appear in speech: the two examples are pronounced in the same way. Some contexts require double quoted alien text in writing.

Alien text is always preceded by one of the alien text markers **hoi, hue, lie, lao, lio, sao, sue**, whose grammar and semantics will be discussed below. Alien text marked with **hoi** or **hue** must be double-quoted. The parser identifies blocks of alien text by looking for these markers (some of the markers have multiple functions and will not always be followed by alien text).

Examples of alien text in Loglan utterances will appear when we discuss the grammatical constructions that use them.

2.1.4 Vocalic diphthongs

In this section, we describe two-letter forms which may appear as the "vowel" component of a syllable.

The consonants **l, m, n, r** we call *continuants.* A doubled continuant **ll, mm, nn, rr** represents a syllabic continuant, which may serve as the vocalic component of a syllable. A syllabic continuant may not be followed or preceded by another instance of the same continuant without an intervening pause in speech.

We now consider how to pronounce sequences of regular vowels not separated by junctures. The issue is how to resolve such a sequence into syllables. The irregular vowel **y** is usually a single syllable, except for occurrences of syllables **iy** and **uy** in rare structure words, which will be discussed later.

We first consider sequences of two regular vowels. Some of these sequences are mandatory monosyllables, some are optional monosyllables, and some cannot be read as monosyllables.

There are four mandatory diphthongs **ai, ei, oi, ao**. The diphthong **ao** has the irregular pronunciation of "ow" in English "cow". The pairs of letters **ai, ei, oi** are not mandatory diphthongs when followed by **i** without an intervening juncture: **aii** is grouped **a-ii**.

The forms **a-i, e-i, o-i, a-o** (the hyphens may be replaced with other junctures), called broken monosyllables, can only occur in names (we do not regard **a-ii** and its kin as containing broken monosyllables, but here we are talking about more than two letters). One may write any other pair of regular vowels separated by a juncture, with the effect of enforcing the two syllable pronunciation (where it is optional).

There are six optional diphthongs, made up of **i** or **u** followed by a regular vowel. (forms **iy** and **uy** also occur in special contexts, to be discussed later). If these are pronounced as a single syllable, initial **i** is pronounced as English "y" and initial **u** is pronounced as English "w". The disyllable pronunciation can be compelled by writing a syllable break. Monosyllabic **iu** cannot be followed without a juncture by **u** and monosyllabic **ui** cannot be followed without a juncture by **i** (so, for example, if **iuu** is encountered it will be read **i-uu**). As a rule, the speaker has a choice when presented with an optional monosyllable of pronouncing it as one or two syllables; sometimes the context forces one of the pronunciations.

Other pairs of adjacent vowels are pronounced as two separate syllables; the use of a glottal stop to separate the components of a disyllabic vowel pair is permitted, but not expressed in the orthography. The glottal stop is **not** allowed as an allophone of the pause phoneme; all required pauses must be distinct, if sometimes brief.[10]

A pair of identical adjacent vowels not pronounced as a monosyllable has the characteristic that one of the vowels must be stressed and the other unstressed. This always holds for **aa, ee, oo** and sometimes holds for **ii**, **uu** (special rules stated above are designed to encourage pronunciation of the latter two pairs as monosyllables whenever possible!)

For a three-vowel sequence appearing in a predicate or name word, the general rule is that formation of monosyllabic **ii** or **uu** is the highest priority (so in **aii**, forming **ii** wins over forming **ai**, which in this context is not a mandatory monosyllable anyway, producing **a-ii**), followed by formation of a mandatory monosyllable (recalling that **i**-final mandatory monosyllables are not followed by **i**; **aoi** is grouped **ao-i** and is not considered to contain a broken monosyllable): e.g., **aiu** is grouped **ai-u**), followed by formation of an optional monosyllable (which is often an optional preference for the speaker; the parser does exercise this preference). An extreme example of speaker freedom is **iue**, which the parser will resolve into two syllables **iu-e** (choosing to group the first two when both pairs have the same precedence) but which the speaker can resolve into two or three syllables in any of the three possible ways.

[10]Previous versions of Loglan do not allow the glottal stop to appear medially in disyllables (we allow it but also allow the traditional Loglan pronunciation, a smooth glide from one vowel to the other); previous versions of Loglan allowed the glottal stop as an allophone of pause, and we do not. Lojban uses the h sound medially in disyllables, which would be allowed for a Loglan speaker who chose always to use the alternative pronunciation of **h**.

We present a formal rule[11] for reading the next syllable from a sequence of regular vowels of any length written without junctures, which is used in resolving predicates and names into syllables. A mandatory diphthong is read as the first syllable if it is present (recalling that if the pair of vowels ends in **i** and is followed by another **i** it is not a mandatory diphthong); a single vowel is read if it is not initial in a mandatory diphthong and the next two vowels form a mandatory diphthong; if neither of the previous two cases holds an optional diphthong is read by preference by the parser (though a disyllabic reading is permitted); as the final option a single vowel is chosen, subject to the rule that **i** or **u** (when not part of a diphthong) cannot be followed by an intervening juncture and a consonantal occurrence of the same vowel (this situation will cause parse failure). The process of resolution of the first syllable from a stream of vowels is repeated until the stream of vowels is completely resolved into syllables. This rule may look forbidding, but it should be noted that sequences of four or more vowels are quite rare in Loglan predicates or names, so the two and three vowel accounts will usually be quite enough.

There is a separate rule, used in resolving certain structure words, in which a sequence of vowels of even length is parsed into vowel pairs, each of which is read as monosyllable or disyllable as the rules require or permit. There is a further special rule for certain structure words with three-vowel sequences, which does not conform with the rule stated above for resolving vowel sequences in predicates and names, which will be stated when these structure words are described.

When the vowel component of a syllable is read, this will be either a syllabic continuant, or **y**, or a vowel or vowel diphthong chosen using the appropriate one of the rules above.

One should note that the rules presented here are not of interest to readers and writers, speakers and listeners, very directly; but they are certainly of interest to word makers, and might briefly be of interest to a dictionary reader encountering a word for the first time. Such phonetic rules exist in natural languages, whose speakers are not necessarily even aware of them; one could imagine that the native Loglander, though her speech will conform perfectly to the rules stated above, will not know much about them unless

[11]The formal rule for reading long sequences of vowels in names appearing in Loglan 1 is incredible, as it requires indefinite lookahead; of course it was also really intended only for use with three or perhaps four vowels.

she is a grammarian!

2.1.5 Consonant grouping

There are different rules for syllable-initial and syllable-final consonant grouping. It is worth noting that consonant grouping only occurs in regular Loglan text in predicates and names. Syllables with final consonants also occur only in predicates and names.

These are governed by two sets of phonetic rules. There is a list of permitted initial pairs of consonants[12]. The initial group of consonants in a syllable consists of a single consonant, or a permissible initial pair of consonants, or a triple of consonants in which each adjacent pair of consonants is an initial pair.[13] We refer to a pair of consonants which would be a permissible initial pair if an intervening juncture were removed as a "broken initial pair".

There is a list of forbidden medial pairs[14] and a list of forbidden medial triples[15]. These cannot occur even if broken by a juncture.

There can be one or two final consonants in a syllable, which cannot be part of a forbidden medial pair or triple whether together (if there are two of them) or combined with consonants taken from the beginning of the following syllable. A pair of final consonants cannot be a non-continuant followed by a continuant (this appears to be pronounceable only as a separate syllable). A final consonant cannot be followed by a regular vowel or a syllabic continuant, even with an intervening juncture (in other words, such a consonant should be read as part of the following syllable).[16]

A consonant in either of these sorts of groups which is a continuant cannot

[12]The initial pairs are **bl br ck cl cm cn cp cr ct dj dr dz fl fr gl gr jm kl kr mr pl pr sk sl sm sn sp sr st sv tc tr ts vl vr zb zl zv**

[13]The rule for initial consonant groups appears in Notebook 3.

[14]The impermissible medial pairs consist of all doubled consonants, any pair beginning with **h**, any pair both of which are taken from **cjsz**, **fv**, **kg**, **pb**, **td**, any of (**fkpt**) followed by either of (**jz**), **bj**, and **sb**.

[15]**cdz**, **cvl**, **ndj**, **ndz**, **dcm**, **dct**, **dts**, **pdz**, **gts**, **gzb**, **svl**, **jdj**, **jtc**, **jts**, **jvr**, **tvl**, **kdz**, **vts**, and **mzb**

[16]The rules forbidding final consonants from participating in illegal medial pairs or triples are found in our sources. The rule forbidding a pair of final consonants from being a non-continuant followed by a continuant seems quite natural but is ours; no word was proposed that violated it, in any case. Other rules that we state depend on a precise definition of the syllable, which appears nowhere in Loglan sources, although the notion of syllable is important in the definition of borrowed predicates in Notebook 3.

be adjacent to another copy of the same continuant, within or without the cluster, even if separated by a juncture. An initial consonant triple cannot be followed by a syllabic continuant at all.

2.1.6 The Loglan syllable

A Loglan syllable consists of three parts.

There is an optional initial group of one, two or three consonants governed by rules stated in the previous subsection.

This is followed by the mandatory vocalic component of the syllable, which is either a pair of identical continuants, a single regular vowel, a vowel diphthong, or **y** (**iy** or **uy** occur only in syllables (C)**iy** and (C)**uy** which are directly allowed as units in structure words but not supported in the formal syllable definition).

This is followed optionally by one or two final consonants, for which rules are stated above, with the additional remark that unless the syllable is of the shape CVC with the vowel regular, no final consonant in the syllable (neither of them, if there are two) may be readable as standing at the beginning of a following syllable (in other words, except in the case of CVC syllables, the automatic placement of syllable breaks where an explicit juncture is not present is as early as possible; but a CVC syllable is preferred to a CV syllable where possible). Explicit junctures will override the preferred syllable breaks, but there are subtle rules about where explicit junctures can be placed: sometimes they will simply cause parse errors.[17]

It is worth noting that previous versions of Loglan had no official formal definition of the syllable, though the syllable did play a role in the definition of some word classes.[18]

[17]The subtleties have to do with the fact that a borrowed predicate cannot resolve into djifoa (see below for these terms); an apparently legal borrowing predicate written with explicit junctures will be rejected if moving some of the junctures would create a legal complex predicate. These are issues which mostly affect the word designer. If you are trying to write a complex predicate from the dictionary with explicit syllable breaks, make sure that the breaks you supply conform with djifoa boundaries and these issues will not arise.

[18]The lack of felt need for a formal definition of the syllable may have come from the fact that structure words and complex predicates resolve into units which are not themselves necessarily syllables, but which are expected to conform with syllable boundaries; it is with the introduction of borrowed predicates that a precise notion of the syllable became essential to someone who wanted to parse words, and once this notion was in hand, it

2.1.7 Pauses and whitespace: general principles

A pause is always expressed as either a comma followed by whitespace (which must be followed by a phonetic block) or simply whitespace, which must be followed by a phonetic block. The former is always a pause; the latter may sometimes not be a pause.

Whitespace at the beginning or the end of alien text must represent an actual pause.

Whitespace after a consonant and/or before a vowel must represent an actual pause.

Names are the only consonant-final words in regular Loglan text, and they must be followed by comma-marked pauses, terminal punctuation, or end of text, or by whitespace followed by another name word or the structure word **ci**.

Logical connective words of class A, some but not all of which are vowel-initial, and sentence connectives of classes I and ICA must be preceded by comma-marked pauses. The APA and IPA logical and sentence connectives, to be discussed below, and the ICA and ICAPA sentence connectives must be followed either by the suffix **fi** or a comma-marked pause. The issues in this paragraph are handled entirely in the grammar section.

Words quoted with **liu** must be followed by a comma-marked pause (or terminal punctuation or end of text).

If the final syllable of a structure word is stressed and it is followed by a predicate, it must be followed by a comma-marked pause. This rule is of course only enforced in our orthography if we actually write explicit stress.[19]

In general, certain lexicographic issues tend to force explicit comma-marked pauses. If a pause in a sequence of structure word syllables breaks a word, it must be explicitly comma-marked as a rule, since if it were written as mere whitespace, not pausing would cause a different interpretation of the utterance. There will be a discussion of multi-syllable structure words in the

became natural to require that names (which were just consonant final strings of phonemes in earlier versions of Loglan) be resolvable into syllables as well. The accuracy of our implementation can be gauged by the fact that almost all words in the dictionary parsed correctly when we ran a test, and the ones which did not parse had recognizable errors which needed to be fixed. It should be noted that we cannot have three final consonants in a syllable, and this is not uncommon in names. This can usually be fixed by doubling a continuant, as in **Hollmz**, **Marrks**, but some names may be found to be definitely foreign.

[19]This rule goes back to the beginnings of Loglan, but as no earlier parser had explicit indications of stress, there was never any occasion for an earlier parser to enforce it.

lexicography section which lays out the situations under which this issue can occur.[20]

The "false name marker" problem creates further need of explicit pauses, which will be discussed below.

This version of Loglan supports a form of orthography known as "phonetic transcript" in which no whitespace appears but comma-marked pauses. This means that we require that in every place where we can or must pause, it must be possible to replace whitespace with a comma-marked pause. It is mostly but not entirely true that every place whitespace is written is a place where one *can* pause: it is possible to create situations with the APA and IPA connectives in their legacy form where a whitespace that one can write cannot represent a pause, and there is a rule that one should not pause after the structure word **ci** before a consonant unless the pause is comma-marked. This whitespace can, however, be omitted. Whitespace which does not represent pauses can always be omitted, though in the case of whitespace after predicates, this may require the writer to insert explicit indications of stress so that the reader can tell where the predicate ends. Whitespace which cannot be omitted can always be replaced with an explicit comma-marked pause.

Because we have phonetic transcript, we do not need a special notation for expressing pronunciation.[21]

2.2 Phonetic word forms

2.2.1 The four forms, and general principles

There are four basic word forms in Loglan:

1. Items of alien text (with their preceding alien text markers), already described above.

2. Phonetic names (name words accompanied with their required preceding pauses or name marker words with intervening optional pause).

[20]In Lojban, apparently all structure word syllables are separate words, but this is not the case in Loglan.

[21]Brown's phonetic notation in the sources is *ad hoc* and reveals such things as very inconsistent notions about syllable breaks.

3. Structure words

4. Predicates, further subdivided into complexes and borrowings.

These classes of words have general characteristics which allow us to distinguish them We leave aside the case of alien text which we have already analyzed.

1. Name words are the only consonant-final words in Loglan (other than alien text). They are thus followed by pauses in speech (and usually by explicit pauses in writing). This makes the right boundary of a name word easy to recognize. One must also pause at the beginning of a name word, unless it is preceded by one of a limited class of name markers. There are few contexts in which a name word can appear without an immediately preceding name marker word, and if a name word happens to include a phonetic copy of a name marker word (a "false name marker") it *must* be immediately preceded by a name marker word (an intervening pause being permitted). Where a name marker word occurs which is not immediately followed by a name word but followed by a name word starting later, a comma-marked explicit pause (or terminal punctuation) must appear somewhere between the name marker word not serving as such and the following name word: this prevents pronunciation of the text in a way which causes everything between the name marker word and the end of the later name word to be construed as a single longer name word.

2. Predicates end with a regular vowel (so they are not names), are penultimately stressed (with qualifications to be stated later); this allows the right boundary of a predicate word to be recognized in speech, or in phonetic transcript), and contain adjacent consonants (in some cases the pair of consonants may be separated by **y**). The left boundary of a predicate is determined by the fact that it must begin CC or (C)V^{n}C(**y**)C. in the latter case with some conditions ensuring that the (C)V^n cannot be construed as a structure word. Predicates can more rarely begin (CVV**y**)^n((C)V^m)CC.

3. Structure words (Loglan **cmapua**) are not names or predicates (actually some are semantically names or predicates, but this is a matter for the grammar). In addition, we specify that they resolve into phonetic

units of the shapes V, VV, CV, CVV (where the VV may be a monosyllable or a disyllable, and **iy**, **uy** are permitted), and the rare Cvv-V, where the vv is a monosyllable (mandatory or optional, but in any case pronounced as such). Further, a V unit may only occur initially, and any structure word which contains a VV unit consists entirely of VV units (except that we allow words of the shapes **no**-VV and VV-**noi**). A sequence of VV units is resolved into syllables by pronouncing each unit as one or two syllables as the grammar requires or permits. Note that the unit cmapua are not necessarily syllables, but their boundaries are syllable boundaries in a structure word. Where a structure word is followed by a predicate beginning with CC, stressing its last cmapua unit might create the possibility of reading the last cmapua unit and the first syllable of the intended predicate word as a predicate: to avert this, we require that a finally stressed structure word must be separated from a following predicate word (not just a CC-initial one) by a comma-marked pause.

It should be noted that the classes of words here should be qualified as phonetic names, phonetic predicates, and phonetic structure words, as there are cases where "words" which are phonetically of one of these shapes are used in a way associated with one of the others.

2.2.2 Phonetic Names

We distinguish between a name word, such as **Djan**, and a phonetic name, such as **la Djan Braon**, which comes equipped with the name marker word or initial pause that a name word requires in its context, and may contain more than one name word after the name marker.

A name word is a phonetic block which resolves into syllables, the last of which ends in a consonant (possibly with a final stress).

A possible name word is a name word, or a name word modified by insertion of whitespace at junctures preceded by a vowel and succeeded by a consonant (so that the whitespace does not necessarily represent a pause).

A marked name is a name marker word followed by a consonant initial name word, possibly with intervening whitespace between the two.

A falsely marked name is a name word with a proper final segment which is a marked name: that is, it is a name word with a false name marker in it. Notice that a phonetic occurrence of a name marker word is not a false name marker unless what follows it is a consonant-initial name word.[22]

The name marker words are **la, hoi, hue, ci, liu, gao, mue**. A subtle point is that **ci** is only a name marker when followed by a pause (an explicit comma-marked pause or whitespace followed by a vowel): this allows us to avoid difficult-to-predict needs for pauses after the many uses of **ci**. It does mean that when whitespace is written after **ci** before a consonant, we presume that the speaker does not pause.

A phonetic name (including its name marker or preceding pause if there is one) is of one of the following kinds:

1. a marked name as described above (a name marker followed by possible whitespace followed by a consonant-initial name word).
2. a vowel initial name word which is not a falsely marked name, or a comma-marked pause followed by a name word which is not a falsely marked name.
3. a name marker followed by optional whitespace or explicit pause followed by a name word, with the additional proviso that the optional whitespace or pause must be present if the name word is vowel-initial.

To any of these, a series of name words marked with **ci** and prenames which are not falsely marked, separated by whitespace, may be appended as part of the phonetic name, so **la Djan Braon** is a phonetic name, and so is **la Pierr ci, Laplas**. In the last example one pauses both before and after **ci**; the second comma must be written, and the use of **ci** is necessary because **Laplas** is a falsely marked name.

It is then required that this be followed either by an explicit pause, terminal punctuation, end of text, or whitespace followed by **ci** followed by a predicate of class `predunit`, a peek forward at the grammar. Note that the

[22]In early versions of Loglan, falsely marked names were simply forbidden, but **la** is very common. Later, they were admitted and some effort was made to avoid problems with them. The idea that a falsely marked name must be marked appeared in the context of implementation of serial names (falsely marked names in a serial name had to be marked with **ci**; we required after 2013 that predicate components of serial names be marked with **ci** as well to avoid the need for two pause phonemes to avoid confusion of serial names with sentences.) We extended the idea that falsely marked names must be marked to all contexts, and in addition reduced the distribution of unmarked names to very few contexts by forbidding unmarked vocatives.

following explicit pause or punctuation or **ci** phrase is not part of the phonetic name: this is information about the context in which a phonetic name can appear.

Names may contain explicit junctures, including ones which form broken monosyllables, and junctures may be required features of name words: **Lo-is** and **Lois** are different names.[23]

2.2.3 Phonetic structure words (cmapua)

Phonetic structure words are sequences of cmapua units as sketched above; we give more details.

Cmapua units are of the shapes V, VV, CV, CVV, Cvv-V, where vv stands for a monosyllable and VV (in VV and CVV units) includes **iy**, **uy**. **y** is also accepted as a V unit.[24]

A phonetic structure word is a string of cmapua units. A cmapua unit not of VV form cannot be followed by a vowel, even with an intervening juncture: this helps to enforce the condition that vowel-initial words must be preceded by pauses in speech, represented at least by whitespace.

Each cmapua unit is restricted by lookahead tests for other classes. A cmapua unit cannot be an alien text marker actually followed by alien text. A cmapua unit cannot be an occurrence of **li** or **kie** which actually stands at the beginning of a quotation or parenthetical free modifier (for the uses of these words, see the grammar section). A cmapua unit cannot be a name marker followed with optional pause by a possible name word (this excludes both actual phonetic names and strings which could be misread as phonetic names by ignoring instances of whitespace one of which should be made an explicit pause).[25] A cmapua unit cannot stand at the beginning of a legal

[23]This goes back to previous versions of Loglan, but we use hyphens instead of close commas.

[24]The practical reason for allowing **y** to occur above is to support names of the letter **y**, legacy **yfi** and modern **ziy** (pronounced “zyuh”!). It seemed more principled to install general phonetic conditions that allowed these forms than to allow them individually by fiat.

[25]This is our definitive solution to the false name marker problem. Difficulties created by the markers other than **ci** should generally be easy to anticipate, by following style rules such as “always pause after a predicate name”. The word **ci** presented special difficulties as a name marker because it has a wide variety of uses some of which have nothing to do with names. Viewing it as a name marker only when followed by a pause seems to be the final refinement of our solution.

predicate (the parser does a lookahead test which identifies strings which can only be predicates if they are grammatical and not possible phonetic names; we describe this test below at the beginning of the discussion of predicates).

A cmapua unit cannot be stressed and then followed by optional whitespace and the start of a consonant-initial predicate (as detected by the test above).

We then provide a phonetic test for the logical and sentence connective classes which must be preceded by a pause. A phonetic connective starts possibly with whitespace followed by possibly by an occurrence of **no** (not starting a predicate) followed definitely by a regular V syllable or **ha**[26], **nuu** (not starting a predicate), not followed by a vowel, and not followed by **fi**, **ma**, or **zi**, which would make a V unit into a legacy letteral (none of these starting a predicate).

We can now describe a phonetic structure word. It takes one of five forms.

1. a VV unit (here and in all clauses here including **iy**, **uy**) followed by **noi** (**noi** not starting a predicate).

2. **no** (not starting a predicate) followed by a VV unit

3. a sequence of VV units

4. a regular or irregular V unit

5. an optional regular or irregular V unit followed by a sequence of one or more consonant-initial cmapua units

A cmapua unit absorbs a following juncture.

Each cmapua unit is blocked from being followed by a vowel without intervening whitespace or by optional whitespace then a phonetic connective; this forces explicit pauses before the logical and sentence connectives.

The phonetic structure words defined here have boundaries dictated entirely by phonetic convenience; the actual boundaries of cmapua words in the proper sense are dictated by rules stated in the lexicography chapter. Some words which appear in other structures, such as the name markers, alien text markers, and **y**, and some others, are from a lexicographic standpoint structure words and do look like them phonetically.

[26]Note that we are ruling here that **ha** and its derivatives must be preceded by a pause.

2.2.4 Primitive Predicates and Combining Forms (djifoa)

The basic "native" predicates of Loglan are of the five letter forms CCVCV and CVCCV. The original stock of native predicates was generated by a rather *ad hoc* statistical comparison with words in major natural languages on which we have no intention of commenting, as we expect it never to be used again.

Each of the native predicates has one or more combining forms (originally called "affixes", a deprecated usage; now usually called *djifoa*, the Loglan word for these forms).

Each five-letter native predicate has a djifoa formed by replacing its final vowel with **y**. This does mean that five letter predicates which have the same final vowel must be semantically very closely related (words for animals and languages can be given fine shades of meaning by adjusting the final letter; we do not intend to create further declensions of this kind, but we see nothing wrong with the ones we have).

In addition, many five-letter djifoa have one or more than one associated three letter djifoa, of one of the forms CVV, CVC, or CCV, which is formed by choosing three letters from the five letter djifoa in order of their occurrence. The process of choosing these djifoa is not likely to be modified or extended at this point, though there is some tension about the ones with doubled vowels which force stress.

There is also a short list of three-letter djifoa built from CV cmapua, by appending **r**, which we supply:

fer: from **fe**, five

for: from **fo**, four

fur: from **fu**, 3d place passive

jur: from **ju**, 4th place passive

ner: from **ne**, one

nir: from **ni**, zero

nor: from **no**, logical negation

nun: from **nu**, 2nd place passive, before**r**

nur: from **nu**, 2nd place passive, not before **r**

por: from **po**, state particle

rar: from **ra**, all

rer: from **re**, most of

ror: from **ro**, many of

ser: from **se**, seven

sor: from **so**, six

sur: from **su**, at least one of, some

ter: from **te**, three

tor: from **to**, two

ver: from **ve**, nine

vor: from **vo**, eight

Further, every CV cmapua unit has a corresponding CV**h** djifoa. CVV cmapua may be extended with **(h)y** (not with **n** or **r**) and used as djifoa: thus **zaiytrena**, A-train. This must be done with care as there may be djifoa derived from cmapua of the same shape.

Loglan complex predicates (native compound predicates) are built from sequences of these djifoa (in which the last item may be a full primitive or borrowed predicate). There are also borrowing djifoa built from borrowed predicates, which we discuss in the next section.

It is necessary to supply additional phonetic glue so that sequences of these djifoa actually can produce predicate words. Each three-letter djifoa has an alternative form with suffixed **y**. CVC**y** djifoa can be broken into syllables either as CVC-**y** or as CV-C**y**. CVV djifoa in initial position will "fall off": so CVV**r** is available as an alternative form (which will form a consonant pair with the following djifoa or predicate word), and CVV**n** is available as an alternative form when it is followed by **r**. Other problems are that CVC djifoa may not occur in final position, and CVV djifoa which have a doubled vowel can only occur in final or in penultimate position, because a predicate word can only contain one stress in a penultimate position. We propose for CVV**y** the alternative form CVV**hy**: a predicate starting this way will not be confused with any other kind of word, and this should be easier to pronounce distinctly.

A pronunciation difficulty with CVV**r** extended djifoa (at least for English speakers) is fixed by a permission which will probably seldom be expressed in writing, but can be: CVV**r** can take the alternative form CVV-**rr** when the VV is a mandatory monosyllable. If this is stressed, the stress falls on the VV and the vocalic continuant is an additional unstressed syllable before the final unstressed regular syllable.

Broken djifoa forms are also available to the parser, obtained from legal djifoa by inserting junctures (C-CV or C-VV or CV-C, for example). These are used to enforce the condition that borrowed predicates cannot decompose into djifoa, and it should not be possible to convert a complex (especially one which is illegal for reasons peculiar to complexes) to a legal borrowed predicate by moving junctures around.

The general point about syllable breaks is that while djifoa are not syllables, the boundaries between djifoa will be syllable breaks in a legal complex. Internal breaks are sometimes optional: a CVV djifoa with an optional monosyllable has two possible forms. The CVCCV five letter predicates may admit two forms CVC-CV and CV-CCV if the medial CC is an initial pair. The parser prefers the first form for technical reasons; it can be coerced by writing an explicit syllable break, and the latter version is often easier to pronounce.

2.2.5 Phonetic predicate words: general principles, and recognizing the beginning of a predicate

A phonetic predicate word ends with a regular vowel (so it is not a phonetic name), contains an adjacent pair of consonants (so it is not a structure word), and has penultimate stress (with the exception that an additional unstressed syllable with **y** or a vocalic continuant may intervene between the stressed and the final syllable), so that one can tell where it ends.

The part of the predicate before the consonant pair can be null (the predicate may start with an initial pair or triple of consonants). This will be followed by a regular vowel, and there are no CC(C)V(V) predicates, in which the initial consonant group is followed just by one or two vowels.

The initial segment before the consonant pair can be an optional consonant followed by one to three regular vowels[27] This is the only alternative which can occur in a borrowed predicate. This can be ensured if the string starting with the consonant group doesn't satisfy the conditions given above to be the start of a predicate: the predicate must then begin with the initial (C)V(V)(V) (it cannot begin with part of the vowel sequence because one must pause before a vowel-initial word). It can also be ensured if the final syllable of the initial (C)V(V)(V) is stressed: if it were the end of a cmapua, it could only be followed by a predicate, and one cannot have a stressed cmapua followed by a predicate without pause. There is a further technical issue, which is explained below in the discussion of borrowing djifoa: for technical reasons, if the (C)V(V)(V) is not of one of the forms CV or CVV, the consonant group cannot be an initial pair followed by a regular vowel.

In a complex, there are other possibilities. A complex might start with one or more CVV**y** djifoa; no other sort of word can start this way. It might start with a CVCC**y** djifoa; no other sort of word can start this way. It might start with a an extended CVC djifoa: CVC**y**. Again, no other Loglan word can start in this way.

We describe a lookahead test: a string which is already known not to be a possible phonetic name cannot be anything but a predicate (or ill-formed) if one of the following things are true (and one of these things will be true of any actual predicate):

1. It begins with a permissible initial group of two or three consonants and is followed by a regular vowel, but not by one or two vowels (with possible junctures) followed by a non-character, nor by a stressed vowel followed by a vowel not in a diphthong (short words of the shapes CC(C)V(V) are not predicates). Junctures may appear after the vowels. This clause of the test ignores any junctures which may appear in the initial group of consonants (to support its use in following clauses).

2. It begins with CV(V) followed by a consonant group, with either the final syllable in the CV(V) [which might extend to a juncture in the consonant group] stressed or the part of the word beginning at the consonant group not meeting the test to start a predicate above. Junctures may appear after the vowels, and there might be a juncture in

[27]Previous versions of Loglan have allowed arbitrarily long sequences of vowels after the initial consonant in this case, but these have never been used and I like the bound on lookahead in this test obtained by forbidding more than three vowels.

the consonant group, which will be ignored in testing whether it begins a predicate.

3. It begins with (C)V(V)(V) followed by a consonant group which is not an initial pair (even one broken by a juncture) followed by a regular vowel, with either the final syllable in the (C)V(V)(V) [which might extend to a juncture in the consonant group] stressed or the part of the word beginning at the consonant group not meeting the test to start a predicate above. Junctures may be inserted after the vowels or there might be one in the consonant group, which will be ignored in testhing whether it begins a predicate.

4. It begins with a consonant and a regular vowel, followed by a regular vowel followed by **y** [or **hy**], or a consonant or pair of consonants followed by **y** (with possible intervening junctures). Both of these initial sequences are possible beginnings for a predicate complex, and what follows the CV^n could not start any legal word in that context.

The point of including this list is making it clear that it is fairly easy to see or hear the beginning of a predicate word (though the detailed description of the cases is admittedly annoying!)

2.2.6 Borrowed Predicates and Borrowing Djifoa

After all that about djifoa, we discuss borrowings first!

A borrowing must resolve into syllables. It must end in a regular vowel. Any explicit stress must be on the second-to-last syllable, not counting syllables with vocalic continuants (one such unstressed syllable may intervene between the stressed syllable and the unstressed final syllable). The end of a predicate is determined by either non-characters such as whitespace or punctuation, or by an explicit stress. A borrowing cannot be followed without intervening whitespace or explicit pause by a vowel, nor by optional whitespace followed by a connective. An explicit stress may force monosyllabic pronunciation on a last syllable which could otherwise be pronounced as a disyllable. There can be only one explicit stress in a borrowing. The deduced stresses in doubled vowels do not play a role in parsing borrowings, as disyllabic doubled vowels are forbidden in borrowings. It is permissible to write a borrowing with explicit junctures, but this cannot change the meaning of the word, and broken monosyllables are not permitted. A borrowing must parse correctly in the absence of explicit junctures.

The beginning of a borrowed predicate must pass the test for beginnings of predicates given above. Since borrowed predicates cannot contain **y**, this enforces the condition that there be a pair of adjacent consonants in a borrowed predicate. We reiterate that a borrowed predicate cannot have the shape CC(C)VV (which is of course not resolvable into djifoa, so could not be the shape of a complex predicate).

There are some restrictions on the phonetics of borrowed predicates. Borrowed predicates may not contain **y** or any doubled vowel other than monosyllabic **ii** and **uu**.[28] Borrowed predicates may contain syllables with vocalic continuants. These never follow a vowel and so far never precede a consonant, and such a syllable is always medial (not first or last). There may not be two such syllables in succession, and such a syllable cannot be stressed. The point of such syllables is that they cannot occur in complexes (with one minor exception which cannot be confused with occurrences of syllabic continuants in borrowings: a CVV**r** djifoa may be expressed as CVV**rr**, in which the syllabic continuant follows a vowel). A borrowed predicate may not resolve into djifoa, including resolutions involving broken forms in which junctures are misplaced.

Borrowing djifoa are formed from borrowings by adding final **y** and moving the stress to the final syllable of the borrowing (still penultimate in the borrowing djifoa). It is permitted to stress the penultimate syllable in a borrowing djifoa and pause after the **y**, if what follows the borrowing djifoa contains a penultimate stress. Thus one may pronounce **bakteriyrodhopsini** as **bakteri'y, rodhopsi'ni**, but one may not pronounce **iglluymao** with a pause. The stress shift is a strong signal that one is not saying **bakte'ri** but its djifoa.[29]

The reason for the difference between the treatment of CV(V) and other (C)V(V)(V) prefixes in the predicate start rules is caused by the danger

[28]Forbidding doubled vowels in borrowings was an action of ours: admitting them made reasoning about the penultimate stress in a borrowing more difficult. Exactly one predicate **alkooli** had to be changed to the better **alkoholi**.

[29]Nothing in this situation is due to me! The strange provision to pause after a borrowing djifoa is in Loglan 1. The definition of borrowing djifoa we use was given in the 1990's. I do not know if anyone noticed the stress shift caused by the change in the definition of borrowing djifoa, but it is all a logical consequence of the way things stood at Brown's death.

that a borrowing djifoa for a (C)V^nCCV predicate might turn into a cmapua followed by a stressed CCV**y** djifoa when a borrowing djifoa was formed (this requires the CC to be an initial pair, of course). This is not a danger when the consonant is present and $n = 1$ or 2 because a borrowing will not be of the shape CV(V)CCV with the CC initial. Thus a string (C)V^nCCV with the CC an initial pair is not allowed to start a predicate unless the initial consonant is present and n is 1 or 2.

2.2.7 Complex Predicates

A complex predicate is formed from a sequence of djifoa in which the final element may be a full primitive or borrowed predicate (and if it is a djifoa may not be CVC, nor may it be extended with **r**, **n** or **y**, nor may it contain **y** at all). A complex may not be formed entirely from CVV**y** djifoa and a final CVV (this prevents forms without adjacent consonants; adjacent consonants may be separated by **y** as in **mekykiu**). A complex must pass the predicate start test. CVV djifoa may need to be extended with **r**, **n**, or **y** when they appear in initial position to keep from being read as cmapua. CVC djifoa in initial position of a complex not of one of the six-letter forms CVCCVV or CVCCCV must be extended with **y** if the final consonant of the djifoa and the following consonant would form an initial pair (a juncture does not affect this): this prevents the complex from being read as a CV djifoa followed by a borrowing.[30] Thus **ficynirli**, mermaid. A CVC djifoa may need to be extended with **y** to prevent formation of an illegal consonant group with the following consonant. Thus **mekykiu**, eye doctor. Regular djifoa must be extended with **y** when they appear before borrowing djifoa: since borrowings cannot contain **y**, this gives us precise information about their boundaries.

Complexes must have penultimate stress among those syllables not containing **y**: an unstressed syllable containing **y** (or a syllable **rr** serving as glue to a stressed CVV) may intervene between the stressed syllable and the unstressed final syllable. The parser does know about the doubled vowel stress rule, and will not accept a complex with a CVV with a doubled vowel unless one of the vowels can carry the penultimate stress. If a CVV with doubled

[30]This rule superseded the historical **slinkui** test, which prevented the CV from falling off the front of a CVCC... complex with the CC an initial pair by forbidding the formation of borrowed predicates obtained by extending a complex initially with a single consonant forming an admissable pair: instead of forbidding **paslinkui** in favor of **pasylinkui**, **paslinkui** was permitted and **slinkui** was forbidden to be a borrowing.

vowel is followed by a CVV with optional monosyllable, the monosyllabic pronunciation is forced; otherwise a CVV with optional monosyllable gives the speaker a choice about where to place the stress. The parser determines where a predicate ends either by the occurrence of explicit stress or by the occurrence of a non-character from which it back-figures the location of the stress. The only stresses in a predicate are its penultimate stress and optionally stresses in its borrowing djifoa (mandatory if the borrowing djifoa is followed by a pause).

A complex may not be followed without intervening space (a juncture doesn't help) by a vowel. A complex may not be followed by whitespace followed by a connective.

An alternative formulation allows the formation of a complex from a sequence of cmapua units and predicate words in which the last item is a predicate word, with successive items separated by the "word" **zao**, optionally flanked on either side by whitespace or comma-marked pauses. This form might be used to avoid borrowing djifoa. I can also imagine its use to clarify the meaning of a complex by replacing its constituent djifoa with the corresponding predicates in full.[31]

Our general view is that the replacement of a djifoa in a complex by another djifoa for the same predicate or even by the full predicate linked with **zao** gives another form for the same predicate word: such forms should not appear as separate dictionary items.[32]

2.2.8 Phonetic Quotes and Parenthetical Expressions

A phonetically valid quoted utterance begins with **li** and ends with **lu** with optional pauses after **li** and before **lu** and with the intervening text, which must be a phonetic utterance, optionally enclosed in double quotes (the quotes being between **li** and **lu**). Replace **li** and **lu** with **kie** and **kiu** and optonal use of quotes with optional use of parentheses (opening with an open parenthesis, closing with a closing parenthesis, and you have the rule for spoken parenthetical expressions.[33]

[31]This is a proposal of John Cowan.

[32]For example, we think the word **heirslicui**, molasses, presents difficulties for an English speaker and she might want to say **hekryslicui** instead; this is precisely the same word.

[33]The use of punctuation here is a proposal of ours: it is quite natural but did necessitate the phonetic parser knowing about these forms.

We give examples: **li, la Djan, lu**; **li "la Djan", lu**; **kie (ji cluva mi) kiu**.

2.2.9 Phonetically valid utterances

A phonetically valid unit phonetic utterance is a phonetic name, phonetic structure word, marked alien text item, phonetic predicate word, quoted or parenthesized expression, hyphen, or ellipsis, possibly with initial whitespace. A phonetic utterance is a sequence of unit phonetic utterances, explicit comma pauses, and items of terminal punctuation. Any Loglan utterance must be a phonetic utterance: of course it must also be grammatically correct, a matter for subsequent chapters. [34]

It is useful to be aware that the parser proceeds in effect in two passes: it checks an entire utterance for phonetic validity, then checks whether it is grammatical in another pass.

2.3 Historical and philosophical note

Loglan phonetics is to our mind rather weird and wonderful.

James Jennings has commented that the choice to recognize word classes by patterns of consonants and vowels in the first instance was the "original sin of the language". Perhaps so, but once this sin was committed, it could not be undone without discarding everything and creating a different language.

1975 Loglan had a very simple procedure for resolution of words, but a method for construction of new predicates which was ultimately found unsatisfactory. The great morphological revolution which consisted in introducing complex predicates as predicates built from djifoa and borrowings as all the phonetically acceptable predicates which were not complexes made the definition of the predicate much more complicated, and more of a challenge for the parser builder.

The problem of false name markers also led to a certain amount of excitement, once it was decided that **la** was too common to ban from names.

We find Loglan phonetics charming as well as weird; the language has a definite phonetic flavor and avoids monotonous regularity. We would not

[34] We note that CCV djifoa are alse unit phonetic utterances, strictly so that they may be quoted with **liu**. CVV djifoa are also cmapua, and CVC djifoa are also names, so they can be **liu**-quoted without special ceremony.

say that it is always easy to pronounce, and its resemblance to a Romance language can be overstated: it does allow quite a lot of consonant clustering. A further charm is that the baroque rules, however arbitrary they may seem, actually work out as almost inevitable consequences of a fairly small number of design decisions. We hope that we have given some hint in our discussion of what these design decisions were.

Chapter 3

The Grammar Proper

In this section, we will present the grammar, pausing now and then to introduce word classes that are needed; we will give short lists of commonly used words in the grammar text and full word lists in an appendix.

3.1 Simple sentence shapes

It is an interesting question where to start. We will begin with the basic Loglan sentence, and work upward to more complicated utterances and other kinds of utterance fragments, and downward to sentence components.

The simplest kind of Loglan sentence is examplified by **La Djan, kamla** (John comes) and **La Djan, donsu le bakso, la Meris** (John gave Mary the box). This is an S(VO) sentence: each of these consists of a subject (**la Djan** in both cases), a species of noun phrase, followed by a verb phrase, which is a verb (**kamla**) in the first, **donsu** in the other), followed optionally by a list of "objects" (noun phrases, none in the first example, two, **la Meris** (Mary) and **le bakso** (the box) in the second.

We pause here to discuss grammatical terminology. We have used the words "noun" and "verb" though Loglan actually has no such word classes. It does however have those functional roles, and we will use (hopefully with care) these words to help communicate what is going on in the grammar. The words **kamla**, **donsu**, and **bakso** all belong to the same word class, Loglan predicates, and can appear in any of the roles. **Ti bakso** (this is a box) is a sentence in which the very nounish (to the English mind) word **bakso** appears...as the verb! What we call a "noun phrase" can also be called an "argument" (terminology taken from logic and also already used in Loglan grammar). Loglan grammarians have up until now used "predicate" indifferently for predicate words and for what we call "verbs" and "verb phrases"; we will continue to use "verb" and "verb phrase" as grammatical terms.

We discuss the difference between this kind of basic sentence and atomic sentences of predicate logic. An atomic sentence of predicate logic is of a form Mx (x is a man), Bxy (x is bluer than y), $Gxyz$ (x gives y to z). There is a predicate (M, B, G) and argument lists (x, xy, xyz). A sentence like Bxy might parse $(B)(x)(y)$ or perhaps $(B)((x)(y))$; the predicate and the individual arguments might be components at the same level or the predicate and the argument list might be components (the list further resolving into individual arguments).

The Loglan parse is different, in a way which brings Loglan closer to natural languages: **da mrenu**: xM, **da blanu de**: $x((B)(y))$, **da donsu de di**: $(x)G((y)(z))$. The weird thing here, from the standpoint of a logician, is the very special role of the subject: the whole sentence breaks at the top into the noun phrase subject and the verb phrase containing the verb and a list of the second and subsequent arguments. The oddities of the way this breaks down become clearly important later when we discuss logically connected predicates.

3.1.1 A brief review of components we use in example sentences

To support examples, we should say something about the components of this sentence and what we are currently putting in for these components.

Any Loglan predicate word can play the role of the "verb". There are more complicated verbs than single predicate words, and we will see some possible additional complexities quite soon. A name such as **la Meris**, **la Djan Braon** is eligible to be a noun phrase (either a subject or an object). Pronouns such as **ti**, **ta** (this, that), or **da, de, di, do du** (pronouns referring to recently mentioned noun phrases by a scheme we will discuss below), or letter names (referring to recently mentioned noun phrases with the given initial), or noun phrases built from predicates such as **le mrenu**, (the man), are other possible arguments we may use before we have fully explained the range of possibilities for noun phrases.

3.1.2 Changes of argument order and omission of arguments

The sentence “I am better than you” is expressed **Mi gudbi tu**. The sentence “You are better than me” can of course be expressed **Tu gudbi mi**, but it can also be expressed **Mi nu gudbi tu**. **nu gudbi** is a verb, just as **gudbi** is. The effect of the particle **nu** is to reverse the first and second arguments. The particle **fu** interchanges the first and third arguments; the particle **ju** interchanges the first and fourth arguments. Compounds are possible: **nufunu** interchanges the second and third arguments, for example: **La Meris, nufunu donsu la Djan, le bakso** (Mary gave John the box). No one is going to carry out the transformation expressed by **nufunu** in three separate steps in their head during a conversation: it should be learned as a separate dictionary word. But if you work it out step by step, you will find that that is what it does. These contructions implement what would be “passives” in other languages: the Loglan term is “conversion”.

A variation implements “reflexives”. **nuo, fuo, juo** have the effect of eliminating the second, third, fourth argument, respectively, by supplying the subject as that argument. **La Meris, nuo donsu la Djan** (Mary gave herself to John) or **la Meris, fuo donsu le bakso** (Mary gave herself the box) exemplify this transformation. Compounds can be formed using the reflexives: for example **nufuonu** eliminates the third argument by identifying it with the second.

Any Loglan predicate word has a certain number of arguments, which have a certain order in the situation it represents, which can be seen in its dictionary entry. Without special contrivances, you cannot supply the predicate with **more** arguments. But you can supply it with fewer arguments. **Mi gudbi tu** means “I am better than you”. Just **Mi gudbi** means “I am good”, with the underlying assertion being “I am better than someone”. Another example **La Meris, donsu le bakso**: “Mary gave the box away (gave it to someone)”.

This can be combined with changes of argument order: **Mi nu gudbi**, “I am bad” (I am worse than someone); **La Meris, nufunu donsu la Djan**: “Mary gave (something) to John.”. The argument omitted is always the last one, but if one changes the order of the arguments, one can put an argument one wishes to omit in the last position.

If the reader wonders why we introduce this transformation of verbs here, they should note two things: as we will see later, this is one of the most tightly binding operations on verbs, and further, it acts exactly on the very simplest features of the structure of the simple Loglan sentence.

3.1.3 Tenses and variations

In this section we introduce tenses of the Loglan verb, which do not necessarily have anything to do with time. Tense is achieved using a structure word (either **ga** or a word of the PA class): the simplest examples are **na**, **pa**, **fa**. the present, past, and future tenses. A nontemporal examples is **vi** (here). There is also a null tense **ga**, which is used in situations where it is grammatically useful to have a tense but we do not actually want to say anything about the time, place or conditions of the assertion.

la Meris, pa cluva la Djan Mary loved John

Mi fa nufunu donsu tu I will give you something

La Ailin, vi danse Eileen dances here

Le mrenu ga sadji The man is wise (in general, no commitment to a particular time). Here the **ga** is grammatically a tense but it doesn't add anything to the semantics. It is needed, because as we will see later, **le mrenu sadji** is not a sentence, but a noun phrase, "the wise man".

We aren't introducing tenses at length: we actually need to introduce them in order to describe a further manipulation of basic sentences. Notice that **nufunu donsu** is tensed, rather than **fa donsu** being converted: the tense is much more loosely attached than the conversion operator. In fact, the tense attaches to the verb phrase as a whole rather than to the verb.[1]

3.1.4 Variations in sentence order

We can put the subject in a sentence after the verb in two ways.

The first kind of sentence we can produce has a tensed verb phrase with its objects (it might be tensed with **ga** strictly for grammatical purposes) followed optionally by **ga** then the subject:

Ga gudbi tu ga mi I am better than you

Ga gudbi tu There are better than you (here the subject is omitted!)

Ga donsu le bakso la Djan, ga la Meris Mary gave the box to John

[1]It is actually possible to convert a sort-of-tensed verb but it is tricky: **Mi nufunu ge donsu je fa gue tu**, in which quite a lot is going on which we will not explain yet!

Nia nu gudbi tu ga mi You are being better than me (combining conversion of the predicate with reordering of sentence components!) The tense word **nia** is the present progressive.

The second kind of sentence with subject delay consists of a tensed verb phrase with no objects followed by **ga** then all the arguments in the sentence.

Ga donsu ga la Meris, le bakso la Djan.

This allows us to achieve VOS and VSO word orders.[2]

We can put some objects before the verb, if we separate those objects from the subject with the particle **gio**.

Mi gio le bakso ga donsu tu I give the box to you (the tense **ga** is actually needed to keep from saying **le bakso donsu**.the boxy giver), or

Mi gio le bakso tu ga donsu

This supports SOV(O) sentence order.

The particle **gio** may optionally be used to set the subject apart from the objects in a VSO sentence:

Ga donsu ga la Meris, gio le bakso la Djan.

There is another device for modifying sentence order by bringing objects to the front, but this device cannot be properly introduced until after we discuss logically connected sentences.

3.1.5 Modifiers and tagged arguments

Tense, location and modal operators (the same words which can decorate verb phrases as tenses) can form *sentence modifiers* which are rather like additional arguments which can be supplied with any verb. In English grammar, these would be "prepositional phrases".

Relative modifiers and arguments (including the tagged arguments introduced below) are called terms. The subject and the list of objects in a verb phrase are both term lists (of slightly different kinds, as we will see). The subject is an arbitrary list of terms containing at least one argument and no

[2]It is a reform of ours to require that gasents (the Loglan jargon for subject-delayed sentences) must have either exactly one argument delayed or all arguments delayed. We want to avert listeners being forced to retroactively change their understanding of the meanings of arguments appearing earlier as in *****Ga donsu le bakso, ga mi tu** in which one would reasonably start out thinking that **le bakso** was the object being given (the second argument) but the presence of two arguments after **ga** (permitted in 1989 Loglan) forces the listener to revise this: the actual meaning of the sentence would be "I gave you to the box".

more than one untagged argument. It is important to notice that if the list of terms before the verb in a sentence does not contain any arguments, the sentence will be either a gasent (if tensed) or an imperative (if not tensed). The object list can contain no more than four untagged arguments (since there is no predicate taking more than five arguments).

Formally, a relative modifier is either a word of class PA (a tense, location or modal operator) followed by a noun phrase, followed optionally by the particle **guua**[3] or the general right closer particle **gu**, or simply a PA word followed optionally by **gu**. In the last kind of sentence modifier, you should suppose that the omitted argument of the PA word is the present situation in which the speaker is delivering their speech (the referent of the pronoun **tio**).

Lists of PA words will appear in the next section.

Examples of such modifiers:

Vi la Djan, mi bleka le nirda Near John I watched the bird

mi bleka le nirda, vi la Djan

Mi godzi na I go now (here the "now" is not a tense but a sentence modifier).

The relationship of a sentence modifier to the sentence is exactly the same no matter where it appears in the sentence. It modifies the verb phrase, or equivalently, the entire situation represented by the sentence, not an argument it happens to be near.

Mi bleka le nirda vi la Djan means that I was near John when I was watching the bird.

Mi bleka le nirda ji vi la Djan means that I was watching the bird which was near John (a different grammatical construction, the subordinate clause, which we have not seen yet).

A modifier or modifiers may appear before the **ga** or tense in a subject-delayed sentence.

Na la Ven, pa kamla ga la Djan John will come at nine

Tagged arguments are arguments which are allowed to float free in a sentence in the same way that relative modifiers do. This can be done with numerical place tags or case tags.

The numerical place tags **zua**, **zue**, **zui**, **zuo**, **zuu** are signs of the first, second, third, fourth and fifth argument of a predicate. This allows argu-

[3]**guua** and some other right closers are new; we did a survey of grammatical constructions closable with **gu** and provided special forms for most of them.

ments to be freely reordered and moreover allows medial arguments to be omitted.

Zui la Djan, donsu zua la Meris Mary gave (something) to John.

It also allows arguments to be placed with the subject, as long as at most one argument in the subject is untagged:

La Meris, zui la Djan, pa donsu le bakso

It also allows more than one argument to be supplied for the same place.

Zua la Meris, zua la Djan, cluva la Ailin Mary and John love Eileen.

Untagged arguments are taken as usual to represent the places of the argument in order, skipping places corresponding to numerical place tags (or case tags) which have already appeared earlier in the sentence. We do not require a listener or reader to displace a sequence of arguments when a **zua** is encountered at the end of a sentence. Numerical place tags have a special effect in term lists appearing before the particle **gi**, which we will describe below.

The case tags are a bizarre idea which *we* would not have installed in this language. This is not to say that similar ideas do not occur in natural languages. With each place of a Loglan predicate, a *case* is associated in the dictionary, and that case tag may be used to reference that argument of that particular predicate in the same way the appropriate numerical case tag would reference it. Another possible use of a case tag is to suggest an argument whose place the speaker has forgotten, or perhaps an argument of the predicate which does not appear in the dictionary!

There is a further issue that the dictionary includes words in which distinct arguments have the same case. To support this, we have recently provided forms which reference the first, second, third, etc. argument of a given case.

The list of case tags will appear in the next section.

We give examples of use of these tags.

La Djan, dio la Meris, cluva John loves Mary. This is another way to get SOV order, and notice that **gio** is not needed.

Dio la Meris, (kao) la Djan, cluva, with the same meaning. The nominative case tag **kao** is optional: it can be used, which illustrates the fact that the subject needs to contain at least one argument and at most one untagged argument (two tagged arguments are all right).

La Djan, pa donsu dio la Meris, le bakso John gave the box to Mary. The **dio** indicates which argument **la Meris** is (she is not being given as a gift) and **le bakso** falls tidily into the first unused argument place.

Dio la Meris, beu le bakso, donsu la Djan means the same thing as the previous sentence.

3.1.6 Imperatives (and observatives)

An untensed sentence consisting of a verb followed by an object list, with possibly some modifiers before the verb, is an imperative.

Donsu le bakso la Djan! Give the box to John.

Na la Ven, donsu le bakso la Djan! At nine, give the box to John.

If a tensed verb followed by an object list (possibly preceded by modifiers) is given, this is actually a subject-delayed sentence with the subject omitted. We call this an *observative*: we note it as a special form mostly to indicate that such sentences are not imperatives.

Na crina It is raining (literally, someone is being rained on). This is a shortening of **Na crina ga ba**

Na donsu le bakso la Djan Someone is giving the box to John.

As an experiment, Loglan has borrowed a concept from Lojban and installed the imperative pronoun **koo**. This is used just like **tu** (you) with the extra force that the usual referent of **tu** is commanded to make the statement true.

Koo donsu le bakso la Djan! Give the box to John!

but also

La Meris, cluva koo! Make Mary love you!

Mi jupni lepo koo gudbi! Make me think well of you! (lit. Make me think you are good).

The imperatives with **koo** are sentences of perfectly general structure and do not belong to the imperative grammatical class which is the subject of this section, usually, though consider

Cluva koo! Love yourself!

3.2 Logically connected sentences

3.2.1 Forethought connected sentences

Chapter 4

Lexicography Appendix: full word lists

4.1 Case tags and indirect reference particles

The case tags, including the positional ones are listed:

beu: (patients/parts),

cau: (quantities/amounts/values),

dio: (destinations/receivers),

foa: (wholes/sets/collectives),

kao: (actors/agents/doers),

jui: (lessers),

neu: (conditions/circumstances/fields),

pou: (products/purposes),

goa: (greaters),

sau: (sources/reasons/causes),

veu: (effects/states/effects/deeds/means/routes),

zua: (first argument),

zue: (second argument),

zui: (third argument),

zuo: (fourth argument),

zuu: (fifth argument),

lae: (lae X = what is referred to by X),

lue: (lue X = something which refers to X)

The operators of indirect reference **lae** and **lue** are a different sort of creature, which originally had the same grammar as case tags, but now have somewhat different behavior. The latter two operators can be iterated (and so can case tags, probably indicating that more than one applies to the same argument).

For each semantic case tag there are forms like **beuzi, beucine** to reference the first argument with that tag, **beuza, beucito** to reference the second argument with that tag, and **beuzu, beucite** to reference the third argument with that tag. Forms like **beucifo, beucife** are theoretically possible.

Chapter 5

The Formal Grammar in PEG Notation

This chapter contains the actual Parsing Expression Grammar (PEG) notation in which the formal grammar is represented: this is the source from which the computer parser is constructed. It is also intended to be the basis of the presentation of the grammar.

I'm hoping to do some work on the PEG parser so that this file can be included here in a way which does not run off the margins.

```
# In this file I will develop the entire Loglan grammar on top of the phonetic propos

# Dated updates now to appear here

# I have further fine-tuning of djifoa gluing in mind.
# Allow the -r glue to be expressed as
# -rr after all mandatory monosyllables, removing the annoying pronunciation problem?
# I was thinking of allowing -hy gluing in other contexts, but it is actually a bad i

# 9/15/2019 installed semantic case tags with order distinctions for use with predica
# one solution is beucine, beucito...  another is beuzi, beuza, beuzu.


# 4/28/2019  Various debugging of the new predicate algorithm.  Added CVVhy as a glue
# added capitalization of djifoa glue!  Confirming my apparent earlier decision that
# by a full predicate complex.

# 4/26/2019:  this incorporates various revisions to the phonetics, correcting
# motivated by my development of the phonetics section of a new grammar docume
# change is that <ci> is now only a name marker if followed by an explicit pau
# changes in writing in serial names.  In speech, it is recommended that one n
# except before a name word.  The benefit is that non-serial-name related uses
# threaten mysterious needs to add explicit pauses before following name words

# I want to add the <zao> proposal of John Cowan.  Done, 4/15/2019.  the imper

#4/25  Making note of the idea that <ci> should not be a name marker unless fo
# by a pause.  This would require that one pause before ci-marked names and it
# remove some very confusing corrections for the false name marker problem.  I
# required the pause to be explicit we would be imposing the expectation that
# after <ci> is not a pause.  Otherwise we could encourage writing a juncture
# to deny presence of a pause, which is reasonable considering the meanings of
# I am implementing the version with explicit pauses between <ci> and names
# and the directive not to pause after <ci> without explicit indication.  This
# involves rewriting existing text only in the rare instances where <ci> prece

# 4/25/2019  Corrected some instances of (expanded) badstress.  Now forbidding

# 4/24/2019  Final consonants in syllables cannot be followed by syllabic cont
# this rationalizes the definition of SyllableA.

# 4/22 I am thinking of explicitly flagging imperative sentences;  not changin
# the grammar but making this visible in the parse.  This might also have some
# effects on logical connections.  4/23 created an imperative class for atomic
# imperative sentences;  this has no actual effect on parses, just
# organizes them in a more enlightening way.

# 4/17-18 2019:  updates commented out which make sentpred linkable with foret
# and afterthought connectives (making some uses of <guu> to share arguments
# unnecessary).  There are subtleties.  Basically, untensed predicates without
# argument lists will be linked by A and KA series connectives.  Such a linked
# set can be tensed as a whole.  Such a linked set will share a following term
# This will probably change many parses in the Visit and other legacy sources.
# This required some really subtle adjustments to work right, divinable from
# the actual rules given.  Definitely experimental.

# 3/9/2019 further, extended LIU1 to handle <ainoi> and its kin
# (actual mod is to class Cmapua)  Further, fixing mismatch
# between connective and A classes.  One does now have to pause
# before <ha> and its compounds.

# 3/9/2019 repaired bugs in negative attitudinals.  A pause
# in a negative attitudinal of the <no, ui> form will not break
# it.  <ainoi> didnt work for two reasons:  the clauses
# in the definition of NOUI were in the wrong order, and
# the connective class mistakenly included <noi> so the
# phonetics checker was crashing!  I had to move N and NOI
# earlier to make this work.  Not yet installed in the other
# version.

# 1/26/2019  added <vie>, JCB's "objective subjunctive" as a PA
# class word.  I should add this to the other file as well.

# 12/22/18:  just a comment:  one does not have to pause before <ha> and its compound
# I do not know whether to fix this.  One did not have to in LIP either.  For the mom
# leave it as it is.  As a matter of style, one probably should pause.

# 10/6/18  minor adjustments, made only in this file.  Allow <sujo> (a wicked thing t
# allow <futo>:  suffixed conversion operators must be nu + suffix.

# 6/2 fixed LIO + alien text.  I also fixed some other glitches described in the refe

# 5/11 making version without "alternative parser" features.  This version allows GAA
# do anything:  the definitions of argumentA and kin are the only point of difference
# becomes "alternative" by reinstating alternative definitions of argumentA and kin.
# recommended in the reference grammar.  ALTERNATIVE -- this is actually my master ve
# this and revise the argumentA and kin entries to make the original version.

# 4/24 discovered and repaired a bug re ci-marked names suffixed to descriptions.  Di
# descriptions yet to be fixed:  <lio> needs to be an alien text marker, maybe taking
# with-suffixed-name bug was actually quite gruesome.  I think it is repaired.
# 4/23 streamlined definition of descriptn.  Shouldn't change anything.  It wa
# in case of further trouble.

# 4/22 I think this will be the  master grammar file, with alternative lines t
# GAA-related features.

# 4/22 allowing general predicates in gasent1.  This removes an extreme oddity
# I do not see any new dangers from this.

# 4/22 I changed the final element of a keksent to be a sentence (new class ut
# several parse errors in the Visit were uncovered by this.

# 4/22:  note that I still have the obligation to restore the <zao> constructi

# 4/9/2018 the large subject marker GAA can also be used to defend the beginni
# from absorbing trailing arguments into an unintended statement.  In this con

# 4/8/2018 this is an alternative version in which an argument which starts an
# as a trailing argument of a previous sentence.  This allows neat termination
# a subject, for example.  Unlike the previous alternative approach, this seem
# tidy change:  it is all an issue of avoiding needs for explicit closure.  Fu
# can be marked with GAA (which is not a tense:  it appears optionally just be
# before sutori arguments marked with GIO if there are any), the "large subjec
# starts an SVO sentence *not marked with GAA* will not be accepted as a trail
# sentence.  This is a sufficiently complex grammar change that it requires th
# in my usual sense.  The fact that GAA carries a mandatory stress is virtuous
# particle GA when used as a tense is not a bad thing:  it would often be used
# a <lepo> clause appearing as a subject, and it is perhaps better for that pu
# and often will be followed by a tense.  This grammar change depends strongly
# SOV(O) sentences must be marked with <gio>:  S gio O^n V (O^m).

# nuu is an atomic A core and there is no nu-affix to A connectives and their

# 1/20/2018 redefined CA cores to include a possible NU prefix. This allows mo

# 1/13/2018 reorganized the internals of class PA in a way which should allow
# this is pursuant on an analysis of the classes NI and PA as phrases, rather
# proposal document.  Enforced explicit pauses after PA phrases appearing as argument

# 12/30/2017 fixed a problem with name markers in the clas NameWord and made a slight
# as dimensions).

# 12/27/2017 installing an alternative treatment of acronyms under which they are sim
# supporting this requires no change at all to acronymic name usage (just use the -n
# and for dimension usage requires <mue> to be a name marker and support for <mue> Pr

# 12/27/2017  Frivolously fooling with the capitalization conventions.  They ought to
# the main new idea was to require that a capitalized embedded letteral actually be f
# (with the obvious exception for a letteral followed by a letteral).  Also changed t
# legal for cmapua.  The general idea is that one can start with a capital letter and
# at which point one can jump back up to caps only at a juncture (after which you can
# after z- (after which lower case resumes) or an embedded literal (after which lower
# attested capitalization patterns in Loglan (including capitalization of embedded li
# and also allows all-caps for individual words (attested in Leith but suppressed in
# of names as in <la Beibi-Djein> (by artful use of syllable breaks:  Leith just has

# 12/26/2017  Installed <niu> (quotation of phonetically legal but so far non-Loglan
# use it with names (where it isn't really appropriate), one would have to pause init

# I note in this connection that quotation of names with li...lu remains limited, sin
# utterances:  one needs the <la>.  I fixed this as an exception in the previous pars
# not, haven't decided.  Single name words can be quoted with <liu>, of course, but n

# 12/24/2017  Refined treatment of vowel pairs for Cvv-V cmapua units.  First 12/24 v
# broken:  this should be fixed!

# 12/23/2017  This is now completely commented, with minor local exceptions to which
# This document is the basis on which I will build all subsequent parsers, with due m
# The Python PEG engine and preamble files contain commands for constructinging a Pyt

# 12/22/2017 major progress on commenting the grammar

# yet later 12/20:  no change in performance of the grammar, extensive commenting in
# grammar section.  Considerable changes in arrangement:  for example, vocatives, inv
# and free modifiers are moved to a much earlier point.  I'm hoping to get a genuinel
# commented grammar...

# later 12/20  starting the process of commenting and editing the grammar, sta
# at basic sentence structures.  Notably rewrote the class [keksent] more comp
# one hopes with no actual effect on parses.

# 12/20/2017  Do not require expression of pause after finally stressed cmapua
# vowel initial predicate as a comma, since the initial vowel signals the paus
# Allow final stress in names.  Fixed bug in CVVHiddenStress.  Prevented
# broken monosyllables in finally stressed CVV djifoa.  refinement of caprule

# 12/19/2017 seem to have had a versioning failure and lost the fix which requ
# CVVy djifoa to be followed by complete complexes.  Restored.

# 12/18/2017 fixed a bug in treatment of stressed syllables in recognizing pre
# narrowed the generalized VCCV rule to allow more of the quite unlikely space
# of vowels before the CC pair.  Probably they should be banned (and none have
# more than three) but that rule is not the context in which to arbitrarily ba
# of the display of parses, for which updated version of logicpreamble.py shou
# to class "connective" checking that apparent logical connectives are not ini
# This has the effect of delaying the declaration of "connective" until after
# "predstart".

# 12/17/2017 further refinement of the 12/16 version:  a couple of bugs spotte

# 12/16/2017 There should be no change in parsing behavior, but the predstart
# and more intelligible, and I realized that Complex doesnt need a check for t
# (the requirement that certain initial CVC cmapua be y hypenated which replac
# at all:  the way predstart works already ensures that initial CV cmapua fall
# cases, the idea being that we test the front of a predicate without lookahea
# addressed the subtle point that one wasn't forced to pause after a predicate
# (not likely to arise as a problem).


# 12/14/2017  Corrected vowel grouping to avoid paradoxical vowel triples whic
# grouped in a way which becomes illegal if made explicit.  SyllableA really s
# consonant:  the previous form was messing up vowel grouping.  Serious bug wh
# and syllable resolution of a predicate may fail to agree.  I think I blocked
# final djifoa are not followed by vowels.  Other fine tuning of the complex algorith
# to repair the check for CVCCCV and CVCCVV predicates.


# 12/13/2017:  added kie ( utterance ) kiu to class LiQuote.  Did fine tuning to ensu
# that cmapua streams stop before <li> or <kie>, that names can stop at double quotes
# parentheses, and that the capitalization rule ignores opening parentheses as well a
# quotes.  One can now adorn li lu with quotes (on the inside) in a reasonable way
# and adorn kie kiu with parentheses (on the inside) in a reasonable way.  One cannot
# *replace* these words (or any words) with punctuation in my model of Loglan.  Also,
# updates to comments, and # (end of utterance) added as a marker of terminal punctua

# END of dated updates

# This is now done, in a first pass.  That is, the grammar is adapted and appears to
# What is needed is comments on the lexicography and the grammar...Phonetics has now
# from the grammar (there are some places where the phonetics accept grammar informat

# Alien text is now handled somewhat differently.  Some issues to do with quoting nam

# I added -iy and -uy as VV forms allowed in general in cmapua but not in other words
# immediately allows me to do is to give Y a name which is not phonetically irregular

# capitalization is roughly back to where it was in the original, but all-caps are al

# acronyms are liable to be horrible.

# Fixed the recursion problem in a way which will not be visible in ordinary parses.
# be broken at name or alien text markers (instead of using lookahead to check that w
# of a name word  or alien text word).  The next cycle will then check for a name or
# badnamemarkers;  no lookahead is happening while a stream of cmapua is being read e
# the markers of names and alien text.  This will change the way phonetic parses look
# break (and sometimes resume) at name markers or alien text markers, but it will not
# parses.

#Part I Phonetics

# Mod bugs, I have implemented all of Loglan phonetics as described in my proposal.
# I have now parsed all the words in the dictionary, and all single words of a
# I have added alien text and quotation constructions which do not conform to
# all Loglan text should parse,  mod some punctuation and capitalization issue
# alien text here are not the same as those in the current provisional parser.

# I believe the conventions for forcing comma pauses before vowel initial cmap
# except in special contexts have been enforced.  In a full grammar, one proba
# to disable pauses before vowel initial letterals (done).  This grammar also
# irregularities in acronyms (and won't).

# This grammar (in Part I) is entirely about phonetics:  all it does is parse
# pauses or name markers), cmapua (qua unanalyzed streams of cmapua units),
# borrowings and complexes, along with interspersed comma pauses and marks
# of terminal punctuation.  It does support conventions about where commas are
# and a simple capitalization rule.  Streams of cmapua break when markers init
# in other forms are encountered (and may in some cases resume when the marker
# are a deception).

# a likely locus for odd bugs is the group of predstartX rules which detect ap
# are actually preambles to predicates.  These are tricky! (and I did indeed f
# problems when I parsed the dictionary).  Another reason to watch this rule p
# is that it carries a lot of weight:  !predstart is used as a lightweight tes
# that what follows is a cmapua (a point discussed in more detail later).

# In reviewing this, I think that very little is different from 1990's Loglan
# are post-1989 L1, but not my creation).  Some things add precision without m
# The requirement that syllabic consonants be doubled is new, and makes some 1
# The requirement that names resolve into syllables is new, and makes some 199
#  usually because they end in three consonants.
# The rule restricting final consonant pairs from being noncontinuant/continua
#   does not affect any actual predicate ever proposed.
# Enhancing the VccV rule to also forbid CVVV...ccV caused one predicate to be
#  (<haiukre> became <haiukrre>, and haiukre was a novelty anyway, using a new
# The exact definition of syllables and use of syllable breaks and stress mark
#  was replaced with the hyphen, so Lo,is becomes Lo-is); but this does not ma
#  incorrect, it merely increases precision and makes phonetic transcript poss
# Forbidding doubled vowels in borrowings was new, was already approved, and c
#  <alkooli> to <alkoholi>.
# Formally allowing the CVccVV and CVcccV predicates without y-hyphens took a proposa
#  Appendix H was careless in describing their abandonment of the slinkui test, but t
#  makes it evident that this was their intent all along.  The slinkui test had alrea
#  abandoned in the 1990s.
# Formally abandoning qwx was already something that the dictionary workers in the 19
#  on; we completed it.
# Allowing glottal stop in vowel pairs and forbidding it as an allophone of pause is
#  feature in the proposal but not reflected in the parser, of course.   Alternative
#  y and h and allowing h in final position are invisible or do not make any 1990's L
# Permitting false name markers in names was already afoot in the 1990's and the basi
#  approach were already in place.  The rule requiring explicit pauses between a name
#  a name word and the beginning of the next name word is new, but reflects something
#  a fact about 1990's Loglan pronunciation:  those pauses had to be made in speech
# (and in the 1990's they had no tools to do relevant computer tests)!  The requireme
#  that names resolve into syllables restricts which literal occurrences of name mark
#  false name markers (the tail they induce in the name must itself resolve into syll
# Working out the full details of borrowing djifoa was interesting:  I'm not sure tha
#  *new* there;  explicitly noting the stress shift in borrowing djifoa might be view
#  new but it is a logical consequence of JCB's permission to pause after a borrowing
#  explicit language about how it is to be stressed, and the
#  final definition of a borrowing djifoa as simply a borrowing followed by -y.  The
#  me as a really good idea anyway, because it marks djifoa with a pause after it as
#  in an additional way other than ending with the very indistinct vowel y.  My rules
#  directly enforce the rule that a borrowing djifoa must be preceded by y but I thin
#  enforce it in all or almost all cases:  the parser tries to read a borrowing djifo
#  any other kind of djifoa, so it is hard to see how to deploy a short djifoa in suc
#  fall off the head of a borrowing without using y.
# These phonetics do not support certain irregularities in acronyms.  We note that
# it is now allowed to insert <, mue> into an acronym, which would be necessary for e
# between a Ceo letteral and a following VCV letteral.

#Sounds

#all vowels

V1 <- [aeiouyAEIOUY]
```

```
#regular vowels

V2 <- [aeiouAEIOU]

#consonants

C1 <- [bcdfghjklmnprstvzBCDFGHJKLMNPRSTVZ]

# letters

letter <- (![qwxQWX] [a-zA-Z])

# a capitalization convention which allows what our current one allows and als
# if case goes down from upper case to lower case, it can only go back up in c
# does allow capitalization of initial segments of words.  There is a forward 
# in that free capitalization of embedded literals is permitted, and capitaliz
# guarded with z in literals as in DaiNaizA.

lowercase <- (![qwx] [a-z])

uppercase <- (![QWX] [A-Z])

caprule <- [\"(]? &([z] V1 (!uppercase/&TAIO)/lowercase TAIO (!uppercase/&TAIO

# syllable markers:  the hyphen is always medial so must be followed by a lett
# the stress marks can be syllable final and word final.  A juncture is never 
# by another juncture.

juncture <- (([-] &letter)/[\'*]) !juncture

stress <- ['*] !juncture

# terminal punctuation

terminal <- ([.:?!;#])

# characters which can occur in words
```

```
character <- (letter/juncture)

# to really get all Loglan text, we should add the alien text constructions and the m
# <lie>, <lao>, <sao>, <sue> and certain quotations which violate the phonetic rules.

# we adopt the convention that all alien text may be but does not have to be enclosed
# it needs to be understood that in quoted alien text, whitespace is understood as <,
# version this is shown explicitly.  This handling of alien text is taken from the fi
# of Linnaeans = foreign names, and extended by us to replace the impossible treatmen
# quotation in 1989 Loglan.

# this is a little different from what is allowed in the previous provisional parser,
# A difference is that all the alien text markers are allowed to be followed by the s

# the forms with <hoi> and <hue> are required to have following quotes in written for
# unintended parses, which otherwise become likely in case of typos in non-alien text

AlienText <- ([,]? [ ]+ [\"] (![\"].)+ [\"]/ [,]? [ ]+ (![, ]!terminal .)+ ([,]? [ ]+

AlienWord <- &caprule ([Hh] [Oo] [Ii] juncture? &([,]? [ ]+ [\"])/[Hh][Uu] juncture?

# while reading streams of cmapua, the parser will watch for the markers of alien tex

alienmarker <- ([Hh] [Oo] [Ii] juncture? &([,]? [ ]+ [\"])/[Hh][Uu] juncture? [Ee] ju

# 5/11/18 added <lio> as an alien text marker, to support numerals.

# the continuant consonants and the syllabic pairs they can form

continuant <- [mnlrMNLR]

syllabic <- (([mM] [mM] !(juncture? [mM]))/([nN] [nN] !(juncture? [nN]))/([rR] [rR] !

# the obligatory monosyllables, and these syllables when broken by a usually bad syll
# The i-final forms are not obligatory mono when followed by another i.

MustMono <- (([aeoAEO] [iI] ![iI]) /([aA] [oO]))
```

```
BrokenMono <- (([aeoAEO] juncture [iI] ![iI])/([aA] juncture [oO]))

# the obligatory and optional monosyllables.  Sequences of three of the same l
# are averted.  Avoid formation of doubled i or u after ui or ui.

Mono <- (MustMono/([iI] !([uU] [uU]) V2)/([uU] !([iI] [iI]) V2))

# vowel pairs of the form found in cmapua and djifoa.
# (other than the special IY, UY covered in the cmapua rules)

# The mysterious prohibition controls a permitted phonetic exception in djifoa
# compua are never followed directly by vocalic continuants in any case.

VV <- !(!MustMono V2 juncture? V2 juncture? [Rr] [Rr]) (!BrokenMono V2 junctur

# the next vocalic unit to be chosen from a stream of vowels
# in a predicate or name.  This is different than in our Sources
# and formally described in the proposal.

NextVowels <- (MustMono/(V2 &MustMono)/Mono/!([Ii] juncture [Ii] V1) !([Uu] ju

# 5/11/18 forbidding consonantal vowels to follow the same vowel.

# the doubled vowels that trigger the rule that one of them must be stressed

DoubleVowel <- (([aA] juncture? [aA])/([eE] juncture? [eE])/([oO] juncture? [o

# the mandatory "vowel" component of a syllable

Vocalic <- (NextVowels/syllabic/[Yy])

# the permissible initial pairs of consonants, and the same pairs possibly
# broken by syllable junctures.

Initial <- (([Bb] [Ll])/([Bb] [Rr])/([Cc] [Kk])/([Cc] [Ll])/([Cc] [Mm])/([Cc]

MaybeInitial <- (([Bb] juncture? [Ll])/([Bb]juncture?  [Rr])/([Cc]juncture?  [
# the permissible initial consonant groups in a syllable.  Adjacent consonants should
# The group should not overlap a syllabic pair.  Such a group is of course followed b

# this rule for initial consonant groups is stated in NB3.

# I forbid a three-consonant initial group to be followed by a syllabic pair.  This s

InitialConsonants <- ((!syllabic C1 &Vocalic)/(!(C1 syllabic) Initial &Vocalic)/(&Ini

# the forbidden medial pairs and triples.  These are forbidden regardless of placemen
# of syllable breaks.

# each of these is actually a single consonant followed by an initial, and the idea w
# would be hard to pronounce.  But the placement of the syllable break is not relevan
# Notice that the continuant syllabic pairs are excluded:  this prevents final conson

NoMedial2 <- (([Bb] juncture? [Bb])/([Cc] juncture? [Cc])/([Dd] juncture? [Dd])/([Ff]

NoMedial3 <- (([Cc] juncture? [Dd] juncture? [Zz])/([Cc] juncture? [Vv] juncture? [Ll

# The syllable.

# there are no formal rules about syllables as such in our Sources, which is odd sinc
# the definition of predicates depends on the placement of stresses on syllables.

# The first rule enforces the special point needed in complexes that
# a CVC syllable is preferred to a CV syllable where possible; we economically apply
# the same rule for default placement of syllable breaks everywhere, which is, with
# that exception, that the break comes as soon as possible.

# the SyllableB approach is taken if the following syllable would otherwise start wit

# the reason for this approach is that if one syllabizes a well formed complex in thi
# the syllable breaks magically fall on the djifoa boundaries.  This does mean that t
# default break in <cabro> is <cab-ro>, which feels funny but is harmless.  Explicitl
# it <ca-bro> will also parse correctly.

SyllableA <- (C1 V2 FinalConsonant (!Syllable FinalConsonant)?)
```

```
SyllableB <- (InitialConsonants? Vocalic (!Syllable FinalConsonant)? (!Syllabl

Syllable <- ((SyllableA/SyllableB) juncture?)

# The final consonant in a syllable.  There may be one or two final consonants
# consonants may not be a non-continuant followed by a continuant.  A final co
# start a forbidden medial pair or triple.

# The rule that a final consonant pair may not be a non-continuant followed by
# is natural and obvious but not in our Sources.  Such a pair of consonants wo
# naturally form another syllable.

FinalConsonant <- !syllabic (!(!continuant C1 !Syllable continuant) !NoMedial2

# Here are various flavors of syllable we may need.

# this is a portmanteau definition of a bad syllable (the sort not allowed in

SyllableD <- &(InitialConsonants? ([Yy]/DoubleVowel/BrokenMono/&Mono V2 Double

# this (below) is the kind of syllable which can exist in a borrowed predicate
# it cannot start with a continuant pair, it cannot have a y as vocalic unit,
# and its vocalic unit (whether it has one or two regular vowels)
# cannot be involved in a double vowel or an explicitly broken
# mandatory monosyllable.

BorrowingSyllable <- !syllabic (!SyllableD) Syllable

# this is the final syllable of a predicate.  It cannot be followed
# without pause by a regular vowel.

VowelFinal <- InitialConsonants? Vocalic juncture? !V2

# syllables with syllabic consonant vocalic units
# this class is only used in borrowings, and we *could* reasonably
# require it to be followed by a vowel.  But I won't for now.
# for gluing this restriction would work, but we might literally borrow predic
# with syllabic continuant pronunciations.

SyllableC <- (&(InitialConsonants? syllabic) Syllable)

# syllables with y

SyllableY <- (&(InitialConsonants? [Yy]) Syllable)

# an explicitly stressed syllable.

StressedSyllable <- ((SyllableA/SyllableB) [\'*])

# a final syllable in a word, ending in a consonant.

NameEndSyllable <- (InitialConsonants? (syllabic/Vocalic &FinalConsonant) FinalConson

# the pause classes actually hang on the letter before the pause.

# whitespace which might or might not be a pause.

maybepause <- (V1 [\'*]? [ ]+ C1)

# explicit pauses:  these are whitespace before a vowel or after a consonant, or comm

pause <- ((C1 [\'*]? [ ]+ &letter)/(letter [\'*]? [ ]+ &V1)/(letter [\'*]? [,] [ ]+ &

# these are final syllables in words followed by whitespace which might not be a paus
# the definition actually doesnt mention the maybepause class.

MaybePauseSyllable <- InitialConsonants? Vocalic ['*]? &([ ]+ &C1)

# The full analysis of names.

# a name word (without initial marking) is resolvable into syllables and ends with a

PreName <- ((Syllable &Syllable)* NameEndSyllable)

# this is a busted name word with whitespace in it -- but not whitespace at which one
```

```
BadPreName <- (MaybePauseSyllable [ ]+/Syllable &Syllable)* NameEndSyllable

# This is a name marker followed by a consonant initial name word without paus

# I deployed a minimal set of name marker words; I can add the others whenever
# I have decided (see below) to retain the social lubrication words as vocativ
# *without* making them name markers, so one must pause <Loi, Djan>.  By not a
# freemods right after vocative markers in the vocative rule, I make <Loi hoi
# without pause.

# MarkedName <- &caprule ((([Ll] !pause [Aa] juncture?)/ ([Hh] [Oo] !pause [Ii

MarkedName <- &caprule ((([Ll] !pause [Aa] juncture?)/ ([Hh] [Oo] !pause [Ii]


# This is an unmarked name word with a false name marker in it.

FalseMarked <- (&PreName (!MarkedName character)* MarkedName)

# This is the full definition of name words.  These are either marked consonan
# names without false name markers beginning with explicit pauses (either comm
# and name markers followed, with or without pause, by name words.  In the lat
# whitespace before a vowel initial name.

# a series of names without false name markers and names marked with ci, separ

# there is a look ahead at the grammar: a NameWord can be followed without exp
# a pause in speech!) by another
# kind of utterance only in a serial name when what follows is of the form <ci
# in the name.

NameWord <- (&caprule MarkedName/([,] [ ]+ !FalseMarked &caprule PreName)/(&V1

# this is the minimal set of name marker words we are using.  We may add more.

# I am contemplating adding the words of social lubrication as name markers, b
# way that in the last provisional parser, in which I made them full-fledged v
# I preserved their status as vocative markers without restoring their status as name

# adding <mue> as a name marker

namemarker <- ([Ll] [Aa] juncture?/[Hh][Oo][Ii] juncture?/([Hh] [Uu] juncture? [Ee] j

# this is the bad name marker phenomenon that needs to be excluded.  This captures th
# that what follows the name could be pronounced without pause as a name word accordi
# orthography, but the fact that whitespace is present shows that this is not the int

# it is worth noting that name markers at heads of name words pass this test
# (because I omitted the test that what follows is not a PreName in the interests
# of minimizing lookahead);
# but this test is only applied to strings that have already been determined not to
# be of class NameWord.

badnamemarker <- namemarker !V1 [, ]? [ ]* BadPreName

# we test for the bad name marker condition at the beginning of each stream of cmapua
# and streams of cmapua stop before name markers (and may resume at a name marker
# if neither a NameWord nor the bad marker condition is found).

# We have at any rate completely solved the phonetic problem of names and their marke

# predicate start tests:  the idea is the same as class "connective" above, to recogn
# the start of a predicate without recursive appeals to the whole nasty definition of
# The reason to do it is to recognize when CV^n followed by CC cannot be a cmapua uni

# New implementation 4/28/2019.  This allows only (C)V(V)(V) before the pair of vowel
# potential lookahead.

Vthree <- (V2 juncture?) (V2 juncture?) (V2 juncture?)

Vfour <- (V2 juncture?) (V2 juncture?) (V2 juncture?) (V2 juncture?)

# predicate starting with two or three consonants:  rules out CC(C)V(V) forms.  Junct
# the initial consonant group ignored.
```

```
predstartA1 <- (&MaybeInitial C1 juncture? MaybeInitial/MaybeInitial) &V2 !(V2

# an apparent cmapua unit followed by a consonant group which cannot start a p

predstartA2 <- C1 V2 juncture? (V2 juncture?)? !predstartA1 C1 juncture? C1

# a stressed CV^n before a consonant group (CV(V) case)

predstartA3 <- C1 !Vthree (!StressedSyllable V2 juncture?)? &StressedSyllable

# other (C)V^n followed by nonpredicate

predstartA4 <- C1? V2 juncture? (V2 juncture?)?  (V2 juncture?)? !predstartA1

# other stressed (C)V^n followed by consonant group

predstartA5 <- C1? !Vfour (!StressedSyllable V2 juncture?)? (!StressedSyllable

# forms with y; implemented CVVhy alternative for CVV cmapua

predstartA6 <- C1 (V2 juncture?) (V2 juncture? [Hh]?/C1 juncture? (C1 juncture

predstart <- predstartA1/predstartA2/predstartA3/predstartA4/predstartA5/preds

# it is worth noting that in the sequel we have systematically replaced tests
# with !predstart.  The former involves lots of lookahead and was causing recu
# in Python.  The phonetics and the grammar are both structured so that any st
# starting with a name marker is tested for NameWord-hood before it is tested
# cmapua-hood; the only thing it is tested for later is predicate-hood, and pr
# is a rough and ready test that something might be a predicate (and at any ra
# cannot be a cmapua).

# this class requires pauses before it, after all the phonetic word classes.
# what is being recognized is the beginning of a logical connective.

# To avoid horrible recursion problems, giving this a concrete phonetic defini
# without much lookahead.  This can go right up in the phonetics section if it
# (and here it is!).
# single vowel cmapua syllables early for connectives

a <- ([Aa] !badstress juncture? !V1)

e <- ([Ee] !badstress juncture? !V1)

i <- ([Ii] !badstress juncture? !V1)

o <- ([Oo] !badstress juncture? !V1)

u <- ([Uu] !badstress juncture? !V1)


Hearly <- (!predstart [Hh])

Nearly <- (!predstart [Nn])


# these appear here for historical reasons and could be moved later


connective <- [ ]* !predstart ([Nn] [Oo] juncture?)? (a/e/o/u/Hearly a/Nearly UU) jun


# cmapua units starting with consonants.  This is the exact description from NB3.  Th
# three cases is enforcing the rule about pausing before a following predicate if str

# consonant initial cmapua units may not be followed by vowels without pause.

# I am adding <iy> and <uy> (always monosyllable, yuh and wuh) as vowel pairs permitt
# it is worth noting that the "yuh" and "wuh" pronunciations of these diphthongs
# are surprising to the English-reading eye.
# The use for this envisaged is that the name <ziy> of Y becomes easy to introduce.
# is always nice, and these words seem pronounceable.  I also made <yfi> possible:  Y
# regular names.

CmapuaUnit <- (C1 Mono juncture? V2 !(['*] [ ]* &C1 predstart) juncture? !V1/C1 (VV/[
# A stream of cmapua is read until the start of a predicate or a name marker w
# the stream might resume with a name marker word if it does not in fact start
# word due to inexplicit whitespace (doesn't satisfy the bad name marker condi

# we force explicit comma pauses before logical connectives, but not before vo
# other conditions force at least whitespace, which does stand for a pause, be

# detect starts of quotes or parentheses with <li> or <kie>

likie <- ([Ll] [Ii] juncture? !V1/[Ki] [Ii] juncture? [Ee] juncture? !V1)

# a special provision is made for NO UI forms as single words.  <yfi> is suppo

Cmapua <- &caprule !badnamemarker (!predstart (VV/[Ii][Yy]/[Uu][Yy]) !(['*] [

# I have apparently now completely solved the problem of parsing cmapua as wel

# Now for predicates.

# the elementary djifoa (not borrowings)

# various special flavors of these djifoa will be needed.
# These are the general definitions.

# The NOY and Bad forms are for use for testing candidate borrowings for resol
# with bad syllable break placements.  Borrowings do not contain Y...

# CVV djifoa with phonetic hyphens.

# added checks to all cmapua classes:  the vowel final ones, when not phonetic
# be followed by a regular vowel.  This is crucial for getting the syllable an
# analysis to end at the same point.

# allowing h to be inserted before y in CVVy djifoa for a CVVhy form.

# allowing -r glue to be expressed as -rr
```

```
CVV <- C1 VV (juncture? [Hh]? [Yy] [-]? &(Complex) /juncture? [Rr] [Rr]? juncture? &C

CVVNoHyphen <- C1 VV juncture? !V2

CVVHiddenStress <- C1 &DoubleVowel V1 [-]? V1 ([-]? [Hh]? [Yy] [-]? &Complex /[Rr] [-

CVVFinalStress <- C1 VV (['*] [Hh]? [Yy] [-]? &Complex /[Rr] ['*] &C1/['*] [Rr] [Rr]

CVVNOY <- C1 VV (juncture? [Rr] [Rr]? juncture? &C1/[Nn] juncture? &[Rr]/juncture? !V

CVVNOYFinalStress <- C1 VV ([Rr] ['*] &C1/['*] [Rr] [Rr] juncture? &C1/[Nn] ['*] &[Rr

CVVNOYMedialStress <- C1 !BrokenMono V2 ['*] V2 [-]? !V2

# CCV djifoa with phonetic hyphens.

CCV <- Initial V2 (juncture? [Yy] [-]? &letter/juncture? !V2)

CCVStressed <- Initial V2 (['*] [Yy] [-]? &letter/['*] !V2)

CCVNOY <- Initial V2 juncture? !V2

CCVBad <- MaybeInitial V2 juncture? !V2

CCVBadStressed <- MaybeInitial V2 ['*] !V2


# CVC djifoa with phonetic hyphens.  These cannot be final and are always followed by
# -y form may be followed by a vowel...
# an eccentric syllable break is supported if the CVC is y-hyphenated:
# <me-ky-kiu> and <mek-y-kiu> are both legal.  The default is the latter.

CVC <- (C1 V2 !NoMedial2 !NoMedial3 C1 (juncture? [Yy] [-]? &letter/juncture? &C1)/C1

CVCStressed <- (C1 V2 !NoMedial2 !NoMedial3 C1 (['*] [Yy] [-]? &letter/['*] &letter)/

CVCNOY <- C1 V2 !NoMedial2 !NoMedial3 C1 juncture? &C1
```

```
CVCBad <- C1 V2 !NoMedial2 !NoMedial3 juncture? C1 &C1

CVCNOYStressed <- C1 V2 !NoMedial2 !NoMedial3 C1 ['*] &C1

CVCBadStressed <- C1 V2 !NoMedial2 !NoMedial3 ['*] C1 &C1

# the five letter forms (always final in complexes)

CCVCV <- Initial V2 juncture? C1 V2 [-]? !V2

CCVCVStressed <- Initial V2 ['*] C1 V2 [-]? !V2

CCVCVBad <- MaybeInitial V2 juncture? C1 V2 [-]? !V2

CCVCVBadStressed <- MaybeInitial V2 ['*] C1 V2 [-]? !V2

CVCCV <- (C1 V2 juncture? Initial V2 [-]? !V2/C1 V2 !NoMedial2 C1 juncture? C1

CVCCVStressed <- (C1 V2 ['*] Initial V2 [-]? !V2/C1 V2 !NoMedial2 C1 ['*] C1 V

# the medial five letter djifoa

CCVCY <- Initial V2 juncture? C1 [Yy] [-]?

CVCCY <- (C1 V2 juncture? Initial [Yy] [-]?/C1 V2 !NoMedial2 C1 juncture? C1 [

CCVCYStressed <- Initial V2 ['*] C1 [Yy] [-]?

CVCCYStressed <- (C1 V2 ['*] Initial [Yy] [-]?/C1 V2 !NoMedial2 C1 ['*] C1 [Yy

# to reason about resolution of borrowings into both syllables and djifoa (we
# but we need to define it adequately) we need to recognize where to stop.  A
# at a non-character (not a letter or syllable mark: whitespace, comma or term
# has an explicit or deducible penultimate stress.  Borrowings do not contain
# have to have explicit stress in the latter case.

# analysis:  the stressed tail consists of a stressed syllable followed by an
# identifying an unstressed final syllable is complicated by recognizing which
# be one syllable.  This will either be an explicitly stressed syllable followed by a
# or a syllable suitable to be stressed followed by an explicitly final syllable.  CV
# contain both syllables in a tail and of course the five letter djifoa have to be ta
# SyllableC (with a continuant) may intervene.

# tail of a borrowing with an explicit stress

BorrowingTail1 <- !SyllableC &StressedSyllable BorrowingSyllable (!StressedSyllable &

# tail of a borrowing or borrowing djifoa with no explicit stress

BorrowingTail2 <- !SyllableC BorrowingSyllable (!StressedSyllable &SyllableC Borrowin

# tail of a stressed borrowing djifoa, different because stress is shifted to the end

BorrowingTail3 <- !SyllableC !StressedSyllable BorrowingSyllable (!StressedSyllable &

BorrowingTail <- BorrowingTail1 / BorrowingTail2

# short forms that are ruled out:  CCVV and CCCVV forms.

CCVV <- (InitialConsonants V2 juncture? V2 juncture? !character / InitialConsonants V

# VCCV and some related forms are ruled out (rule predstartF above is about this)

# a continuant syllable cannot be initial in a borrowing and there cannot be successi
# syllables.  There really ought to be no more than one!

# borrowing, before checking that it doesnt resolve into djifoa

PreBorrowing <- &predstart!CCVV!Cmapua!SyllableC(!BorrowingTail!(StressedSyllable)!(S

# ditto for an explicitly stressed borrowing

StressedPreBorrowing <- &predstart!CCVV!Cmapua!SyllableC(!BorrowingTail!(StressedSyll

# borrowing djifoa without explicit stress (before resolution check)
```

```
PreBorrowing2 <- &predstart!CCVV!Cmapua!SyllableC(!BorrowingTail!(StressedSyll

# stressed borrowing djifoa (before resolution check).

PreBorrowing3 <- &predstart!CCVV!Cmapua!SyllableC(!BorrowingTail3!(StressedSyl

# Now comes the problem of trying to say that a preborrowing cannot resolve in
# recognizing the tail, so making sure that the two resolutions stop in the sa

# we know because it is a borrowing that there is at most one explicit stress,
# in one of the cmapua!  This should make it doable.

# borrowing djifoa are terminated with y, so the final djifoa needs to take th

# the idea behind both djifoa analyses is the same.  If we end with a final dj
# a non-character, we improve our chances of ending the syllable analysis at t
# this by identifying djifoa with stresses in them:  a medially stressed djifo
# (and the syllable analysis will find its stressed syllable and end at its fi
# that djifoa cannot be followed by vowels ensuring that the syllable analysis
# When the djifoa is finally stressed, the complex analysis ends with a furthe
# just one syllable, and the syllable analysis again will stop in the same pla
# and borrowing djifoa of course are finally stressed mod an additional unstre
# by the syllable analysis, because it allows one to ignore an actually penult
# a syllabic consonant.  In the case where we never find a stress and end up a
# analysis will carry right through to the same final point.

# in the attempted resolution of borrowings, our life is easier because we do
# borrowing djifoa or medial five letter forms to consider, or any forms with

RFinalDjifoa <- (CCVCVBad/CVCCV/CVVNoHyphen/CCVBad/CVCBad) (&[Yy]/!character)

RMediallyStressed <- (CCVCVBadStressed/CVCCVStressed/CVVNOYMedialStress)

RFinallyStressed <- (CVVNOYFinalStress/CCVBadStressed/CVCBadStressed/CVCNOYStr

BorrowingComplexTail <- (RMediallyStressed/RFinallyStressed (&(C1 Mono) CVVNoH

ResolvedBorrowing <- (!BorrowingComplexTail(CVVNOY/CCVBad/CVCBad))* BorrowingC
```

```
# borrowed predicates

Borrowing <- !ResolvedBorrowing &caprule PreBorrowing !([ ]* (connective))

# explicitly stressed borrowed predicates

StressedBorrowing <- !ResolvedBorrowing &caprule StressedPreBorrowing !([ ]* &V1 Cmap

#This is the shape of non-final borrowing djifoa.  Notice that a final stress is allo
#The curious provision for explicitly stressing a borrowing djifoa and pausing is sup

# borrowing djifoa without explicit stress (stressed ones are not of this class!)
# Note that one can pause after these (explicitly, with a comma)

BorrowingDjifoa <- !ResolvedBorrowing &caprule PreBorrowing2 (['*] y [,] [ ]+/junctur

# stressed borrowing djifoa finally implemented!

StressedBorrowingDjifoa <- !ResolvedBorrowing &caprule PreBorrowing3 [y] [-]? ([,] [ 

# We resolve complexes twice, once into syllables and once into djifoa.  We again hav
# we end up in the same place!  The syllable resolution is very similar to that of bo
# the unstressed middle syllable of the tail can be a SyllableY, and can also be a
# SyllableC if the final djifoa is a borrowing.

# A stressed borrowing djifoa with the property that the tail is still a phonetic com
# a unit for this analysis.

# note here that I specifically rule out a complex being followed without pause by y.
# this out for the vowel final djifoa because they can be followed by y at the end of
# djifoa.

PhoneticComplexTail1 <- !SyllableC !SyllableY &StressedSyllable Syllable (!StressedSy

PhoneticComplexTail2 <- !SyllableC !SyllableY Syllable (!StressedSyllable &(SyllableC

PhoneticComplexTail <- PhoneticComplexTail1 / PhoneticComplexTail2
```

```
# note the explicit predstart test here.

PhoneticComplex <- &predstart!CCVV!Cmapua!SyllableC(StressedBorrowingDjifoa &P

# the analysis of final djifoa and stressed djifoa differs only in details fro
# what is above for resolution of borrowings.  The issues about CVV djifoa wit
# vowels are rather exciting.

# a stressed borrowing djifoa with the tail still a phonetic complex is a blac
# this construction.

# My approach imposes the restriction on JCB's "pause after a borrowing djifoa
# the pause must itself contain a penultimate stress:  <iglluʼymao> is a predi
# while <iglluy', gudmao> is a predicate.

# the analysis of the djifoa resolution process is the same as above, with add
# about doubled vowel syllables:  notice that where the complex tail involved
# without explicit stress, we insist on that djifoa or the single-syllable nex
# a non-character:  in the absence of explicit stress, we always rely on white
# to indicate the end of the predicate.

# all sorts of subtleties about borrowings and borrowing djifoa are finessed b
# them first.  There are no restrictions re fronts of borrowings or borrowing
# djifoa;  the fact that borrowing djifoa end in y and borrowings do not conta
# possible to tell when one is looking at the head of a borrowing djifoa.  Reg
# djifoa need to be y-hyphenated so as not to be absorbed into the front of th
# that I actually need to impose a formal rule to this effect, though I am not
# be difficult to formulate [and does appear in the previous version, where it
# of PEG code]).

FinalDjifoa <- (Borrowing/CCVCV/CVCCV/CVVNoHyphen/CCVNOY) !character

MediallyStressed <- (StressedBorrowing/CCVCVStressed/CVCCVStressed/CVVNOYMedia

FinallyStressed <-(StressedBorrowingDjifoa/CCVCYStressed/CVCCYStressed/CVVFina

ComplexTail <- (CVVHiddenStress (&(C1 Mono) CVVNoHyphen/CCVNOY) !character/Fin
PreComplex <-  (!CVVHiddenStress (!ComplexTail)(StressedBorrowingDjifoa &PhoneticComp

# originally I had complicated tests here for the conditions under which an initial
# CVC cmapua has to be y-hyphenated:  I was being wrong headed, the predstart rules
# already enforce this (in the bad cases, the initial CV- falls off).  The user will
# simply find that they cannot put the word together otherwise.  The previous version
# did need this test because it actually used full lookahead to check for the start o

Complex <- &caprule &PreComplex PhoneticComplex !([ ]* (connective))

# format for the LI quote and KIE parenthesis

LiQuote <- (&caprule [Ll][Ii]juncture? comma2? [\"] phoneticutterance [\"] comma2? &c

# the condition on Word that a Cmapua is not followed by another Cmapua
# with mere whitespace between was used by <liu> quotation, but is now redundant,
# because I have required that <liu> quotations be closed with explicit pauses in all

Word <- (NameWord / Cmapua !([ ]*Cmapua)/ Complex/CCVNOY)

# it is an odd point that all borrowings parse as complexes -- so when I parsed all t
# parsed as complexes.  A borrowing is a complex consisting of a single final borrowi
# I did redesign this so that borrowings are parsed as borrowings.  (This is the clas
# I used to parse the dictionary).

# Yes, CVC djifoa do get parsed as names in the dictionary, so the CVC case here is r
# think that only the CCV djifoa actually get parsed as such.

SingleWord <- (Borrowing !./Complex !./ Word !./PreName !. /CCVNOY) !.

# name word appearing initially without leading spaces is important, because one type

phoneticutterance1 <- (NameWord /[ ]* LiQuote/[ ]* NameWord/[ ]* AlienWord/[ ]*Cmapua

phoneticutterance <- (phoneticutterance1/[,][ ]+/terminal)+

# consonants and vowel groups in cmapua
# as noted above, !predstart stands in for the computationally disastrous &Cma

badstress <- ['*] [ ]* &C1 predstart

B <- (!predstart [Bb])

C <- (!predstart [Cc])

D <- (!predstart [Dd])

F <- (!predstart [Ff])

G <- (!predstart [Gg])

H <- (!predstart [Hh])

J <- (!predstart [Jj])

K <- (!predstart [Kk])

L <- (!predstart [Ll])

M <- (!predstart [Mm])

N <- (!predstart [Nn])

P <- (!predstart [Pp])

R <- (!predstart [Rr])

S <- (!predstart [Ss])

T <- (!predstart [Tt])

V <- (!predstart [Vv])

Z <- (!predstart [Zz])
```

```
# the monosyllabic classes may be followed by one vowel
# if they start a Cvv-V cmapua unit;  the others may never
# be followed by vowels.  Classes ending in -b are
# used in Cvv-V cmapua units.

# the single vowel classes were moved before the class
# connective in the phonetics section.


V3 <- juncture? V2 !badstress

AA <- ([Aa] juncture? [Aa] !badstress juncture? !V1)

AE <- ([Aa] juncture? [Ee]  !badstress juncture? !V1)

AI <- ([Aa] [Ii]  !badstress juncture? !(V1))

AO <- ([Aa] [Oo]  !badstress juncture? !(V1))

AIb <- ([Aa] [Ii]  !badstress juncture? &(V2 juncture? !V1))

AOb <- ([Aa] [Oo]  !badstress juncture? &(V2 juncture? !V1))

AU <- ([Aa] juncture? [Uu]  !badstress juncture? !V1)

EA <- ([Ee] juncture? [Aa]  !badstress juncture? !V1)

EE <- ([Ee] juncture? [Ee]  !badstress juncture? !V1)

EI <- ([Ee] [Ii]  !badstress juncture? !(V1))

EIb <- ([Ee] [Ii]  !badstress juncture? &(V2 juncture? !V1))

EO <- ([Ee] juncture? [Oo]  !badstress juncture? !V1)

EU <- ([Ee] juncture? [Uu]  !badstress juncture? !V1)
```

```
IA <- ([Ii] juncture? [Aa]   !badstress juncture? !(V1))

IE <- ([Ii] juncture? [Ee]  !badstress juncture? !(V1))

II <- ([Ii] juncture? [Ii]  !badstress juncture? !(V1))

IO <- ([Ii] juncture? [Oo]  !badstress juncture? !(V1))

IU <- ([Ii] juncture? [Uu]   !badstress juncture? !(V1))

IAb <- ([Ii] juncture? [Aa]   !badstress juncture? &(V2 juncture? !V1))

IEb <- ([Ii] juncture? [Ee]  !badstress juncture? &(V2 juncture? !V1))

IIb <- ([Ii] juncture? [Ii]  !badstress juncture? &(V2 juncture? !V1))

IOb <- ([Ii] juncture? [Oo]  !badstress juncture? &(V2 juncture? !V1))

IUb <- ([Ii] juncture? [Uu]   !badstress juncture? &(V2 juncture? !V1))

OA <- ([Oo] juncture? [Aa]  !badstress juncture? !V1)

OE <- ([Oo] juncture? [Ee]  !badstress juncture? !V1)

OI <- ([Oo] [Ii]  !badstress juncture? !(V1))

OIb <- ([Oo] [Ii]  !badstress juncture? &(V2 juncture? !V1))

OO <- ([Oo] juncture? [Oo]  !badstress juncture? !V1)

OU <- ([Oo] juncture? [Uu]   !badstress juncture? !V1)

UA <- ([Uu] juncture? [Aa]   !badstress juncture? !(V1))

UE <- ([Uu] juncture? [Ee]  !badstress juncture? !(V1))

UI <- ([Uu] juncture? [Ii]   !badstress juncture? !(V1))
```

```
UO <- ([Uu] juncture? [Oo]  !badstress juncture? !(V1))

UU <- ([Uu] juncture? [Uu]  !badstress juncture? !(V1))

UAb <- ([Uu] juncture? [Aa]   !badstress juncture? &(V2 juncture? !V1))

UEb <- ([Uu] juncture? [Ee]  !badstress juncture? &(V2 juncture? !V1))

UIb <- ([Uu] juncture? [Ii]   !badstress juncture? &(V2 juncture? !V1))

UOb <- ([Uu] juncture? [Oo]  !badstress juncture? &(V2 juncture? !V1))

UUb <- ([Uu] juncture? [Uu]  !badstress juncture? &(V2 juncture? !V1))

# adding the new IY and UY, which might see use some time.
# they are mandatory monosyllables but do not take a possible additional
# following vowel as the regular ones do.  So far only used in <ziy>.

IY <- [Ii] [Yy] !badstress juncture? !V1

UY <- [Uu] [Yy] !badstress juncture? !V1

# this is a pause not required by the phonetics.  This is the only
# sort of pause which could in principle carry semantic freight (the
# pause/GU equivalence beloved of our Founder) but we have abandoned
# this.  There is one place, after initial <no> in an utterance, where
# a pause can have effect on the parse (but not on the meaning, I believe,
# unless a word break is involved).

# this class should NEVER be used in a context which might follow
# a name word.  In previous versions, pauses after name words were included
# in the name word;  this is not the case here, so a PAUSE
# after a name word would not be recognized as a mandatory pause.

# in any event, as long as we stay away from pause/GU equivalence, this
# is not a serious issue!

# this class does do some work in the handling of issues surrounding the legacy
# shape of APA connectives, concerning which the less said, the better.

PAUSE <- [,] [ ]+ !(V1/connective) &caprule

# more punctuation

comma <- [,] [ ]+ &caprule

comma2 <- [,]? [ ]+ &caprule

# Part II Lexicography

# In this section I develop the grammar of words in Loglan.  I'll work by edit

# I place the start of this section exactly here, just before two final items
# punctuation, because these items of punctuation look forward not only to lex
# but to the full grammar!

# the end of utterance symbol <#> should be added in the phonetics
# section as a species of terminal marker. Done.  We do *not* actually
# endorse use of this marker, but we can notionally support it and it is in
# our sources.

end <- (([ ]* '#' [ ]+ utterance)/([ ]+ !.)/!.)

# this rule allows terminal punctuation to be followed by an inverse vocative,
# a frequent occurrence in Leith's novel, and something which makes sense.

period <- (([!.:;?] (&end/([ ]+ &caprule))) (invvoc period?)?)

# Letters with y will be special cases
# idea:  allow IY and UY (always monosyllables) as vowel combinations in cmapu
# done:  Y has a name now.  <yfi> is also added.

# the classes in this section after this point are the cmapua word classes of
# I suppose the alien text classes are not really word classes, but they are l
# Paradoxically, the PA and NI classes admit internal explicit pauses.  So of
# Loglan does admit true multisyllable cmapua:  there are words made of cmapua units 
# units at which one cannot pause without breaking the word.  Lojban, I am told, does

# this version has the general feature that the quotation and alien text construction
# they are supported by the phonetic rules (as dire exceptions, of course) and the gr
# conform with the phonetic layer.  Alien text and utterances quoted with <li>...<lu>
# LI only supports full utterances, for the moment.  All alien text constructors take
# the vocative and inverse vocative *require* quotes to avoid misreading ungrammatica
# as correct (inverse) vocatives.

# the names <yfi>, <ziy> for Y are supported.  The Ceo names are left as they are.  I
# of letteral pronouns is actually a reasonable use of short words, and the Ceio word

TAI0 <- (V1 juncture? M a/V1 juncture? F i/V1 juncture? Z i/!predstart C1 AI/!predsta

# a negative suffix used in various contexts.  Always a suffix:  its use as a prefix 
# think still supported in LIP.  Ambiguities demonstrably followed from this usage (a
# of non-ambiguity of 1989 Loglan was compromised by the opaque lexicography).

NOI <- (N OI)

# the logical connectives.  [A0] is the class of core logical connectives.  [A] is th
# possible nu- (always in nuno- or nuu) and no- prefixes, possible -noi suffix, and p
# with -fi (our new proposal) or an explicit pause.

A0 <- &Cmapua (a/e/o/u/H a/N UU)

A <- [ ]* !predstart !TAI0 (N [o])? A0 NOI? !([ ]+ PANOPAUSES PAUSE) !(PANOPAUSES !PA

# 4/18 in connected sentpreds, fi must be used to close, not a pause.

# A2 <- [ ]* !predstart !TAI0 (N [o])? A0 NOI? !([ ]+ PANOPAUSES PAUSE) !(PANOPAUSES 



# A not closed with -fi or a pause

ANOFI <- [ ]* (!predstart !TAI0 ( (N [o])? A0 NOI? PANOPAUSES?))
```

```
A1 <- A

# versions of A with different binding strength

ACI <- (ANOFI C i)

AGE <- (ANOFI G e)

# a tightly binding series of logical connectives used to link predicates
# this also includes the fusion connective <ze> when used between predicates.

CA0 <- (( (N o)? ((C a)/(C e)/(C o)/(C u)/(Z e)/(C i H a)/N u C u)) NOI?)

CA1 <- (CA0 !([ ]+ PANOPAUSES PAUSE) !(PANOPAUSES !PAUSE [ ,]) (PANOPAUSES ((F

CA1NOFI <- (CA0 PANOPAUSES?)

CA <- ([ ]* CA1)

# the fusion connective when used in arguments

ZE2 <- ([ ]* (Z e))

# sentence connectives.  [I] is the class of utterance initiators (no logical
# the subsequent classes are inhabited by sentence logical connectives with va
# strengths.

I <- ([ ]* !predstart !TAI0 i !([ ]+ PANOPAUSES PAUSE) !(PANOPAUSES !PAUSE [ ,

ICA <- ([ ]* i ((H a)/CA1))

ICI <- ([ ]* i CA1NOFI? C i)

IGE <- ([ ]* i CA1NOFI? G e)

# forethought logical connectives
```

```
KA0 <- ((K a)/(K e)/(K o)/(K u)/(K i H a)/(N u K u))

# causal and comparative modifiers

KOU <- ((K OU)/(M OI)/(R AU)/(S OA)/(M OU)/(C IU))

# negative and converse forms

KOU1 <- (((N u N o)/(N u)/(N o)) KOU)

# the full type of forethought connectives, adding the causal and comparative connect

KA <- ([ ]* ((KA0)/((KOU1/KOU) K i)) NOI?)

# the last component of the KA...KI... structure of forethought connections

KI <- ([ ]* (K i) NOI?)

# causal and comparative modifiers which are *not* forethought connectives

KOU2 <- (KOU1 !KI)

# a test used to at least partially enforce the penultimate stress rule on quantifier

BadNIStress <- ((C1 V2 V2? stress (M a)? (M OA)? NI RA)/(C1 V2 stress V2 (M a)? (M OA

# root quantity words, including the numerals

NI0 <- (!BadNIStress ((K UA)/(G IE)/(G IU)/(H IE)/(H IU)/(K UE)/(N EA)/(N IO)/(P EA)/

# the class of SA roots, which modify quantifiers

SA <- (!BadNIStress ((S a)/(S i)/(S u)/(IE (comma2? !IE SA)?)) NOI?)

# the family of quantifiers which double as suffixes for the quantifier predicates
# this class perhaps should also include some other quantifier words. <re> for exampl
# No action here, just a remark.
```

```
RA <- (!BadNIStress ((R a)/(R i)/(R o)/R e/R u))

# re and ru added to class RA 5/11/18

# quantifier units consisting of a NI or RA root with <ma> 00 or <moa> 000 app
# append a digit to iterate <moa>:  <fomoate> is four billion, for example.  <

# a NI1 or RA1 may be followed by a pause before another NI word other than a
# one is allowed to breathe in the middle of long numerals.  I question whethe
# provision makes sense in RA1.

NI1 <- ((NI0 (!BadNIStress M a)? (!BadNIStress M OA NI0*)?) (comma2 !(NI RA) &

RA1 <- ((RA (!BadNIStress M a)? (!BadNIStress M OA NI0*)?) (comma2 !(NI RA) &N

# a composite NI word, optional SA prefix before a sequence of NI words or a R
# or a single SA word [which will modify a default quantifier not expressed],
# possibly negated, connected with CA0 roots to other such constructs.

NI2 <- (( (SA? (NI1+/RA1))/SA) NOI? (CA0 ((SA? (NI1+/RA1))/SA) NOI?)*)

# a full NI word with an acronymic dimension (starting with <mue>, ending with
# and figure out its semantics.  An arbitrary name word may now be used as a d

NI <- ([ ]* NI2 (&(M UE) Acronym (comma/&end/&period) !(C u)/comma2? M UE comm

# mex is now identical with NI, but it’s in use in later rules.

mex <- ([ ]* NI)

# a word used for various tightly binding constructions:  a sort of verbal hyp
# also a name marker, which means phonetic care is needed (pause after constru

CI <- ([ ]* (C i))

# Acronyms, which are names (not predicates as in 1989 Loglan) or dimensions (
# units in acronym are TAI0 letterals, zV short forms for vowels, the dummy un
# quantity units.  NI1 quantity units may not be initial. <mue> units may be p
# An acronym has at least two units.

# it is worth noting that acronyms, once viewed as names, could be entirely suppresse
# grammar by really making them names (terminate them with -n).  I suppose a similar
# for dimensions, allowing any name word to serve as a dimension.  <mue> would be a n
# with dimensions in this case.  <temuedain>, three dollars.  Now supported.

Acronym <- ([ ]* &caprule ((M UE)/TAI0/(Z V2 !V2)) ((comma &Acronym M UE)/NI1/TAI0/(Z

# the full class of letterals, including the <gao> construction whose details I shoul

TAI <- ([ ]* (TAI0/((G AO) !V2 [ ]* (PreName/Predicate/CmapuaUnit))))

# atomic non-letteral pronouns.

#4/15/2019 reserved <koo> for a Lojban style imperative pronoun, though not officiall

DA0 <- ((T AO)/(T IO)/(T UA)/(M IO)/(M IU)/(M UO)/(M UU)/(T OA)/(T OI)/(T OO)/(T OU)/

# letterals (not including <gao> constructions and atomic pronouns optionally suffixe
# suffixed forms, because <ci> is a name marker.

DA1 <- ((TAI0/DA0) (C i ![ ] NI0)?)

# general pronoun words.

DA <- ([ ]* DA1)

# roots for PA words:  tense and location words, prepositions building relative modif

PA0 <- (NI2? (N u !KOU)? ((G IA)/(G UA)/(P AU)/(P IA)/(P UA)/(N IA)/(N UA)/(B IU)/(F 

# the form used for actual prepositions and suffixes to A words, with minimal pauses
# these are built by concatenating KOU2 and PA0 units, then linking these with CA0 ro
# no- prefixes and -noi suffixes, and next to which one *can* pause), optionally suff

PANOPAUSES <- ((KOU2/PA0)+ ((comma2? CA0 comma2?) (KOU2/PA0)+)*)
```

```
# prepositional words

PA3 <- ([ ]* PANOPAUSES)

# class PA can appear as tense markers or as relative modifiers without argume
# are allowed not only next to CA0 units but between KOU2/PA units.  Like NI w
# words are a class of arbitrary length constructions, and we think breaths wi
# (especially complex ones) are natural.

PA <- ((KOU2/PA0)+ (((comma2? CA0 comma2?)/(comma2 !mod1a)) (KOU2/PA0)+)*) !mo

PA2 <- ([ ]* PA)

GA <- ([ ]* (G a))

# the class of tense markers which can appear before predicates.

PA1 <- ((PA2/GA))

# suffixes which indicate extent or remoteness/proximity of the action of prep

ZI <- ((Z i)/(Z a)/(Z u))

# the primitive description building "articles".  These include <la> which req
# care in its use because it is a name marker.

LE <- ([ ]* ((L EA)/(L EU)/(L OE)/(L EE)/(L AA)/(L e)/(L o)/(L a)))

# articles which can be used with abstract descriptions:  these include some q
# this means that some abstract descriptions are semantically indefinites:  I
# could be improved by having a separate abstract indefinite construction.

LEFORPO <- ([ ]* ((L e)/(L o)/NI2))

# the numerical/quantity article.

LIO <- ([ ]* (L IO))
```

```
# structure words for the ordered and unordered list constructions.

LAU <- ([ ]* (L AU))

LOU <- ([ ]* (L OU))

LUA <- ([ ]* (L UA))

LUO <- ([ ]* (L UO))

ZEIA <- ([ ]* Z EIb a)

ZEIO <- ([ ]* Z EIb o)

# initial and final words for quoting Loglan utterances.

LI1 <- (L i)

LU1 <- (L u)

# quoting Loglan utterances, with or without explicit double quotes (if they appear,
# appear on both sides).  The previous version allowed quotation of names;  likely th
# be restored.

LI <- ([ ]* LI1 comma2? utterance0 comma2? LU1/[ ]* LI1 comma2? [\"] utterance0 [\"]

# the foreign name construction.  This is an alien text construction

LAO <- ([ ]* &([Ll] [Aa] [Oo]juncture?) AlienWord)

# the strong quotation construction.  This is an alien text construction.

LIE <- ([ ]* &([Ll] [Ii] juncture? [Ee]juncture?) AlienWord)

LIO <- ([ ]* &([Ll] [Ii] juncture? [Oo]juncture?) AlienWord)
```

```
# I am not sure this class is used at all.

LW <- Cmapua

# articles for quotation of words

LIU0 <- ((L IU)/(N IU))

# this now imposes the condition that an explicit comma pause (or terminal pun
# Word or PreName quoted with <liu>.  This seems like a good idea, anyway.

# this class appeals to the phonetics.  Words and PreNames can be quoted.  The
# here may remove the need to quote them with <li>...<lu>.  Of course, some Wo
# than single words:  we will see whether the privileges afforded are used.  T
# use of letterals as actual names of letters.

# added <niu>:  didn't make it a name marker.

LIU1 <- ([ ]* ([Ll]/[Nn])[iI] juncture? [Uu] juncture? !V1 comma2? (PreName/Wo

# the construction of foreign and onomatopoeic predicates.  These are alien te

SUE <- ([ ]* &([Ss] [Uu] juncture? [Ee] juncture?/[Ss] [Aa] [Oo] juncture?) Al

# left marker in a predicate metaphor construction

CUI <- ([ ]* (C UI) )

# other uses of GA

GA2 <- ([ ]* (G a) )

# ge/geu act as "parentheses" to make an atomic predicate from a complex metap
# and logically connected predicates;  <ge> has other left marking uses.

GE <- ([ ]* (G e) )

GEU <- ([ ]* ((C UE)/(G EU)) )
```

```
# final marker of a list of head terms

GI <- ([ ]* ((G i)/(G OI)) )

# used to move a normally prefixed metaphorical modifier after what it modifies.

GO <- ([ ]* (G o) )

# marker for second and subsequent arguments before the predicate; NEW

GIO <- ([ ]* (G IO) )

# the generic right marker of many constructions.

GU <- ([ ]* (G u) )

# various flavors of right markers.

# It should be noted that at one point I executed a program of simplifying these to
# reduce the likelihood that multiple <gu>'s would ever be needed to close an utteran
# first of all, I made the closures leaner, moving them out of the classes closed
# to their clients so that they generally can be used only when needed.
# Notably, the grammar of <guu> is quite different.   Second,
# I introduced some new flavors of right marker.  All can be realized with <gu>,
# but if one knows the right flavor one can close the right structure with a single
# right closure.

# right markers of subordinate clauses (argument modifiers).
# <gui> closes a different class than in the trial.85 grammar, with
# similar but on the whole better results.

GUIZA <- ([ ]* (G UI) (Z a) )

GUIZI <- ([ ]* (G UI) (Z i) )

GUIZU <- ([ ]* (G UI) (Z u) )
```

```
GUI <- (!GUIZA !GUIZI !GUIZU ([ ]* (G UI) ))

# right markers of abstract predicates and descriptions.
# probably the forms with z are to be preferred (and the other
# two are not needed) but I preserve all five classes for now.

GUO <- ([ ]* (G UO) )

GUOA <- ([ ]* (G UOb a/G UO Z a) )

GUOE <- ([ ]* (G UOb e) )

GUOI <- ([ ]* (G UOb i/G UO Z i) )

GUOO <- ([ ]* (G UOb o) )

GUOU <- ([ ]* (G UOb u/G UO Z u) )

# right marker used to close term (argument/predicate modifier) lists.
# it is important to note that in our grammar GUU is not a component of
# the class termset, nor is it a null termset:  it appears in other classes
# which include termsets as an option to close them.  The effects are similar
# to those in the trial.85 grammar, but there is less of a danger that
# extra unexpected closures will be needed.

GUU <- ([ ]* (G UU) )

# a new closure for arguments in various contexts

GUUA <- ([ ]* (G UUb a) )

# a new closure for sentences.  In particular, it
# may have real use in closing up the scope of a list of
# fronted terms before a series of logically connected sentences.

GIUO <- ([ ]* (G IUb o) )

# right marker used to close arguments tightly linked with JE/JUE.
```

```
GUE <- ([ ]* (G UE) )

# a new closure for descpreds

GUEA <- ([ ]* (G UEb a) )


# used to build tightly linked term lists.

JE <- ([ ]* (J e) )

JUE <- ([ ]* (J UE) )

# used to build subordinate clauses (argument modifiers).

JIZA <- ([ ]* ((J IE)/(J AE)/(P e)/(J i)/(J a)/(N u J i)) (Z a) )

JIOZA <- ([ ]* ((J IO)/(J AO)) (Z a) )

JIZI <- ([ ]* ((J IE)/(J AE)/(P e)/(J i)/(J a)/(N u J i)) (Z i) )

JIOZI <- ([ ]* ((J IO)/(J AO)) (Z i) )

JIZU <- ([ ]* ((J IE)/(J AE)/(P e)/(J i)/(J a)/(N u J i)) (Z u) )

JIOZU <- ([ ]* ((J IO)/(J AO)) (Z u) )

JI <- (!JIZA !JIZI !JIZU ([ ]* ((J IE)/(J AE)/(P e)/(J i)/(J a)/(N u J i)) ))

JIO <- (!JIOZA !JIOZI !JIOZU ([ ]* ((J IO)/(J AO)) ))

# case tags, both numerical position tags and the optional semantic case tags.

DIO <- ([ ]* ((B EU)/(C AU)/(D IO)/(F OA)/(K AO)/(J UI)/(N EU)/(P OU)/(G OA)/(S AU)/(

# markers of indirect reference.  Originally these had the same grammar as case tags,
# but they are now different.
```

```
LAE <- ([ ]* ((L AE)/(L UE)) )

# <me> turns arguments into predicates, <meu> closes this construction.

ME <- ([ ]* ((M EA)/(M e)) )

MEU <- ([ ]* M EU )

# reflexive and conversion operators:  first the root forms, then those with
# optional numerical suffixes.

NUO <- ((N UO)/(F UO)/(J UO)/(N u)/(F u)/(J u))

NU <- [ ]* (((N u/N UO) !([ ]+ (NI0/RA)) (NI0/RA)?)/NUO)+ freemod?

# abstract predicate constructors (from sentences)

# I do *not* think
# that <poia> will really be confused with <po ia>, particularly
# since we do require an explicit pause before <ia> in the latter case,
# but I record this concern:  the forms with z might be preferable.

PO1 <- ([ ]* ((P o)/(P u)/(Z o)))

PO1A <- ([ ]* ((P OIb a)/(P UIb a)/(Z OIb a)/(P o Z a)/(P u Z a)/(Z o Z a)))

PO1E <- ([ ]* ((P OIb e)/(P UIb e)/(Z OIb e)))

PO1I <- ([ ]* ((P OIb i)/(P UIb i)/(Z OIb i)/(P o Z i)/(P u Z i)/(Z o Z i)))

PO1O <- ([ ]* ((P OIb o)/(P UIb o)/(Z OIb o)))

PO1U <- ([ ]* ((P OIb u)/(P UIb u)/(Z OIb u)/(P o Z u)/(P u Z u)/(Z o Z u)))

# abstract predicate constructor from simple predicates

POSHORT1 <- ([ ]* ((P OI)/(P UI)/(Z OI)))
```

```
# word forms associated with the above abstract predicate root forms

PO <- ([ ]* PO1 )

POA <- ([ ]* PO1A )

POE <- ([ ]* PO1E )

POI <- ([ ]* PO1E )

POO <- ([ ]* PO1O )

POU <- ([ ]* PO1U )

POSHORT <- ([ ]* POSHORT1 )

# register markers

DIE <- ([ ]* ((D IE)/(F IE)/(K AE)/(N UE)/(R IE)) )

# vocative forms:  I still have the words of social lubrication as
# vocative markers.

HOI <- ([ ]* ((H OI)/(L OI)/(L OA)/(S IA)/(S IE)/(S IU)) )

# the verbal scare quote.  The quantifier suffix indicates how many preceding words a
# this is an odd mechanism.

JO <- ([ ]* (NIO/RA/SA)? (J o) )

# markers for forming parenthetical utterances as free modifiers.

KIE <- ([ ]* (K IE) )

KIU <- ([ ]* (K IU) )

KIE2 <- [ ]* K IE comma2? [(]
KIU2 <- [ ]* [)] comma2? K IU

# marker for forming smilies.

SOI <- ([ ]* (S OI) )

# a grab bag of attitudinal words, including but not restricted to the VV form

UI0 <- (!predstart (!([Ii] juncture? [Ee]) VV juncture?/(B EA)/(B UO)/(C EA)/(

# negative forms of the attitudinals.  The ones with <no> before the two vowel
# should also be (though they present no pronunciation problem) so that they a

NOUI <- (([ ]* UI0 NOI)/([ ]* N [o] juncture? comma? [ ]* UI0 ))

# all attitudinals (adding the discursives nefi, tofi... etc)
# there is a technical problem with mixing UI0 roots of VV and CVV shapes.

UI1 <- ([ ]* (UI0+/(NI F i)) )

# the inverse vocative marker

HUE <- ([ ]* (H UE))

# occurrences of <no> as a word rather than an affix.

NO1 <- ([ ]* !KOU1 !NOUI (N o) !(comma2? Z AO comma2? Predicate) !([ ]* KOU) !

# a technical closure for the alternative parser approach:  the "large subject

GAA <- (NO1 freemod?)* ([ ]* (G AA))


# Names, acronyms and PreNames from above.

AcronymicName <- Acronym &(comma/period/end)
```

```
DJAN <- (PreName/AcronymicName)

# predicate words which are phonetically cmapua

# "identity predicates".  Converses are provided as a new proposal.

BI <- ([ ]* (N u)? ((B IA)/(B IE)/(C IE)/(C IO)/(B IA)/(B [i])) )

# interrogative and pronoun predicates

LWPREDA <- ((H e)/(D UA)/(D UI)/(B UA)/(B UI))

# here I should reinstall the <zao> proposal.

# the predicate words defined above in the phonetics section

Predicate <- (CmapuaUnit comma2?  Z AO comma2?)* Complex (comma2? Z AO comma2? Predic

# predicate words, other than the "identity predicates" of class [BI]
# these include the numerical predicates (NI RA), also cmapua phonetically.

# we are installing John Cowan's <zao> proposal here, experimentally, 4/15/2019

PREDA <- ([ ]* &caprule (Predicate/LWPREDA/(![ ] NI RA)) )

# Part 3:  The Grammar Proper

# right markers turned into classes.

guoa <- (PAUSE? (GUOA/GU) freemod?)

guoe <- (PAUSE? (GUOE/GU) freemod?)

guoi <- (PAUSE? (GUOI/GU) freemod?)

guoo <- (PAUSE? (GUOO/GU) freemod?)

guou <- (PAUSE? (GUOU/GU) freemod?)
```

```
guo <- (!guoa !guoe !guoi !guoo !guou (PAUSE? (GUO/GU) freemod?))

guiza <- (PAUSE? (GUIZA/GU) freemod?)

guizi <- (PAUSE? (GUIZI/GU) freemod?)

guizu <- (PAUSE? (GUIZU/GU) freemod?)

gui <- (PAUSE? (GUI/GU) freemod?)

gue <- (PAUSE? (GUE/GU) freemod?)

guea <- (PAUSE? (GUEA/GU) freemod?)

guu <- (PAUSE? (GUU/GU) freemod?)

guua <- (PAUSE? (GUUA/GU) freemod?)

giuo <- (PAUSE? (GIUO/GU) freemod?)

meu <- (PAUSE? (MEU/GU) freemod?)

geu <- GEU

# Here note the absence of pause/GU equivalence.

gap <- (PAUSE? GU freemod?)

# this is the vocative construction.  It can appear early because all of its c

# the intention is to indicate who is being addressed.  This can be handled vi
# alien text name (the last must be quoted).  The complexities of these gramma
# introduced.

# HOIO <- [ ]* [Hh] [Oo] [Ii] juncture?

# restore words of social lubrication as vocative markers but not as name mark
# I do not allow a freemod to intervene between a vocative marker and the associated
# utterance, to avoid unintended grabbing of subjects by the words of social lubricat
# as vocative markers.  This lets <Loi, Djan> and <Loi hoi Djan> be equivalent.  The
# first because the social lubrication words are in this version not name markers.

HOI0 <- ([ ]* ((([Hh] OI)/([Ll] OI)/([Ll] OA)/([Ss] IA)/([Ss] IE)/([Ss] IU)))) junctur

voc <- (HOI0 comma2? name /(HOI comma2? descpred guea? namesuffix?)/(HOI comma2? argu

# this is the inverse vocative.  It can appear early because all of its components ar

# the intention is to indicate who is speaking.  The range of ways this can be handle
# handled for the vocative;  there is the further option of a sentence (the [statemen
# for the case where an argument is used (to avoid it inadvertantly expanding to a se

HUE0 <- [ ]* &caprule [Hh] [Uu] juncture? [Ee] juncture? !V1

invvoc <- (HUE0 comma2? name/HUE freemod? descpred guea? namesuffix?/(HUE freemod? st


# this is the class of free modifiers.  Most of its components are head marked (those
# and it is useful for it to appear early because these things appear everywhere in s
# of whatever sort, is a freely insertable gadget which modifies the immediately prec
# if it is initial.

# NOUI is a negated attitudinal word.  UI1 is an attitudinal word:  these express an
# assertion (noting that EI marks questions (yes or no answer expected) and SEU marks

# SOI creates smilies in a general sense:  <soi crano> indicates that the listener sh
# similarly for other predicates.

# DIE and NO DIE are register markers, communicating the social attitude of the speak
# example is "dear"

# KIE...KIU constructs a full parenthetical utterance as a comment, which can be encl
# the marker words.
# JO is a scare quote device.

# the comma is a freemod with no semantic content:  this is a device for disca
# and the speaker's optional pauses alike.   The pause before a non-pause mark
# is excluded.  Ellipses and dashes are fancy pauses supported as freemods.

freemod <- ((NOUI/(SOI freemod? descpred guea?)/DIE/(NO1 DIE)/(KIE comma? utte

# the classes juelink to linkargs describe very tightly bound arguments which
# the context of metaphorical modifications and the use of predicates in descr

# note that we allow predicate modifiers (prepositional phrases) to be bound w
# allowed in 1989 Loglan, but which we believe is supported in Lojban.

juelink <- (JUE freemod? (term/(PA2 freemod? gap?)))

links1 <- (juelink (freemod? juelink)* gue?)

links <- ((links1/(KA freemod? links freemod? KI freemod? links1)) (freemod? A

jelink <- (JE freemod? (term/(PA2 freemod? gap?)))

linkargs1 <- (jelink freemod? (links/gue)?)

linkargs <- ((linkargs1/(KA freemod? linkargs freemod? KI freemod? linkargs1))

# class abstractpred supports the construction of event, property, and quantit
# closable with <guo> if introduced with <po,pu,zo> and closable with suffixed
# variants of <po,pu,zo> (a NEW idea but it is clear that closure of these pre
# used associated descriptions) is an important issue).

abstractpred <- ((POA freemod? uttAx guoa?)/(POA freemod? sentence guoa?)/(POE

# predunit1 describes the truly atomic forms of predicate.

# PREDA is the class of predicate words (the phonetic predicate words along wi
# above under the PREDA rule.  NU PREDA handles permutations and identificatio
# SUE contains the alien text constructions with <sao> and <sue>, semantically quite 
# in the same way.

# <ge>...<geu/cue> (the closing optional) can parenthesize a fairly complex predicate
# forms can have conversion or reflexive operators (NU) applied.  I should look into 
# is different.  An important use of this is in metaphor constructions, but it has ot

# abstractpred is the class of abstraction predicates just introduced above.  These a
# be noted that their privileges in the trial.85 grammar are (absurdly) limited.

# <me>...<meu> (the closing optional, but important to have available) forms predicat
# objects to which the argument refers.  <Ti me le mrenu> :  this is one of the men w

predunit1 <- ((SUE/(NU freemod? GE freemod? despredE (freemod? geu comma?)?)/(NU free

# <no> binds very tightly to predunit1:  a possibly multiply negated predunit1 (or an

predunit2 <- ((NO1 freemod?)* predunit1)

# an instance of NO2 is one not absorbed by a predunit.  Example:  <Da no kukra prano
# <Da no ga kukra prano>  (X is not a fast runner, and in fact may not run at all).

NO2 <- (!predunit2 NO1)

# a predunit3 is a predunit2 with tightly attached arguments.

predunit3 <- ((predunit2 freemod? linkargs)/predunit2)

# a predunit is a predunit3 or a predunit3 converted by the short-scope abstraction o
# <poi/pui/zoi> to an abstraction predicate.  This is the kind of predicate which can
# a component in a serial name.

predunit <- ((POSHORT freemod?)? predunit3)

# a further "atomic" (because tightly packaged) form is a forethought connected pair 
# of predicates (this being the full predicate class defined at the end of the proces
# possibly closed with <guu>, possibly multiply negated as well.
# the closure with guu eliminated the historic rule against kekked heads of me

kekpredunit <- ((NO1 freemod?)* KA freemod? predicate freemod? KI freemod? pre

# there follows the construction of metaphorically modified predicates,
# along with tightly logically linked predicates.

# CI and simple juxtaposition of predicates both represent modification of the
# predicate by the first.  We impose no semantic conditions on this modificati
# except in the case of modification by predicates logically linked with CA,
# which do distribute logically in the expected way both as modifiers and as m
# We do not regard <preda1 preda2> as necessarily implying preda2:  we do rega
# it as having the same place structure as preda2.  It is very often but not a
# a qualification or kind of preda2;  in any case it is a relation analogous t

# modification with CI binds most tightly.

# we eliminated the distinction between the series of sentence and description
# predicate preliminary classes:  there seems to be no need for it even in the
# trial.85 grammar.

despredA <- ((predunit/kekpredunit) (freemod? CI freemod? (predunit/kekpreduni

# this is logical connection of predicates with the tightly binding CA
# series of logical connectives.  CUI can be used to expand the scope of
# a CA connective over a metaphor on the left.  <ge>...<geu> is used to expand
# scope on the right (and could also be used on the left, it should be noted).
# descpredC is an internal of despredB assisting the function of CUI.
# the !PREDA in front of CUI is probably not needed.

despredB <- ((!PREDA CUI freemod? despredC freemod? CA freemod? despredB)/desp

despredC <- (despredB (freemod? despredB)*)

# tight logical linkage of despredB's

despredD <- (despredB (freemod? CA freemod? despredB)*)
```

```
# chain of modifications of despredD's (grouping to the left)

despredE <- (despredD (freemod? despredD)*)

# the GO construction allows inverse modification:  <preda1 GO preda2> is <preda2 pre
#  there are profound effects on grouping.

descpred <- ((despredE freemod? GO freemod? descpred)/despredE)

# this version which appears in sentence predicates as opposed to descriptions differ
# in allowing loosely linked arguments (termsets) instead of those linked with <je/ju
# moved to the end by GO.

# 4/17/2019 shared argument experiment

# sentpred <- (( (KA freemod? sentpred freemod? KI freemod? sentpred !guu)/ (despredE

sentpred <- ((despredE freemod? GO freemod? barepred)/despredE)

# sentpred <- ((despredE freemod? GO freemod? barepred)/despredE) (A1 ((despredE free

# the construction of predicate modifiers (prepositional phrases usable as terms alon

mod1a <- (PA3 freemod? argument1 guua?)

# note special treatment of predicate modifiers without actual arguments.
# the !barepred serves to distinguish these predicate modifiers from actual
# "tenses" (predicate markers).

mod1 <- ((PA3 freemod? argument1 guua?)/(PA2 freemod? !barepred gap?))

# forethought connection of modifiers.  There is some subtlety in
# how this is handled.

kekmod <- ((NO1 freemod?)* (KA freemod? modifier freemod? KI freemod? mod))

mod <- (mod1/((NO1 freemod?)* mod1)/kekmod)
```

```
# afterthought connection of modifiers

modifier <- (mod (A1 freemod? mod)*)

# the serial name is a horrid heterogenous construction!  It can involve
# components of all three of the major phonetic classes essentially!

# However, I believe I have the definition right, with all the components
# correctly guarded :-)

name <- (PreName/AcronymicName) (comma2? !FalseMarked PreName/comma2? &([Cc] [

LAO <- [ ]* [Ll] [Aa] juncture?

LANAME <- (LAO comma2? name)

# general constructions of arguments with "articles".

# the rules here have the "possessive" construction as in <lemi hasfa; le la D
# construction in 1989 Loglan, though speakers might think they are.  Here the
# be "indefinite" (cannot start with a quantifier word);  the possessor can be
# <le la Djan, na hasfa>, "John's present house", by analogy with <lemina hasf
# LIP accepts <lemina> as a word).

# there are other subtleties to be reviewed.

#descriptn <- (!LANAME ((LAU wordset1)/(LOU wordset2)/(LE freemod? ((!mex arg1

descriptn <- (!LANAME ((LAU wordset1)/(LOU wordset2)/(LE freemod? ((!mex arg1a


# abstract descriptions.  Note that abstract descriptions are closed with <guo
# <le po preda guo> does not have a grammatical component <po preda guo>.  Thi
# in Lojban.

abstractn <- ((LEFORPO freemod? POA freemod? uttAx guoa?)/(LEFORPO freemod? PO

# a wider class of basic argument constructions.  Notice that LANAME is always
```

```
namesuffix <- (&(comma2 !FalseMarked PreName/[ ]* [Cc][Ii] juncture? comma2? (PreName

arg1 <- (abstractn/(LIO freemod? descpred guea?)/(LIO freemod? argument1 guua?)/(LIO 

# this adds pronouns (incl. the fancy <gao> letterals) and the option of left marking

arg1a <- ((DA/TAI/arg1/(GE freemod? arg1a)) freemod?)

# argument modifiers (subordinate clauses)

argmod1 <- ((([ ]* (N o) [ ]*)? ((JI freemod? predicate)/(JIO freemod? sentence)/(JIO

# we improved the trial.85 grammar by closing not argmod1 but argmod with <gui>.  But
# when building an argmod1 have the argmod1 construction closed with the correspondin
# gui and guiza actually have different grammar.

# trial.85 did not provide forethought connected argument modifiers, and we also see 
# though they could readily be added.

argmod <- (argmod1 (A1 freemod? argmod1)* gui?)

# affix argument modifiers to a definite argument

arg2 <- (arg1a freemod? argmod*)

# build a possibly indefinite argument from an argument:  to le mrenu

arg3 <- (arg2/(mex freemod? arg2))

# build an indefinite argument from a predicate

indef1 <- (mex freemod? descpred)

# affix an argument modifier to an indefinite argument

indef2 <- (indef1 guua? argmod*)
```

```
indefinite <- indef2

# link arguments with the fusion connective <ze>

arg4 <- ((arg3/indefinite) (ZE2 freemod? (arg3/indefinite))*)

# forethought connection of arguments.  Note use of argx

arg5 <- (arg4/(KA freemod? argument1 freemod? KI freemod? argx))

# arguments with possible negations followed by possible indirect reference co

argx <- ((NO1 freemod?)* (LAE freemod?)* arg5)

# afterthought connection with the tightly binding ACI connectives

arg7 <- (argx freemod? (ACI freemod? argx)?)

# afterthought connection with the usual A connectives.  Can't start with GE
# to avoid an ambiguity (to which 1989 Loglan is vulnerable) involving AGE con

arg8 <- (!GE (arg7 freemod? (A1 freemod? arg7)*))

# afterthought connection (now right grouping, instead of the left grouping ab
# using the AGE connectives.  GUU can be used to affix an argument modifier at

argument1 <- (((arg8 freemod? AGE freemod? argument1)/arg8) (GUU freemod? argm

# possibly negated and case tagged arguments.  We (unlike 1989 Loglan) are car
# to use argument only where case tags are appropriate.

argument <- ((NO1 freemod?)* (DIO freemod?)* argument1)
```

```
# an argument which is actually case tagged.

argxx <- (&((NO1 freemod?)* DIO) argument)

# arguments and predicate modifiers actually associated with predicates.

term <- (argument/modifier)

# a term list consisting entirely of modifiers.

modifiers <- (modifier (freemod? modifier)*)

# a term list consisting entirely of modifiers and tagged arguments.

modifiersx <- ((modifier/argxx) (freemod? (modifier/argxx))*)

# the subject class is a list of terms (arguments and predicate modifiers) in which a
# of the arguments are tagged, and there is at least one argument, tagged or otherwis

subject <- ((modifiers freemod?)? ((argxx subject)/(argument (modifiersx freemod?)?))

# this case is identified as an aid to experimental termination of argument lists

statement1 <- (subject freemod? (GIO freemod? terms1)? predicate)

# these classes are exactly argument, but are used to signal
# which argument position after the predicate an argument occupies.
# I think the grammar is set up so that these will actually
# never be case tagged, though the grammar does not expressly forbid it.

# I am trying a simple version of the "alternative parser" approach:
# a term list will refuse to digest an argument which starts a new
# SVO sentence (statement1).

argumentA <- !statement1 argument

# argumentA <- argument
```

```
argumentB <- !statement1 argument

#  argumentB <- argument

argumentC <- !statement1 argument

#  argumentC <- argument

argumentD <- !statement1 argument

#  argumentD <- argument

# for argument lists not guarded against absorbing a following subject

argumentA1 <- argument

argumentB1 <- argument

argumentC1 <- argument

argumentD1 <- argument

# a general term list.  It cannot contain more than four untagged arguments (t
# with the lettered subclasses given above).

terms <- ((modifiersx? argumentA (freemod? modifiersx)? argumentB? (freemod? m

# terms list not guarded against absorbing a following subject

terms1 <- ((modifiersx? argumentA1 (freemod? modifiersx)? argumentB1? (freemod

# innards of ordered and unordered list constructions.  These are something I
# unsatisfactory state in trial.85.  Note the use of comma words to separate i

word <- (arg1a/indef2)

words1 <- (word (ZEIA word)*)
```

```
words2 <- (word (ZEIO word)*)

wordset1 <- (words1? LUA)

wordset2 <- (words2? LUO)

# the full term set type to be affixed to predicates.

# forethought connection of term lists

termset1 <- (terms/(KA freemod? termset2 freemod? guu? KI freemod? termset1))

# afterthought connection of term lists.  There are cunning things going on here gett
# to work correctly.  Note that <guu> is NOT a null term list as it was in trial.85.

termset2 <- (termset1 (guu &A1)? (A1 freemod? termset1 (guu &A1)?)*)

# there is an interesting option here of a list of terms followed by <go> followed by
# intended to metaphorically modify the predicate to which the terms are affixed.  Is
# why we cannot have a more complex construction in place of terms?

termset <- ((terms freemod? GO freemod? barepred)/termset2)

# this is the untensed predicate with arguments attached.  Here is the principal locu
# of closure with <guu>, but it is deceptive to say that <guu> merely closes barepred
# as we have seen above, for example in [termset2].

# modified for 4/17/2019 shared argument experiment

barepred <- (sentpred freemod? ((termset guu?)/(guu (&termset)))?)

# barepred <- (sentpred freemod? ((termset guu?)/(guu (&termset/&A1)))?)

# tensed predicates

markpred <- (PA1 freemod? barepred)

# there follows an area in which my grammar looks different from trial.85.  Distinct
# marked and unmarked predicates are demonstrably not needed even in trial.85.
# connectives is plain weird in trial.85; here we treat ACI connectives in the
# binding more tightly.

# units for the ACI construction following -- possibly multiply negated bare o

# adding shared termsets to logically connected predicates are handled differe
# which uses a very elegant but dreadfully left-grouping rule which a PEG cann
# should be manageable.

backpred1 <- ((NO2 freemod?)* (barepred/markpred))

# ACI connected predicates.  Shared termsets are added.  Notice how we first g
# group backpreds.

backpred <- (((backpred1 (ACI freemod? backpred1)+ freemod? ((termset guu?)/(g

# A connected predicates; same comments as just above.  Cannot start with GE t

predicate2 <- (!GE (((backpred (A1 !GE freemod? backpred)+ freemod? ((termset

# predicate2's linked with right grouping AGE connectives (A and ACI are left

predicate1 <- ((predicate2 AGE freemod? predicate1)/predicate2)

# identity predicates from above, possibly negated

identpred <- ((NO1 freemod?)* (BI freemod? argument1 guu?))

# predicates in general.  Note that identity predicates cannot be logically co
# except by using forethought connection (see above).

predicate <- (predicate1/identpred)


# The gasent is a basic form of the Loglan sentence in which the predicate lea
# The basic structure is <PA word (usually a tense) or <ga>) followed optional
# <ga> followed by terms.  The list of terms after <ga> (if present) will eith
# at least one argument and no more than one untagged argument
# (a subject) [gasent1] or all the arguments of the predicate [gasent2].  We deprecat
# 1989 Loglan because they would cause unexpected reorientation of the arguments alre
# and further arguments were read after <ga>.  [barepred] is an untensed predicate po
# is "simply a verb", i.e., a predicate without arguments.

# there is a semantic change from 1989 Loglan reflected in a grammar change here:
# in [gasent1] the final (ga subject) is optional.  When it does not appear, the resu
# sentence is an observative (a sentence with subject omitted), not an imperative.
# Imperatives for us are unmarked.

# In the alternative version, the use of the large subject marker GAA can prevent ina

# 4/22 allowing general predicates in gasent.  Otherwise the spaces of observatives a

#gasent1 <- ((NO1 freemod?)* (GAA? freemod? PA1 freemod? barepred (GA2 freemod? subje

gasent1 <- ((NO1 freemod?)* (GAA? freemod? &markpred predicate (GA2 freemod? subject)

gasent2 <- ((NO1 freemod?)* (GAA? PA1 freemod? sentpred modifiers? (GA2 freemod? subj

gasent <- (gasent2/gasent1)

# this is the simple Loglan sentence in various basic orders.  The form "gasent" is d
# Predicate modifiers
# can be prefixed to the gasent.  The final form given here is the basic SVO sentence
#(arguments and predicate modifiers) containing at most one un-case-tagged argument.
#by <gio> followed by a list of terms (1989 Loglan allowed more than one untagged arg
#in which preceding constructions with errors in them may supply extra unintended arg
#a single argument before the predicate, followed by the predicate, which may itself
#(even multiple times).

# re <gio> and some other changes, in his comments on the NB3 grammar  JCB often note
# intends but which he thought were hard to implement in the machine grammar.  The ap
# in an SVO sentence was one of these (though later he takes it as a virtue that the
# consider it a virtue to have unmarked SOV after observing unintended parses appeari
# (which would not have been hard for JCB to implement, in fact) is our restriction o
# His comments make it clear that he does not want arguments among those terms.
```

```
statement <- (gasent/(modifiers freemod? gasent)/(subject freemod? GAA? freemo

# this is a forethought connected basic sentence.  It is odd (and actual odd r
# of these rules is of the very general class uttA1, which includes some quite

# 12/20/2017 I rewrote the rule in a more compact form.  This rule looks ahead
# for the moment notice that [sentence] will include [statement].

# 4/14 tentatively allowing initial modifiers here and leaving this out of utt
# The intention is to eliminate weird sentence fragments.

keksent <- modifiers? freemod? (NO1 freemod?)* (KA freemod? headterms? freemod

# sentence negation.  We allow this to be set off from the main sentence with
# it does not differ in meaning from the result of negating the first argument

neghead <- ((NO1 freemod? gap)/(NO2 PAUSE))

# this class includes [statement], predicate modifiers preceding a predicate (
# a predicate, and a keksent.  Of these, the first and third are imperatives.

# in the alternative version, the large subject marker GAA can prevent inadver

# 4/23/2019 added actual rule for imperative sentences.  This should not
# affect the parse in any essential way.

imperative <- ((modifiers freemod?)? GAA? !gasent predicate)

sen1 <- (neghead freemod?)* (imperative/statement/keksent)

# sen1 <- ((neghead freemod?)* ((modifiers freemod? GAA? !gasent predicate)/st

# the class [sentence] consists of sen1's afterthought connected with A connec

sentence <- (sen1 (ICA freemod? sen1)*)

# [headterms] is a list of terms (arguments and predicate modifiers) ending in
# causes all predicates in the [sen1] to share these arguments.  We propose either th
# appended to the argument list of each component of the [sen1], or that there is an
# of the headterms list, and the list is inserted at the appropriate position in each
# the condition described in Loglan I, which presupposes that we always know what the

headterms <- (terms GI)+

# this is the previous class prefixed with a list of fronted terms.
# we think the <giuo> closure might prove useful.

uttAx <- (headterms freemod? sentence giuo?)

# weird answer fragments

uttA <- ((A1/mex) freemod?)

# a broad class of utterances, including various things one would usually only say as
# that this utterance class can take terminal punctuation.

uttA0 <- sen1/uttAx

uttA1 <- ((sen1/uttAx/links/linkargs/argmod/(modifiers freemod? keksent)/terms/uttA/N

# possibly negated utterances of the previous class.

uttC <- ((neghead freemod? uttC)/uttA1)

# utterances linked with more tightly binding ICI sentence connectives.  Single sente
# if not linked with ICI or ICA.

uttD <- ((sentence period? !ICI !ICA)/(uttC (ICI freemod? uttD)*))

# utterances of the previous class linked with ICA.  I went to some trouble to ensure
# [sentence] is actually parsed as a sentence, not a composite uttD, which was a defi
# LIP and of the trial.85 grammar.

uttE <- (uttD (ICA freemod? uttD)*)
```

```
# utterances of the previous class linked with I sentence connectives.

uttF <- (uttE (I freemod? uttE)*)

# the utterance class for use in the context of parenthetical freemods or quot

utterance0 <- (!GE ((!PAUSE freemod period? utterance0)/(!PAUSE freemod period

# Notice that there are two passes here:  the parser first checks that the ent
# is phonetically valid, then returns and checks for grammatical validity.

# the full utterance class.  This goes to end of text, and incorporates the ph
# in which a freemod is initial.   The IGE connectives bind even more loosely
# left grouping.

utterance <- &(phoneticutterance !.) (!GE ((!PAUSE freemod period? utterance)/
```