Converting SPECIALIST to an NLP Lexicon

Ken Litkowski
CL Research
9208 Gue Road
Damascus, MD 20872 USA
ken@clres.com

Abstract

The Specialist Lexicon, made publicly by the National Library of Medicine, contains 519,694 terms, primarily highly specialized medical terms. The Lexicon, updated annually, is not alphabetically and is not specifically identified for computational purposes. We have converted these terms so they can be used as a dictionary. An ordinary dictionary contains about 120,000 entries, while Specialist contains 500,000 direct entries and another 350,000 variants. While the conversion is designed for creating a DIMAP dictionary, and enabling searching and other kinds of processes, the data may be useful in its raw form. This paper describes a help file containing a characterization of the Specialist lexicon and all the steps that were used for the conversion. The raw data has also been made available, where it can be downloaded in a zip file.[1]

UMLS Specialist Lexicon

The Unified Medical Language System, developed and made publicly available by the National Library of Medicine, includes a Specialist Lexicon that provides a linguistic characterization and dictionary for 519,694 terms. While the vast majority of these lexical entries are highly specialized medical terms, the Specialist Lexicon also comprehensively covers common words of the English language. Each entry provides a full characterization of its syntactic properties (noting here that the entries do not provide semantic information or subsenses for a syntactic sense).

The Specialist Lexicon is publicly available and is an excellent resource for syntactic information for its entries. However, the actual data is provided in forms that may not be convenient: (1) a flat file (LEXICON) without perceptible order, to which new entries are appended annually, and (2) a series of files suitable for use with highly-structured database systems.

CL Research has prepared an alphabetic version of the Specialist Lexicon from the file LEXICON (with 499,923 entries), with an accompanying alphabetic version of "variant" forms (with 342,808 entries). These lexicons, or dictionaries, are accessible with CL Research's DIMAP Dictionary Maintenance Program software, for which a publicly available demonstration version is available. With DIMAP, a user can manipulate entries in a dictionary in a variety of ways, including the creation of user-specified extractions from any dictionary. The full description of DIMAP is contained in its accompanying help file.

This help file provides detailed descriptions of how each part of speech type is characterized in the Specialist Lexicon and how the Specialist Lexicon was converted to DIMAP format. This conversion of the file LEXICON was performed with a Perl script, which is available in the distribution of the DIMAP Specialist Lexicon dictionaries.

[1] See https://www.clres.com/elec_dictionaries.php#rumls.

Description of the Specialist Lexicon

Details describing the Specialist NLP Tools are available at National Library of Medicine. A detailed description of the Specialist Lexicon is provided in the technical report, *The SPECIALIST LEXICON*, Allen C. Browne, Alexa T. McCray, Suresh Srinivasan, Lister Hill National Center for Biomedical Communications, National Library of Medicine, Bethesda, Maryland (REVISED: JUNE 2018, 2018 Editors: Amanda Payne, Destinee Tormey). The technical report describes the basic form an entry and then provides a description of the types of information that may be contained in an entry. A copy of this technical report is included in the distribution of the DIMAP alphabetic dictionaries.

The topics discussed in the report describe the treatment of spelling variation, syntactic part of speech, variants, complementation, nominalizations, acronyms, proper nouns, stative adjectives, interrogatives, negation, and pronouns. In addition to identifying how this information is coded in the data file, the report gives a full discussion of the linguistic phenomena that underlie the encoding. Thus, the report, combined with the data, provides a comprehensive treatment of basic grammatical phenomena in the English language.

Form of an Entry in SPECIALIST

In the distribution, the file LEXICON consists (in 2010) of over 499,900 entries in the format exemplified below. In the conversion of LEXICON to an alphabetic format accessible in DIMAP, these are the entries that are converted. In general, an entry is created for each **base**, with a sense created for each base that is alphabetically identical. For the example below, at least two senses are created for the entry *anesthetic*, one a noun sense and one an adjective sense. Any spelling variants (**spelling_variant=**) follow next, then the entry identifying number (**entry=**). Each base then has a part of speech category (**cat=**); the remaining data for the sense list the properties appropriate to the category. In addition, many entries are abbreviations or acronyms; these entries also have a category (usually noun) and the meaning of each letter is identified (e.g., ACT has several acronyms and abbreviations, in addition to "act" with a noun sense and a verb sense).

```
{base=anesthetic
spelling_variant=anaesthetic
entry=E0008769
  cat=noun
  variants=reg
}
{base=anesthetic
spelling_variant=anaesthetic
entry=E0008770
  cat=adj
  variants=inv
  position=attrib(3)
}
```

The number of entries increases annually (about 4,000, e.g., between 2019 to 2020). The development of the lexicon is described in the technical report. The use of the SPECLIALIST lexicon is emblematic by the paper: Ide, N., R. Loane, and D. Demner-Fushman. 2007. "Essie: A Concept-based Search Engine for Structured Biomedical Text", *Journal of the American Medical Informatics Association*, 14:3 (253-263).

Spelling Variants

The Specialist Lexicon may contain spelling variants for an entry. Spelling variants identify minor changes in the spelling that are considered to represent the same main entry. Spelling variants include

- capitalization of the first letter of a word (*Adriamycin* vs. *adriamycin*)
- presence or absence of a dash or space (*A-1* vs. *A1* or *Alu 1* vs. *Alu1*),
- presence or absence of an apostrophe (*Alzheimer's disease* vs. *Alzheimer disease*),
- British English or American English spellings (*lipedema* vs. *lipoedema*), and
- use of periods in abbreviations (*AAMD* vs. *A.A.M.D.*)

A spelling variant occurs in the Specialist Lexicon as the line:

spelling_variant=*variant*

An entry may have several spelling variants.

In the DIMAP entry, the spelling variants for an entry are contained in the set of **Features** for a sense, with a feature name of *var* and the variant(s) as the feature value (e.g., **var = adriamycin**), with multiple variants separated by semicolons.

Spelling variants also are included in the DIMAP Variants dictionary. In this dictionary, each spelling variant is a distinct entry in the DIMAP dictionary and contains only a single sense, with "none" as the part of speech. This sense has a DIMAP Feature **varOf** and a value equal to the base form of the entry. It is possible that a sense may have multiple **varOf** features, each pointing to a different base. The effect of spelling variants may be unusual. For example, the Specialist Lexicon contains a base "*altho*" with the variants *although*, *allthough*, and *altho'*, whereas users might expect *although* would be the main entry, with the others as the variants.

Entry Identifying Number

Each entry in the Specialist Lexicon has a unique identifier that begins with the letter **E** and is followed by 7 digits. In UMLS terminology, this identifier is known as an *Entry Unique Identifier*, or **EUI**. This item occurs in the Specialist Lexicon as the line:

entry=*identifying number*

In the DIMAP entry, this identifier is contained in the set of **Features** for a sense, with a feature name of *id* and the identifier as the feature value (e.g., **id = E0065594**).

Category (Part of Speech)

Each entry in the Specialist Lexicon has a unique category or part of speech. This item occurs in the Specialist Lexicon as the line:

cat=*part of speech*

The values for the part of speech in the Specialist Lexicon are (with the 3-letter code used in converting these entries into DIMAP):

- verb (vrb): verbs
- aux (aux): auxiliary verbs (e.g., *be*)
- modal (aux): modal verbs (e.g., *might* and *would*)
- noun (nou): common and proper nouns
- pron (pro): pronouns
- adj (adj): adjectives
- adv (adv): adverbs
- prep (prp): prepositions
- conj (ccj): conjunctions
- compl (rel): complementizers (only the word *that*, identified as a "relative" part of speech in DIMAP)
- det (det): determiners (e.g., *a* and *the*)

Most of these parts of speech have a set of properties. The links provide details about how these properties are handled and represented.

Nouns

Most of the noun properties in the Specialist Lexicon pertain to their inflection; these are identified as **variants**, described in the discussion of **Noun Variants**. The remaining properties pertain to noun complementation (**compl**) patterns, whether the noun is a proper noun (**proper**), whether the noun is a nominalization (**nominalization_of**), whether the noun is a trademark, and whether the noun is an acronym or abbreviation. These are described in the discussion of **Miscellaneous Noun Forms**. Details about these characteristics are provided in the technical report (pp. 9, 21-28, 51-54).

Many nouns are phrases, i.e., entry names containing a space; while these are not discussed in describing the Specialist Lexicon, their phraseology is important in computational processing and described in the DIMAP variants dictionary.

See DIMAP Noun Processing to describe the steps for characterize the properties of noun entries.

Noun Phrases

Noun phrases in the Specialist Lexicon, i.e., any entry that has a noun part of speech and that also contains a space, gives rise to two entries in the Specialist DIMAP variants dictionary. Noun phrases are processing in the end of entry processing.

- An entry is created for the **last word** in the phrase, with the full phrase identified as a DIMAP instance. For the entry *ACH index*, an entry is created for *index*, with an instance *ACH index*. (All phrases in Specialist ending with the word *index* are collected together in one sense in the variants dictionary.)
- An entry is created for the **first word** in the phrase, with the remainder of the phrase captured in a DIMAP feature with a name **kind** and a value consisting of a tilde ("~"), a space, and the remainder of the phrase. For the entry *ACH index*, an entry is created for *ACH*, with a feature **kind** and value equal to "~ index". (All phrases in Specialist beginning with the word *ACH* are collected together in one sense in the variants dictionary.)

Noun Variants

All information regarding the inflection of a noun is contained in the Specialist Lexicon in lines of the form:

variants=*variant*

The **variants** slot does not actually contain variants but rather contains codes that describe the syntactic characteristics of the noun. An entry may have several **variants** lines. Details about these characteristics are provided in the technical report in the section **Noun Inflection**; these details not only describe the particular codes, but also provide a thorough introduction to English grammar.

The following codes are used in the Specialist Lexicon, with the links describing the code, a brief description of the property, and how the information contained in the property is represented in DIMAP:

- Count nouns (**glreg**, **metareg**, **sing**, **plur**, **inv**, **irreg**, and **reg**)
- Uncount nouns (**uncount** and **groupuncount**)

Countability is a major property of nouns, affecting not only its inflection, but also forms that may occur in the context surrounding their use, particularly with determiners, types of adjectives, and verb forms.

Count Nouns

Count nouns in English have a plural.

A regular count noun follows the regular pattern of English plural formation, and is represented in the Specialist Lexicon with the line:

variants=reg

In DIMAP, a noun sense has a feature **reg** with a value **+**. Rules for the formation of regular plurals are provided in the technical report; these rules describe how a noun is pluralized depending on the end of the base word (one ending in **y**, **s**, **z**, **x**, **ch**, **sh**, or with an **s** if not ending with one of the above endings).

An irregular count noun has several possible plurals. Nouns of classical origin, which are common in the biomedical domain, often retain their Latin or Greek inflectional patterns. For these entries, this is represented with the line

variants=glreg

In DIMAP, a noun sense has a feature **infl** with a value **gl** (indicating that it has a Greco-Latin plural) and a feature **pl** with a value of the plural, using the GrecoLatinPlural subroutine. Rules for the formation of regular Greco-Latin plurals are provided in the technical report. These rules are followed in determining the plural form and an entry is created in the DIMAP variants dictionary for the plural form. This entry has a feature **plOf** ("plural of") with a value equal to the base form.

The plurals of acronyms (*Ph.D.'s* or *Ph.D.s*), numbers (*5's* or *5s*), and other orthographically meta-linguistic nouns may form their plural with the addition of the letter **s** or an apostrophe and the letter **s**. An entry that has this alternative has the line

variants=metareg

In DIMAP, a noun sense has a feature **infl** with a value **mr**.

The plurals of irregular nouns are explicitly listed in the Specialist lexicon. For example, the word *calf* has the following line, where the base form is repeated before the plural forms are listed.

variants=irreg|calf|calves|

An entry may have more than one irregular plural. In DIMAP, the entry for the base form has a feature **pl** with a value consisting of all plural forms separated by a semicolon. In addition, an entry is created in the DIMAP variants dictionary for each form. This entry has a feature **plOf** ("plural of") with a value equal to the base form. A given orthographic form may have several of these features, although this is a rare occurrence.

Some English nouns behave like count nouns but lack a plural form. Thus, the noun *clatter* acts like a singular noun, but does not have a plural form (see the technical report for other criteria). In the Specialist Lexicon, this is indicated by the line

variants=sing

In DIMAP, the sense has a feature **sing** with a value **+**.

Some English plural nouns are fixed plurals with no singular form, but still behave like count nouns (e.g., *cattle*, *police*, and *surroundings*). In the Specialist Lexicon, this is indicated by the line

variants=plur

In DIMAP, the sense has a feature **plur** with a value **+**.

Some English nouns use the same form for the singular and the plural (e.g., *sheep*, *deer*, *means*, and *nexus*). In the Specialist Lexicon, this is indicated by the line

variants=inv

In DIMAP, the sense has a feature **inv** with a value **+**.

Some English nouns are collective in number, referring to a group. The singular form of group nouns is indeterminate as to number and can agree with either a singular or plural verb. Some examples are *committee*, *family*, *herd*, and *bunch*.

Uncount Nouns

Uncount nouns in English do not have a plural. Many uncount nouns are abstract nouns (such as *sincerity*) or mass nouns (such as *beer*). Others are collective nouns (such as *United States*, *faculty*, and *mankind*).

Abstract and mass nouns agree only in the singular and may appear without a determiner. They are represented in the Specialist Lexicon with the line:

variants=uncount

In DIMAP, a noun sense has a feature **count** with a value **-** (minus sign).

Collective nouns may agree in either the singular or the plural.and may appear without a determiner. They are represented in the Specialist Lexicon with the line:

variants=groupuncount

In DIMAP, a noun sense has a feature **groupcount** with a value +/**-** (plus or minus sign).

Miscellaneous Noun Forms

Specialist provides specific lines in an item entry for further description of noun entries, including

- Proper nouns
- Noun complements
- Nominalization roots
- Trademarks
- Acronyms
- Abbreviations

Proper nouns are marked in Specialist with the feature **proper**, e.g., **Austria**. This appears as a simple line in an item entry.

proper

In DIMAP, a proper noun has a feature **proper** with a value + (plus sign).

Specialist identifies complement patterns, where appropriate, that are associated with noun entries. An entry may have more than one complement pattern, of the following forms (see the Technical Manual for a detailed linguistic description of these forms):

compl=infcomp:interp
compl=fincomp()
compl=whfincomp
compl=whinfcomp:interp
compl=ascomp:interp
compl=pphr(,)

These are infinitive clause complements (**infcomp**), finite clause complements (**fincomp**), WH finite clause complements (**whfincomp**), or "as" complements (**ascomp**). When there is a colon and **interp**, one of the complement codes is used,

In DIMAP, each of the complements is made into a feature **compl** with a value equal to whatever follows the **'='** sign, along with . In addition, the prepositonal phrase complements (**pphr**) are futher analyzed to extract the prepositions that are frequently collocated with the base form; the results of this analysis are printed in the file **comp.dmp**.

Specialist identifies noun entries that are nominalizations of other entries (i.e., noun forms derived from verbs or adjectives). For example, *Buddhism* and *abandonment* have the following lines in their entries:

nominalization_of=Buddhist|adj|E0001162
nominalization_of=abandon|verb|E0006429

Thus, Specialist identifies the underlying form, its part of speech, and its entry identifier.

In DIMAP, a noun that is also a nominalization is captured with a feature name **nomOf** and a feature value equal to the underlying form. The part of speech and entry identifier of the underlying form is not captured.

Entries that are trademarks are identified in Specialist and their generic names are also identified. For example, the entries *Adriamycin* and *alundum* have the lines

trademark=doxorubicin
trademark

In DIMAP, a trademark has a feature **tm** with a value set to what is given after the "=" sign, thus identifying the generic entry associated with the trademark. If nothing is given, no generic entry is identified and the value is + (plus sign), to indicate that the entry is a trademark name.

Adjectives

Adjective properties in the Specialist Lexicon pertain to their inflection, complements, and syntactic positions. Adjective inflections (i.e., how their comparative and superlative forms are created) are identified in a **variants** line, described in the discussion of **Adjective Inflections**. The remaining properties pertain to whether an adjective is stative, the possible positions of an adjective, adjective complementation (**compl**) patterns, whether the adjective can be nominalized (**nominalization**), and whether the adjective is an acronym or abbreviation. These are described in the discussion of **Miscellaneous Adjective Properties**. Details about these characteristics are provided in the technical report (pp. 9-10, 19-20, 56-58, 61-67).

See **DIMAP Adjective Processing** to describe the steps for characterizing the properties of adjective entries.

Adjective Inflections

All information regarding the inflection of an adjective is contained in the Specialist Lexicon in lines of the form:

variants=*variant*

The **variants** slot does not actually contain variants but rather contains codes that describe how the adjective is inflected (regular, regular doubling, irregular, invariant, or periphrastic). Details about these characteristics are provided in the technical report. See **DIMAP Adjective Processing** to describe the steps for characterizing adjective inflections.

Regular adjectives form their comparative and superlative according to specific rules for adding the comparative suffix *er* and the superlative suffix *est*. Regular adjectives are marked in Specialist with the line

variants=reg

In DIMAP, a regular adjective has a feature **infl** with a value **reg**.

Regular doubling adjectives follow the regular pattern, but double the final consonant before the suffixes are added (e.g., *fat*, *fatter*, *fattest*). Regular doubling adjectives are marked in Specialist with the line

variants=regd

In DIMAP, a regular adjective has a feature **infl** with a value **regd**.

Irregular adjectives do not follow regular patterns and the irregular forms are specifically entered. Irregular adjectives are marked in Specialist with the line

variants=irreg|bad|worse|worst|

In DIMAP, an irregular adjective has a feature **infl** with a value equal to the final two words, giving the comparative and superlative forms. In addition, an entry is created in the DIMAP variants dictionary for each form. These entries have a feature **compOf** ("comparative of") or **suplOf** ("superlative of") with a value equal to the base form.

Invariant adjectives (such as *medical* or *daily*) have no morphological comparative or superlative form. Invariant adjectives are marked in Specialist with the line

variants=inv

In DIMAP, an invariant adjective has a feature **infl** with a value **inv**.

Periphrastic adjectives form their comparative and superlative with *more* and *most*. Peripharstic adjectives are marked in Specialist with the line

variants=inv;periph

In DIMAP, a periphrastic adjective has a feature **infl** with a value **inv;periph**.

Miscellaneous Adjective Properties

Specialist provides specific lines in an item entry for further description of adjective entries, including

- Stative
- Attributive position
- Other adjective position
- Adjective complements
- Adjective nominalizations
- Acronyms

Stative adjectives are static in nature, meaning that they cannot be used with the progressive or imperative (see the technical manual for more details). They are marked in Specialist with the line

stative

In DIMAP, a stative adjective has a feature **stative** with a value + (plus sign).

Attribute adjectives occur between determiners and nouns. They are marked in Specialist with a line showing **attrib(N)** in the **position** slot, where **N** is a number between 1 and 3 identifying the order in which these adjectives (respectively, "qualitative", "color", "classifying") may occur between the determiners and noun, e.g.,

position=attrib(3)

The technical manual describes the kinds of adjectives that can occur in each of these positions.

In DIMAP, an attributive adjective has a feature **attrib** with a value equal to the number **N**.

Other adjective positions include predicative, post-nominal, and attributive with discontinuous complements. These are marked in Specialist with a property of the **position** field, i.e.,

position=pred
position=post
position=attribc

The technical manual describes these kinds of adjectives in more detail.

In DIMAP, an adjective which has one or more of these adjective positions has a feature **position** with a value equal to **pred**, **post**, and **attribc**, respectively.

Adjectives which have nominalized forms have the slot **nominalization** filled by the base form (of the noun), the category (noun), and its entry identifier, e.g., *abeyant* has the line

nominalization=abeyance|noun|E0006482

In DIMAP, an adjective that has a nominalization has a feature **nom** with a value equal to the base form of the noun.

Specialist identifies complement patterns, where appropriate, that are associated with adjective entries. An entry may have more than one complement pattern, of the following forms (see the Technical Manual for a detailed linguistic description of these forms):

compl=infcomp;interp
compl=binfcomp;interp
compl=fincomp()
compl=whfincomp
compl=whinfcomp
compl=advbl

compl=pphr(,)

In DIMAP, each of the complements is made into a feature **compl** with a value equal to whatever follows the **'='** sign. In addition, the prepositonal phrase complements (**pphr**) are futher analyzed to extract the prepositions that are frequently collocated with the base form; the results of this analysis are printed in the file **comp.dmp**.

Some adjectives are acronyms (*AB* - "able-bodied", **acronym_of**) or abbreviations (*AB* - "abdominal", **abbreviation_of**). Abbreviations and Acronyms are handled in End of Entry Processing.

Adverbs

Adverb properties in the Specialist Lexicon pertain to their inflection, modification type, effect on negation, and whether they are interrogative. Adverb inflections (i.e., how their comparative and superlative forms are created) are identified in a **variants** line, described in the discussion of **Adverb Inflections**. The remaining properties pertain to what kind of modification the adverb provides (**modification_type**), its effect on polarity (**negative**), whether it is an interrogative adverb, and whether the adverb is an acronym or abbreviation. These are described in the discussion of **Miscellaneous Adverb Properties**. Details about these characteristics are provided in the technical report (pp. 10, 20-21, 67-72).

See **DIMAP Adverb Processing** to describe the steps for characterizing the properties of adverb entries.

Adverb Inflections

All information regarding the inflection of an adverb is contained in the Specialist Lexicon in lines of the form:

variants=*variant*

The **variants** slot does not actually contain variants but rather contains codes that describe how the adverb is inflected (regular, irregular, invariant, or periphrastic). Details about these characteristics are provided in the technical report. See **DIMAP Adverb Processing** to describe the steps for characterizing adverb inflections.

Regular adverbs (such as *early*) form their comparative and superlative according to specific rules for adding the comparative suffix *er* and the superlative suffix *est* (using *ie* when ending in *y*). Regular adverbs are marked in Specialist with the line

variants=reg

In DIMAP, a regular adverb has a feature **infl** with a value **reg**.

Irregular adverbs do not follow regular patterns and the irregular forms are specifically entered. Irregular adverbs (for *well, far, less, thin, much,* and *bad*) are marked in Specialist with the line

variants=irreg|well|better|best|

In DIMAP, an irregular adverb has a feature **infl** with a value equal to the final two words, giving the comparative and superlative forms. In addition, an entry is created in the DIMAP variants dictionary for

each form. These entries have a feature **compOf** ("comparative of") or **suplOf** ("superlative of") with a value equal to the base form.

Invariant adverbs (such as *always*) have no morphological comparative or superlative form. Invariant adverbs are marked in Specialist with the line

variants=inv

In DIMAP, an invariant adverb has a feature **infl** with a value **inv**.

Periphrastic adverbs (such as *abnormally*) form their comparative and superlative with *more* and *most*. Periphrastic adverbs are marked in Specialist with the line

variants=inv;periph

In DIMAP, a periphrastic adverb has a feature **infl** with a value **inv;periph**.

Miscellaneous Adverb Properties

Specialist provides specific lines in an item entry for further description of adverb entries, including

- Modification Type (sentence, verb, particle, or intensifier, with sentence and verb modifiers subdivided into locative, temporal, or manner)
- Negative
- Broad Negative
- Interrogative
- Acronyms
- Abbreviations

Each adverb in Specialist has at least one slot identifying its type of modification: **sentence**, **verb**, **particle**, or **intensifier**. Adverb sentence and verb modifiers are further specified into **locative** (indicating direction or location), **temporal** (indicating time or duration), or **manner** (indicating the way an action was accomplished) types. Some adverbs may be used as a sentence modifier or a verb modifier.

Adverb sentence modifiers modify whole sentences. These are marked in Specialist with a property of the **modification_type** field (e.g., for *frankly*), i.e.,

modification_type=sentence_modifier;manner

In DIMAP, an adverb sentence modifier has a feature **mod** with a value equal to **sm(*modtype*)**, where *modtype* has a value **loc**, **man**, or **temp**.

Adverb verb modifiers modify the verb phrase. These are marked in Specialist with a property of the **modification_type** field (e.g., for *carefully*), i.e.,

modification_type=verb_modifier;manner

In DIMAP, an adverb verb modifier has a feature **mod** with a value equal to **vm(*modtype*)**, where

modtype has a value **loc**, **man**, or **temp**.

Adverb intensifiers modify adjectives or other adverbs. These are marked in Specialist with a property of the **modification_type** field (e.g., for *quite*), i.e.,

modification_type=intensifier

In DIMAP, an adverb verb modifier has a feature **mod** with a value equal to **intensifier**.

Adverb particles are associated with verbs. These are marked in Specialist with a property of the **modification_type** field (e.g., for *up*), i.e.,

modification_type=particle

In DIMAP, an adverb verb modifier has a feature **mod** with a value equal to **particle**.

Negative adverbs (such as *never*) produce true sentence negation. These are marked in Specialist with a bare feature **negative**, i.e.,

negative

In DIMAP, an adverb verb modifier has a feature **neg** with a value equal to **+** (plus sign).

Broadly negative adverbs (such as *rarely* and *seldom*) are not strictly sentence negators, but trigger certain syntactic phenomena associated with negation, such as polarity, positive question tags, and fronting with subject-auxiliary inversion (see the technical manual for more details). These are marked in Specialist with a bare feature **broad_negative**, i.e.,

broad_negative

In DIMAP, an adverb verb modifier has a feature **broad_neg** with a value equal to **+** (plus sign).

Interrogative adverbs (such as *how* or *when*) begin a sentence. These are marked in Specialist with a bare feature **interrogative**, i.e.,

interrogative

In DIMAP, an adverb verb modifier has a feature **interrog** with a value equal to **+** (plus sign).

Some adverbs are acronyms (*ASAP*) or abbreviations (*approx*). Abbreviations and Acronyms are handled in End of Entry Processing.

Pronouns

Pronouns are a closed class, with a small number of entries. Specialist characterizes them as the category pron. (See the technical manual for a complete characterization of the properties' codes. See DIMAP Pronoun Processing to describe the steps for characterizing the properties of pronoun entries.) Specialist provides specific lines in an pronoun entry for further description, including

- Person and Number
- Gender for Personal Pronouns

- Pronoun Type (Government, Possession, Reflexivity, Quantification, and Deixis)
- Interrogativity

Person and number are matters of agreemant or concord. These are marked in Specialist with a property of the **variants** field (e.g., for *she*, indicating the third person singular), i.e.,

variants=thr_sing

In DIMAP, a pronoun marked for person and number has a feature **agr** with a value equal to the code used in Specialist (see the technical manual for the full list of codes).

Gender allows restrictions on the referents or antecedents of pronouns in terms of humanness and sex (masculine, feminine, or neuter). These are marked in Specialist with a property of the **gender** field (e.g., for *she*, indicating that it is feminine), i.e.,

gender=pers(fem)

In DIMAP, a pronoun marked for gender has a feature **agr** with a value equal to the code used in Specialist (see the technical manual for the full list of codes).

Pronoun type covers a variety of phenomena. Government identifies the pronoun case. Possession identifies possessive pronouns. Reflexivity identifies reflexive pronouns. Quantification identifies universal qualifiers or indefinite pronouns (corresponding to *some*, *any*, or *no*, which are called non-assertive, assertive, or negative, respectively). Deixis identifies demonstrative pronouns. A pronoun may have several types. These are marked in Specialist with a property of the **type** field (e.g., for *none*, indicating that it is a negative indefinite pronoun), i.e.,

type=indef(neg)

In DIMAP, a pronoun marked for type has a feature corresponding to the type (**case**, **quant**, **poss**, **possnom**, **quant**, **refl**, or **dem**, respectively), with a value equal to the code used in Specialist for **case** and **quant** (see the technical manual for the full list of codes) or **+** (plus sign) for the other features.

Interrogative pronouns such as *what* or *which* are marked in Specialist with a property **interrogative**, i.e.,

interrogative

In DIMAP, a pronoun marked as interrogative has a feature **interrog**, with a value equal to **+** (plus sign).

Determiners

Determiners are a closed class, with a small number of entries. Specialist characterizes them as the category det. See DIMAP Determiner Processing to describe the steps for characterizing the properties of determiner entries, which provides specific lines in an item entry for further description, including

- Determiner agreement

- Deixis (demonstrative determiners)
- Interrogativity (interrogative determiners)

Determiner agreement properties identify the number characteristics of the nouns that they determine, e.g., *a* can only be used with a singular count noun. Agreement is marked in Specialist with a property of the **variants** field (e.g., for *many*, indicating that the determiner can only be used with plural count nouns), i.e.,

variants=plur

Specialist identifies six types of agreement: (1) determiners of singular nouns (**sing**, *each*), (2) determiners of count nouns (**plur**, *many*), (3) determiners of uncount nouns (**uncount**, *much*), (4) determiners of singular and uncount nouns (**singuncount**, *this*), (5) determiners of plural and ucount nouns (**pluruncount**, *more*), and (6) free determiners (**free**, *some*).

In DIMAP, determiners have a feature **agr** with a value equal to the code used in Specialist (see the technical manual for the full explanation of the codes).

The feature **demonstrative** marks the deictic determiners (*this*, *that*, *these*, and *those*). This is marked in Specialist with a single line (e.g., for *that*), i.e.,

demonstrative

In DIMAP, a determiner marked as demonstrative has a feature **dem**, with a value equal to **+** (plus sign).

The feature **interrogative** marks the interrogative determiners (*which*, *whatever*, *which*, and *whichever*). This is marked in Specialist with a single line (e.g., for *whatever*), i.e.,

interrogative

In DIMAP, a determiner marked as interrogative has a feature **interrog**, with a value equal to **+** (plus sign).

Verbs

Verb properties in the Specialist Lexicon pertain to their inflection, transitivity, complementation patterns, nominalizations, and whether the verb is an acronym or abbreviation. Verb inflections (i.e., their present, past, present participle, and past participle forms) are described in the discussion of **Verb Inflections**. Verb transitivity (whether a verb takes an object) and complementation or subcategorization patterns (types of phrases and clauses associated with a verb) are described in **Verb Transitivity and Complementation**. The remaining properties are described in the discussion of **Miscellaneous Verb Properties**. Details about these characteristics are provided in the technical report (pp. 8-9, 11-15, 32-51).

See **DIMAP Verb Processing** to describe the steps for characterize the properties of noun entries.

Verb Inflections

All information regarding the inflection of a verb is contained in the Specialist Lexicon in lines of the form:

variants=*variant*

The **variants** slot either contains a code (for regular inflections) or contains a list of the inflections for the different tenses. Details about these characteristics are provided in the technical report.

Regular verbs (such as *abandon*) form their third singular present, past, past participle, and present participle tenses according to a systematic set of rules (described in detail in the Technical Manual). Regular verbs are marked in Specialist with the line

variants=reg

In DIMAP, a regular verb has a feature **infl** with a value **reg**.

Regular doubling verbs (such as *abet*) end in a consonant that is doubled when forming their third singular present, past, past participle, and present participle tenses, but according to a systematic rule for orthographic consonant doubling (described in detail in the Technical Manual). Regular doubling verbs are marked in Specialist with the line

variants=regd

In DIMAP, a regular verb has a feature **infl** with a value **regd**.

Irregular verbs (such as *alight*) do not follow rules that can be systematically generalized and their tense forms are each listed in Specialist, e.g.,

variants=alight|alights|alit|alit|alighting|

Some verbs are nearly regular, as in the example, but each tense form is still listed in Specialist. Some verbs have more than one **variants** line when there are alternative forms for some tenses.

In DIMAP, an irregular verb has features **pres-t** (third singular present), **pt** (past), **pp** (past participle), and **pres-p** (present participle), with a value for the feature equal to the tensed form. In addition, an entry is created in the DIMAP variants dictionary for each tensed form. and "%%1 $presp %%2 %%6 presPartOf %%v $base\n" to VAR if the variable is a non-empty string. These entries have a feature **presOf** ("third singular present of"), **pastOf** ("past of"), **pastPartOf** ("past participle of"), or **presPartOf** ("present participle of") with a value equal to the base form.

Verb Transitivity and Complementation

Information regarding the transitivity and complementation or subcategorization patterns of a verb is contained in the Specialist Lexicon in lines beginning with:

intran
tran
ditran
link

cplxtran

These represent the five categories of verbs recognized in Specialist: intransitive, transitive, ditransitive, linking, and complex-transitive. Each of these lines has additional information that describes the complements (or verb phrase constituents) that typically accompany the verb. A verb may have many of these lines, describing the full variety of complementation patterns associated with a verb. While the full set for an individual verb cannot be considered exhaustive, they are quite comprehensive. For example, the verb *give* has 29 lines describing intransitive, transitive, ditransitive, and complex-transitive patterns, including

intran;part(in)
tran=np;part(back)
ditran=np,pphr(to,np);datmvt
cplxtran=np,infcomp:objr

As can seen in the sample lines, Specialist codes for the complementation patterns are quite complex and describe twelve basic linguistic complement types. See the Technical Manual for a full description, not only of the codes, but also of the underlying linguistic phenomena.

In DIMAP, the transitivity and complementation patterns are captured in features **type** (transitivity) and **compl** (complement pattern). The **type** feature has values **intran** (intransitive), **tran** (transitive), **ditran** (ditransitive), **cplxtran** (complex transitive), or **link** (linking). The **compl** feature has a value equal to the exact pattern listed in Specialist. Intansitive verbs with a particle have a feature **particle** with a value equal to the particle listed in the parentheses (e.g., *in* in the example above). In addition, any prepositonal phrase complements (**pphr**) are futher analyzed to extract the prepositions that are frequently collocated with the base form; the results of this analysis are printed in the file **comp.dmp**.

Miscellaneous Verb Properties

Specialist provides specific lines in an item entry for further description of verb entries, including

- Verb nominalizations
- Acronyms
- Abbreviations

Verbs which have nominalized forms have the slot **nominalization** filled by the base form (of the noun), the category (noun), and its entry identifier, e.g., *accumulate* has the line

nominalization=accumulation|noun|E0006765

In DIMAP, a verb that has a nominalization has a feature **nom** with a value equal to the base form of the noun.

Some verbs are acronyms (*KO* - keep out) or abbreviations (*cath* - catheterise). Abbreviations and Acronyms are handled in End of Entry Processing.

Modal and Auxiliary Verbs

Modal and auxiliary verbs differ from main verbs in having a much more elaborate set of inflections, as well as clitic (an unstressed word that normally occurs only in combination with another word, for example, *'m* in *I'm*) and negative contracted forms. These are represented in the Specialist Lexicon in **variant** lines in the entries for these verbs (specifically, "be", "do", "have", "may", "must", "ought", "shall", "will", "can", and "dare"). All information regarding these inflections is contained in the Specialist Lexicon in lines of the form:

variants=*variant*

The **variants** slot contains a wide variety of codes that describe case, number, tense, negative, and clitic forms. Details about these characteristics are provided in the technical report (pp. 16-19).

In DIMAP, a modal or auxiliary verb has features **pres-t** (third singular present), **pt** (past), **pp** (past participle), and **pres-p** (present participle), with a value for the feature that corresponds to the Specialist codes.

See DIMAP Modal and Auxiliary Processing to describe the steps for characteriz the properties of modal and auxiliary entries.

Prepositions and Conjunctions

Prepositions and conjunctions occur in the Specialist Lexicon as the line either:

cat=*prep* (217 entries)
cat=*conj* (67 entries)

There are a few entries that are characterized as acronyms or abbreviations for these categories; otherwise, there no other properties in the Specialist Lexicon for these entries. Further, there is no distinction between coordinating or subordinating conjunctions. The conjunctions are always characterized as coordinating conjunctions (**ccj**) in DIMAP since there is no basis for making this distinction.

See DIMAP Preposition and Conjunction to describe the steps for characterize the properties of preposition and conjunction entries.

Abbreviations and Acronyms

An entry in the Specialist Lexicon may be an acronym or an abbreviation for one or more other entries. This occurs in the Specialist Lexicon as a line under some base entry, e.g.,

acronym_of=*abdominal aortic aneurysmectomy|E0429482*
abbreviation_of=*abdomen|E0006443*

occur under the base forms *AAA* and *AB*, respectively. The Specialist Lexicon base entry is for the acronym or abbreviation, and contains the base form(s) of their expansions, along with the entry identifying number with a pipe "|" symbol. Acronyms and abbreviations are generally found in noun and adjective entries.

In the DIMAP entry for *AAA* or *AB*, DIMAP roles are used to record all the acronyms or abbreviations. For acronyms, the role name is **acrOf**; for abbreviations, the role name is **abbOf**. A role-link is created for acronym and abbreviation. The entry identification number included in Specialist (beginning with the E) is not captured. An entry may have several acronyms and/or be an abbreviation of several terms. Each of these is captured, so that DIMAP may contain several of these roles for a particular entry.

Complement Interpretations

Many nouns, verbs, and adjectives require that they have complements. In the description of the patterns for these parts of speech, there is an "interpretation" of the complement.

The type of complements are:

- **np**: noun phrase
- **np| |**: a specific noun inside the two vertical bars
- **pphr(,)**: a prepositional phrase, with a preposition before the comma and the object of the preposition after the comma
- **adj**: adjective
- **advbl**: adverbs and adverbial prepositional phrases
- **edcomp**: past-participial clause
- **infcomp**: infinitive clause, introduced by *to*
- **binfcomp**: "bare" infinitive clause
- **ingcomp**: present participial clause
- **fincomp( )**: finite clause complement (with eight variations of arguments (see pp. 47-49)
- **whfincomp**: Wh-finite complements (*whether, how, why, where, when, who, if*)
- **whinfcomp**: an infinitive clause with a *wh*-element
- **ascomp**: *As* absolute clauses, a predicate introduced by *as*

When specifying a particular part of speech, in a table describing a part of speech, the type of complement is identified following with the letters **interp**, indicating that the type of complement must be listed in **fincomp**, **whfincomp**, **whinfcomp**, or **ascomp**. These complement interpretations have codes, as follows

- **arbc: Arbitarty Control** indicates that the subject of the lower clause is not linguistically controlled.
- **nsc: Non-Subject Control** means that the subject of the matrix verb controls a missing non-subject NP in the present particle clause.
- **objc: Object Control** means that the direct object in the higher clause is logically both the object of the higher verb and the subject of the embedded (non-finite) clause.
- **objr: Object Raising** means that the direct object in the higher clause is logically the subjec to the non-finite clause and not the logical object of the higher clause.
- **subjc: Subject Control** means that the subject of the higher clause is also the logical subject of the embedded infinitival clause.
- **subjr: Subject Raising** indicates that the subject of the higher clause is the logical subject of the embedded infinitival clause.

The **Technical Report** (pp. 42-45) provides many examples of these complements in characterizing each of the parts of speech.

DIMAP Dictionary Maintenance Program

CL Research's Dictionary Maintenance Program (DIMAP) provides a generalized structure for dictionaries used in natural language processing. A full description of DIMAP can be seen at CL Research or in the help file that accompanies the full version of DIMAP or in the demonstration version available at CL Research. The demonstration version contains most of the basic functionality in DIMAP, including regular expression searches over all fields, subdictionary creation based on matches, uploading dictionary data, dictionary downloading to user specifications, automatic dictionary creation from texts, and integrated WordNet lookup

An entry in DIMAP consists of all forms that are equivalent in lowercase form. An entry contains one or more senses. Each sense contains fields for part of speech, labels, definitions, and other information. In creating entries for the Specialist Lexicon, the most-used fields are:

- Features
- Instances
- Roles

DIMAP Instances

Each sense of an entry in DIMAP may contain any number of **instances**. Each instance is generally viewed as a hyponym or subordinate of an entry. In converting SPECIALIST to DIMAP, an instance is created in the **variants dictionary (specvar)** for each multiword noun (phrase) that has an entry as the last word. For example, the entry *index* has nearly 200 instances, including such phrases as *ACH index*, *articulation index*, *leukopenic index*, and *stroke index*.

DIMAP Features

Each sense of an entry in DIMAP may contain any number of **features**. Each feature has a **name** and a **value**. A feature is displayed as the name, an equals sign, and the value.

DIMAP Roles

Each sense of an entry in DIMAP may contain any number of **roles**. Each **role** has at least one and possibly many **links**. A feature is displayed as the name, an equals sign, and the value. In converting SPECIALIST to DIMAP, roles are created for acronyms and abbreviations, when the entry is an acronym and/or abbreviation.

Conversion to DIMAP

In creating a DIMAP dictionary for the Specialist Lexicon, the entries are converted into a format that DIMAP uses for uploading entries, using two Perl scripts. Details of the format are provided in the help file that accompanies DIMAP. In this discussion, the focus is on what is done to an entry in the Specialist Lexicon and where that information will be found in a DIMAP entry. In general, the conversion handles each part of speech differently, so the presentation will provide details oriented for each part of speech

and how information associated with that part of speech is handled.

The major trigger for processing an entry in the Specialist Lexicon is the first line of an entry, which identifies the base form of a lexical item, with a line that begins with

{base=*word*

A DIMAP entry is created for each *word*. There may be several entries for a given word, i.e., with identical word forms, where the case of the word form is ignored. Thus, *cold* and *COLD* (an acronym) are treated as identical. In creating a DIMAP entry, the occurrence of multiple entries from the Specialist Lexicon gives rise to a distinct sense under each entry.

In addition to the forward-looking dictionary containing the main entries from the Specialist Lexicon, a DIMAP Variants dictionary (VAR) has also been created, to enable identification of base forms from several types of variants.

DIMAP Variants Dictionary

In the creation of the main DIMAP alphabetic Specialist Lexicon, a variants dictionary is also created (**specvar.dmp**, VAR). This dictionary contains entries for several types of phenomena that occur within the Specialist Lexicon. These include:

- Spelling Variants (linked to a base form)
- Noun Phrases (entries for the first and last words)
- Noun Variants (plurals linked to their base)
- Adjective Inflections (linked to their base)
- Adverb Inflections (linked to their base)
- Verb Inflections (linked to their base)

Conversion Scripts

The creation of files to be uploaded into DIMAP Specialist and Specialist Variants dictionaries is accomplished with Perl scripts:

- spec2dmp.pl (converts the UMLS SPECIALIST lexicon (**LEXICON**) to DIMAP upload format in the file **umls.dmp** (DMP) to create the dictionary **umls.dct**, along with Specialist variants, **specvar.dmp** (VAR) and a file of preposition complements **comp.dmp** (COMP))
- collect.pl (collects variants in the file **specvar.dmp** (VAR) to create another version that is uploaded to the dictionary **specvar.dct**)

spec2dmp.pl

Unit
spec2dmp.pl

Description
Converts the UMLS SPECIALIST lexicon to DIMAP upload format, creating the files **umls.dmp** (DMP),

specvar.dmp (VAR), and **comp.dmp** (COMP) from the UMLS file **LEXICON**.

Implementation
Opens the files for input (**LEXICON**) as SPEC, and output (**umls.dmp**, **specvar.dmp**, and **comp.dmp**), as DMP, VAR, and COMP. Intializes **$count** (used to count the number of entries) and **$errors** (used to count the number of errors) at 0.

Enters a while loop over the lines of SPEC, reporting progress every 1000 entries. The script envisions that the first line of the file contains the string "base=", which is the beginning of an entry. This initializes the processing of the entry data. The variable **$base** is set to the string following the match. If this begins with a character not a letter (not A-Z or a-z), '#' is prepended. A DIMAP entry line "%%1 **$base**\n" is printed to DMP. The variables **$pos** (part of speech), **$id** (entry id number) **$spvar** (spelling variants) and **$plurals** (plural forms) are initialized as empty strings and the arrays @**acrs** (acronyms) and @**abbs** (abbreviatons) are initialized.

Within the overall while loop, another while loop is entered. This loop continues until the line signaling the end of an entry (a right brace, **}**) is reached. The major steps in the loop are describing in **Processing an Entry's Contents**. When the end of the entry is reached, the final steps are described in **End of Entry Processing**.

DMP

A text file, **umls.dmp**, in a format suitable for upload to DIMAP, containing the conversion of Specialist item entries in **spec2dmp.pl**. This file contains a line identifying each entry word, followed by several indented lines containing information for the senses associated with the entry. The file may contain several entries that are identical in lowercase form (e.g., noun, verb, and acronym forms). When uploaded into DIMAP, each such entry will constitute a distinct sense.

The resultant **umls.dmp** file contains 2.8 million lines. These lines all begin a code using "%%". In the Specialist Lexicon, the headword **base** is converting to a **umls.dmp** line beginning with "%%1" to identify the entry for DIMAP (519,694 lines). Every other line is indented, as follows:

- %%2 identifies the part of speech (category) (519,694 lines)
- %%6 lines identify DIMAP features, the variety of properties as specified in **spec2dmp.pl** (1,739,557 lines)
- %%5 lines identify DIMAP roles, used only when the headword **base** is an abbreviation or acronym (49.218 lines)

VAR

A text file, **specvar.dmp**, in a format suitable for upload to a DIMAP variants dictionary after running the script **collect.pl**. This dictionary contains an entry for each variant form (not following regular rules of inflection) linked to its base form, as well as entries for the first and last words of multiword nouns. The variant forms include noun plurals, adjective inflections, adverb inflections, and verb inflections.

The resultant **specvar.dmp** file contains 716,120 lines. These lines all begin a code using "%%1" followed by one or more characters (possibly including spaces) until reaching the string "%%2". These strings identify the variant entry word(s) for the DIMAP Variants Dictionary. All other codes for the DIMAP entry are in the same line beginning with the headword. The other information in the line includes, as follows:

- %%2 identifies the part of speech (category) (716,120 occurrences, all of which contain for actual part of speech, which is recorded in DIMAP as having no part of speech)

- %%6 items specify 454,780 DIMAP features, identifying the feature names and the feature values, of which 261,345 have the name **kind** with a value indicating the end of multiword entry (e.g., the entry "antibiotic" has a kind "~ lock therapy"), 169,890 have the name **varOf** with a value indicating the base headword, and 23.545 specify inflected variations identifying the base form (e.g., "awoke" as the past of "awake")
- %%4 lines identify 261,345 DIMAP instances, where the entry headword has hyponyms (e.g., "medicine" has the hyponym, or instance, "aviation medicine")

COMP

A text file, **comp.dmp**, containing preposition complements associated with a verb, noun, or adjective entry. This text file is a tab-separated file, suitable for import into a spreadsheet. Each line consists of a preposition, the part of speech of the governing word ("nou" for noun, "vrb" for verb, and "adj" for adjective), the phrase type of the complement (almost always "np"), the base form of the entry that is the head word of the prepositional phrase, and if present, a particle associated with the base entry (usually a verb).

Processing an Entry's Contents

Unit
spec2dmp.pl

Description
The main part of the script, processing each line of an entry.

Implementation
The steps within the while loop for processing an entry's contents examine the current line for distinguishing features, with particular emphasis on processing lines according to the current entry's category or part of speech, including:

- Annotations (matching **annotation**, with no further processing, continuing to the next line; there are currently no such lines)
- End of Entry Processing (matching a right brace, **}**)
- Entry Identifier (matching **entry=**)
- Spelling Variants (matching **spelling_variant=**)
- Category (matching **cat=**)
- Noun Processing
- Adjective Processing
- Adverb Processing
- Pronoun Processing
- Determiner Processing
- Verb Processing
- Modal and Auxiliary Verb Processing
- Preposition and Conjunction Processing

In the description of each of these components, the Perl comment where processing begins is identified in parentheses.

End of Entry Processing

Unit
spec2dmp.pl (# end of entry processing)

Description
Completes the processing of an entry: (1) creating a DIMAP role-link line for DMP (with the role **abbOf** or **acrOf**) if the base is an abbreviation or an acronym (with the base as the link), (2) creating two DIMAP entry lines for VAR for nouns containing a space (one for the last word with the first part as a DIMAP instance and one for the first word with a DIMAP feature **kind** with the remainder as its value), and (3) a DIMAP feature **pl** with its irregular plurals as its value to DMP and a DIMAP entry for each plural with a DIMAP feature **plOf** and the base form as its value to VAR.

Implementation
When **$line** contains "}" (the terminating brace), end of entry processing is performed. Finishes up processing for an entry, with the following specific steps:

- If **@abbs** is not empty, prints a DIMAP role name "\t%%5 abbOf" to DMP and for each **$abb** of **@abbs**, prints a role link " %%k **$abb**" to DMP, printing an end of line after all abbreviations are printed.
- If **@acrs** is not empty, prints a DIMAP role name "\t%%5 acrOf" to DMP and for each **$acr** of **@acrs**, prints a role link " %%k **$acr**" to DMP, printing an end of line after all acronyms are printed.
- If **$base** contains a space, calls **&CreateHeadsAndKinds** with **$base** and **$pos** as arguments to print, for noun phrases (i.e., containing a space), two DIMAP entry lines to the variant file (VAR), one with the last word as an entry with the full entry as a DIMAP instance and one with the first word as an entry and the remainder of the phrase as a DIMAP feature **kind** with a value consisting of a tilde, a space, and the remainder of the phrase.
- If **$plurals** is not empty, prints a DIMAP feature "\t%%6 pl %%v **$plurals**\n" to DMP. Sets **$plurals0** to **$plurals** and splits it on semicolons. For each **$pl** in **@pls**, prints "%%1 **$pl** %%2 %%6 plOf %%v **$base**\n" to VAR, thus creating an entry for each plural.

Ends the while loop getting lines from SPEC so that preparations are made for the next entry.

Entry Identifier

Unit
spec2dmp.pl (# gets the Entry Unique Identifier (EUI))

Description
Recognizes "**entry=**" and sets **$id** to the string after the match.

Implementation
Occurs when "**entry=**" is matched. Sets **$id** equal to the string after the match and continues to the next line.

Spelling Variants

Unit
spec2dmp.pl (# captures **spelling variants**)

Description
Recognizes "**spelling_variant=**" and sets **$spvar** to the concatenation of the string after the match of all such lines in an entry.

Implementation
If **$line** matches "**spelling_variant=**", then if **$spvar** is not an empty string, appends a semicolon and a space. Appends whatever follows the match to **$spvar** and continues to the next line.

Category

Unit
spec2dmp.pl (# gets the part of speech of an entry)

Description
Prints to **DMP** a DIMAP sense line containing the part of speech (see **Category (Part of Speech)**), a DIMAP feature line **id** with the Entry Unique Identifier as its value (see **Entry Identifying Number**), and possibly a DIMAP feature line **var** with all spelling variants as its value. If an entry has spelling variants, a combination line is printed to **VAR**, consisting a DIMAP entry code with the variant as the entry, an empty part of speech code, and a DIMAP feature **varOf** with the base form as its value.

Implementation
Occurs when "cat=" is matched. Sets **$pos** to the string after the match. Resets **$pos** to the result of calling **&GetPos** with **$pos** as argument to obtain the DIMAP three-letter code for each part of speech. Prints to DMP the lines:

- "\t%%2 $pos\n" (a sense entry, identifying the part of speech)
- "\t%%6 id %%v **$id**\n" (a feature, identifying the Entry Unique Identifier)

If **$spvar** (spelling variants) is non-empty, prints to **DMP**

- "\t%%6 var %%v **$spvar**\n" (a list of spelling variants)

In addition, if **$spvar** is non-empty, each variant in **$spvar** gives rise to a line in **VAR**

- "%%1 **$var** %%2 %%6 varOf %%v **$base**\n" (an entry for the variant, with one sense having no part of speech and containing a feature "varOf" identifying the base form).

After this processing, continues to the next line of SPEC.

Noun Processing

Unit
spec2dmp.pl (# noun processing)

Description
Handles lines that appear in noun entries, specifically, noun variants, and/or miscellaneous noun forms, including proper nouns, noun complements, nominalizations, acronyms, abbreviations, and trademarks, primarily writing DIMAP feature lines to DMP.

Implementation
Handles entry lines when **$pos** equal "nou", specifically:

- Noun Variants (matching "**variants\=**"), matching several specific inflection types: (1) "**groupcount**" (prints "\t%%6 groupcount %%v +/-\n" to DMP); (2) "**uncount**" (prints "\t%%6 count %%v -\n" to DMP); (3) "**glreg**" (prints "\t%%6 infl %%v gl\n" to DMP, sets **$plural** to the result of calling **&GrecoLatinPlural** with **$base** as argument obtain the plural of the argument, which is the singular form, following Greco-Latin inflection rules, and if this is non-empty, adds it to **$plurals**, appending "; " if **$plurals** already has a non-empty value; (4) "**metareg**" (prints "\t%%6 infl %%v mr\n" to DMP); (5) "**sing**" (prints "\t%%6 sing %%v +\n" to DMP); (6) "**plur**" (prints "\t%%6 plur %%v +\n" to DMP); (7) "**inv**" (prints "\t%%6 inv %%v +\n" to DMP); (8) "**irreg\|**" (sets **$plform** to the remainder after the match, matches "**\|**" to remove the base form, resets **$plform** to the remainder, removes any further "**\|**", and concatenates **$plform** to **$plurals**); or (9) "**reg**" (prints "\t%%6 reg %%v +\n" to DMP). (If **$plurals** is non-empty, it is handled in End of Entry Processing.) If a line with "**variants\=**" is encountered but not processed, an error message is written.
- Proper Nouns (matching "**proper**"): prints "\t%%6 proper %%v +\n" to DMP
- Noun Complement Patterns (matching "**compl**"): Sets **$complement** to the remainder, prints "\t%%6 compl %%v **$complement**\n" to DMP, and calls **AnalyzeComp** with **$pos** and **$complement** as arguments to extract preposition complements to the file **comp.dmp** (COMP).
- Nominalizations (matching "**nominalization_of=**"): sets **$nomOf** to the remainder, resets it to whatever precedes "**\|**", and prints "\t%%6 nomOf %%v $nomOf\n" to DMP
- Acronyms (matching "**acronym_of=**"): sets **$acr** to the remainder, resets it to whatever precedes "**\|**", and pushes **$acr** onto @**acrs**. (If @**acrs** is non-empty, it is handled in End of Entry Processing.)
- Abbreviations (matching "**abbreviation_of=**"): sets **$abb** to the remainder, resets it to whatever precedes "**\|**", and pushes **$abb** onto @**abbs**. (If @**abbs** is non-empty, it is handled in End of Entry Processing.)
- Trademarks (matching "**trademark**"): prints "\t%%6 tm %%v **$tm**\n" to DMP, where **$tm** may be '+' if no generic form is identified.

If a line under "nou" is not handled by one of the preceding options, an error message is written.

Adjective Processing

Unit
spec2dmp.pl (# adjective processing)

Description
Handles lines that appear in adjective entries, specifically, adjective inflection (for comparison), stative adjectives, attributive adjective position, other adjective positions, adjective complements, adjective nominalizations, acronyms, and abbreviations, primarily writing DIMAP feature lines to DMP, with non-standard inflections to VAR and complements to COMP.

Implementation
Handles entry lines when **$pos** equal "adj", specifically:

- Stative Adjectives (matching "**stative**"): prints "\t%%6 stative %%v +\n" to DMP
- Adjective Variants (matching "**variants\=**"), matching several specific types: (1) "**inv:periph**" (prints "\t%%6 infl %%v inv;periph\n" to DMP); (2) "**inv**" (prints "\t%%6 infl %%v inv\n" to DMP); (3) "**irreg**" (splits on "|" to obtain the comparative and superlative forms in **$infs**, prints "\t%%6 infl %%v **$infs[1]**|**$infs[2]**" to DMP, and prints "%%1 **$infs[1]** %%2 %%6 compOf %%v **$base**\n" and "%%1 "%%1 **$infs[2]** %%2 %%6 suplOf %%v **$base**\n" to VAR), (4) "**regd**" (prints "\t%%6 infl %%v regd\n" to DMP); and (5) "**reg**" (prints "\t%%6 infl %%v reg\n" to DMP).
- Attributive Position (matching "**position=attrib\((.*?)\)**"): prints "\t%%6 attrib %%v $1\n" to DMP

- Other Adjective Position (matching "**position=**"): sets **$position** to the value after the match and prints "\t%%6 position %%v **$position**\n to DMP (with expected values of *pred*, *post*, and *attribc*)
- Adjective Complements (matching "**compl=**"): sets **$compl** to the string after the match, prints "\t%%6 compl %%v **$compl**\n" to DMP, and calls **AnalyzeComp** with **$pos** and **$compl** to extract preposition complements to the file **comp.dmp** (COMP)
- Adjective Nominalizations (matching "**nominalization=**"): extracts the nominalization (**$nom**) at the beginning of the slot and prints "\t%%6 nom %%v **$nom**\n" to DMP
- Acronyms (matching "**acronym_of=**"): sets **$acr** to the remainder, resets it to whatever precedes "**\|**", and pushes **$acr** onto @**acrs**. (If @**acrs** is non-empty, it is handled in End of Entry Processing.)
- Abbreviations (matching "**abbreviation_of=**"): sets **$abb** to the remainder, resets it to whatever precedes "**\|**", and pushes **$abb** onto @**abbs**. (If @**abbs** is non-empty, it is handled in End of Entry Processing.)

In each of the above, processing continues to the next line of an adjective entry after handling a specific line. If processing does not continue, prints a line "Unknown adjective line $line under $base\n" to standard output.

Adverb Processing

Unit
spec2dmp.pl (# adverb processing)

Description
Handles lines that appear in adverb entries, specifically, adverb inflection (for comparison), adverb modification, features identifying negative, broad negative, and interrogative adverbs, acronyms, and abbreviations, primarily writing DIMAP feature lines to DMP, with non-standard inflections to VAR.

Implementation
Handles entry lines when **$pos** equal "adv", specifically:

- Adverb Variants (matching "**variants\=**"), matching several specific types: (1) "**inv:periph**" (prints "\t%%6 infl %%v inv;periph\n" to DMP); (2) "**inv**" (prints "\t%%6 infl %%v inv\n" to DMP); (3) "**irreg**" (splits on "|" to obtain the comparative and superlative forms in **$infs**, prints "\t%%6 infl %%v **$infs[1]**|**$infs[2]**" to DMP, and prints "%%1 **$infs[1]** %%2 %%6 compOf %%v **$base**\n" and "%%1 "%%1 **$infs[2]** %%2 %%6 suplOf %%v **$base**\n" to VAR), and (4) "**reg**" (prints "\t%%6 infl %%v reg\n" to DMP).
- Adverb Modification Types (matching "**modification_type=**"), matching several specific types: (1) verb modifiers (matching "**verb_modifier;**"), with subtypes (a) verb modifiers (matching "**verb_modifier;**") of subtypes locative, manner, and temporal (matching "**locative**", "**manner**", and "**temporal**"), printing "\t%%6 mod %%v vm(loc)\n", "\t%%6 mod %%v vm(man)\n", and "\t%%6 mod %%v vm(temp)\n" to DMP, (b) verb modifiers (matching "**sentence_modifier;**") of subtypes locative, manner, and temporal (matching "**locative**", "**manner**", and "**temporal**"), printing "\t%%6 mod %%v sm(loc)\n", "\t%%6 mod %%v sm(man)\n", and "\t%%6 mod %%v sm(temp)\n" to DMP, (c) particle (matching "**particle;**"), printing "\t%%6 mod %%v particle\n", and (d) intensifier (matching "**intensifier;**"), printing "\t%%6 mod %%v intensifier\n"
- Broad Negative (matching "**broad_negative**"), prints "\t%%6 broad_neg %%v +\n" to DMP
- Interrogative (matching "**interrogative**"), prints "\t%%6 interrog %%v +\n" to DMP
- Negative Adverbs (matching "**negative**"), prints "\t%%6 neg %%v +\n" to DMP
- Acronyms (matching "**acronym_of=**", such as *ASAP*): sets **$acr** to the remainder, resets it to whatever precedes "**\|**", and pushes **$acr** onto @**acrs**. (If @**acrs** is non-empty, it is handled in End of Entry Processing.)

- Abbreviations (matching "**abbreviation_of=**", such as *approx*): sets **$abb** to the remainder, resets it to whatever precedes "**\|**", and pushes **$abb** onto **@abbs**. (If **@abbs** is non-empty, it is handled in End of Entry Processing.)

Pronoun Processing

Unit
spec2dmp.pl (# pronoun processing)

Description
Handles lines that appear in pronoun entries, specifically, properties for person and number agreement, gender, government, reflexivity, quantification, possession, and interrogativity, writing DIMAP feature lines to DMP.

Implementation
Handles entry lines when **$pos** equal "pro" (when **cat** is **pron**), specifically:

- Pronoun Variants (matching "**variants\=**"), capturing person and number agreement, where the filler encodes first, second, and third person and singular and plural, printing "\t%%6 agr %%v **$agr**\n" to DMP, where **$agr** is the filler.
- Pronoun Gender (matching "**gender=**"), recording referents/antecedents in terms of humanness and sex, printing "\t%%6 agr %%v **$gen**\n" to DMP, where **$gen** is the filler.
- Pronoun Types (matching "**type=**"), setting **$type** to the filler and matching several specific types: (1) concerned with government ("subj" or "obj"), printing "\t%%6 case %%v **$type**\n" to DMP; (2) concerned with quantification ("univ" or containing "indef"), printing "\t%%6 quant %%v **$type**\n" to DMP; (3) concerned with reflexivity ("refl"), printing "\t%%6 refl %%v **+**\n" to DMP; (4) concerned with deixis ("dem"), printing "\t%%6 dem %%v **+**\n" to DMP; or (5) concerned with possession ("poss" or "possnom"), printing "\t%%6 poss %%v **+**\n" or "\t%%6 possnom %%v **+**\n" to DMP.
- Interrogative Pronouns (matching "**interrogative**"), prints "\t%%6 interrog %%v +\n" to DMP

Determiner Processing

Unit
spec2dmp.pl (# determiner processing)

Description
Handles lines that appear in determiner entries (see **Determiners**), specifically, properties for agreement, deixis, and interrogativity, writing DIMAP feature lines to DMP.

Implementation
Handles entry lines when **$pos** equal "pos", specifically:

- Determiner Agreement (matching "**variants\=**"), giving the number characteristics of the nouns they determine, printing "\t%%6 agr %%v **$numcnt**\n" to DMP, where **$numcnt** is the filler.
- Demonstrative Determiners (matching "**demonstrative**"), printing "\t%%6 dem %%v **+**\n" to DMP
- Interrogative Determiners (matching "**interrogative**"), printing "\t%%6 interrog %%v **+**\n" to DMP

Verb Processing

Unit
spec2dmp.pl (# verb processing)

Description
Handles lines that appear in verb entries, specifically, identifying regular and regular doubling inflection, irregular inflections, transitive, intransitive, and linking verbs with their complements, verb nominalizations, acronyms, and abbreviations, primarily writing DIMAP feature lines to DMP, with non-standard inflections to VAR and complements to COMP. (Note that the ordering of the processing avoids ambiguity.)

Implementation
Handles entry lines when **$pos** equals "vrb" (**cat=verb**), specifically:

- Regular doubling inflection verbs (matching "**variants=regd**"), printing "\t%%6 infl %%v regd\n" to DMP;
- Linking verbs (matching "**link=**"), setting **$complement** to the filler, prints "\t%%6 type %%v link\n" and "\t%%6 compl %%v **$complement**\n" to DMP, and calls **AnalyzeComp** with **$pos** and **$complement** as arguments to extract preposition complements (containing a "pphr") to the file **comp.dmp**
- Regular inflection verbs (matching "**variants=reg**"), prints "\t%%6 infl %%v reg\n" to DMP
- Transitive verbs (matching "**tran=**"), matching three specific subtypes: (1) complex transitive verbs (matching "**cplxtran=**"); (2) ditransitive verbs (matching "**ditran=**"); or (3) other transitive verbs (matching "**tran=**"), in each case setting **$complement** to the filler, prints "\t%%6 type %%v {cplxtran|ditran|tran}\n" and "\t%%6 compl %%v **$complement**\n" to DMP, and calls **AnalyzeComp** with **$pos** and **$complement** as arguments to extract any preposition complements to the file **comp.dmp**
- Intransitive verbs (matching "**intran**"), and then further matching "**;**" (a semicolon) to capture any particle, printing "\t%%6 type %%v intran\n" and if there is a particle, "\t%%6 particle %%v $'\n", to DMP
- Verb nominalizations (matching "**nominalization=**"), setting **$nom** to the nominalization and printing "\t%%6 nom %%v **$nom**\n" to DMP;
- Irregular inflection verbs (matching "**variants=irreg\|**"), setting **$vars** to what follows the match and then replacing bar in **$vars** with "#" and splitting **$vars** on this into **$bform** (the base form), **$prest** (the present tense), **$pt** (the past tense), **$pp** (the past participal), and **$presp** (the present participal). Prints "\t%%6 pres-t %%v **$prest**\n", "\t%%6 pt %%v **$pt**\n", "\t%%6 pp %%v **$pp**\n", and "\t%%6 pres-p %%v **$presp**\n" to DMP (the tense forms) and prints "%%1 $prest %%2 %%6 presOf %%v $base\n", "%%1 $pt %%2 %%6 pastOf %%v $base\n", "%%1 $pp %%2 %%6 pastPartOf %%v $base\n", and "%%1 $presp %%2 %%6 presPartOf %%v $base\n" to VAR to create entries for the tense variant forms to VAR
- Abbreviations (matching "**abbreviation_of=**"): sets **$abb** to the remainder, resets it to whatever precedes "**\|**", and pushes **$abb** onto **@abbs**. (If **@abbs** is non-empty, it is handled in End of Entry Processing.)
- Acronyms (matching "**acronym_of=**"): sets **$acr** to the remainder, resets it to whatever precedes "**\|**", and pushes **$acr** onto **@acrs**. (If **@acrs** is non-empty, it is handled in End of Entry Processing.)

If a line under "vrb" is not handled by one of the preceding options, an error message is written.

Modal and Auxiliary Processing

Unit
spec2dmp.pl (# modal and auxiliary processing)

Description

Handles **variant** lines that appear in the entries for modal and auxiliary verbs ("be", "do", "have", "may", "must", "ought", "shall", "will", "can", "need", and "dare"), capturing inflectional forms specific to these verbs, writing DIMAP feature lines to DMP.

Implementation
Handles entry lines when **$base** equals specific modal and auxiliary words and a line matches "**variant=**", with processing grouped into three sets, depending on the base form:

- when **$base** equals "be", "do", "have", "may", "must", "ought", "shall", "will": sets **$varform** to what follows **variant=** and then sets **$form** to what precedes a semicolon and **$tense** to what follows the semicolon; if **$tense** equals "infinitive", continues without further processing (since this is the main entry form); deals with several cases based on the value of **$tense**: (1) if **$tense** matches "pres(", sets **$casenum** to what follows this match and **$neg** to an empty string, if **$casenum** matches "negative", sets **$neg** to "neg", if **$casenum** matches "thr_sing", "fst_plur,second,thr_plur", or "fst_sing", sets **$casenum** to "3s", "1p,2,3p", or "1s", respectively; prints "\t%%6 pres-t %%v **$form** {**$casenum**" to DMP; if **$neg** is non-empty, prints ":**$neg**" to DMP; and finally prints "}\n" to DMP. (2) if **$tense** matches "pres", sets **$neg** to what follows this match and if **$neg** matches "negative", sets it to "neg"; prints "\t%%6 pres-t %%v **$form**" to DMP; IF **$neg** is non-empty, prints " {$neg}" to DMP; finally, prints "\n" to DMP; (3) if **$tense** matches "past\(", sets **$casenum** to what follows this match and **$neg** to an empty string, if **$casenum** matches "negative", sets **$neg** to "neg", if **$casenum** matches "thr_sing,fst_sing" or "fst_plur,second,thr_plur", sets **$casenum** to "1s,3s" or "1p,2,3p", respectively; prints "\t%%6 pt %%v **$form** {**$casenum**" to DMP; if **$neg** is non-empty, prints ":**$neg**" to DMP; and finally prints "}\n" to DMP. (4) if **$tense** matches "past",sets **$neg** to what follows this match and if **$neg** matches "negative", sets it to "neg"; prints "\t%%6 pt %%v **$form**" to DMP; IF **$neg** is non-empty, prints " {$neg}" to DMP; finally, prints "\n" to DMP; (5) if **$tense** matches "past_part", prints "\t%%6 pp %%v $form\n" to DMP; (6) if **$tense** matches "pres_part", prints "\t%%6 pres-p %%v $form\n" to DMP
- when **$base** equals "can": sets **$varform** to what follows **variant=** and then sets **$form** to what precedes a semicolon and **$tense** to what follows the semicolon; if **$tense** matches "past" or "pres", sets **$neg** to what follows this match and if **$neg** matches "negative", sets **$neg** to "neg"; prints "\t%%6 pt %%v **$form**" or "\t%%6 pres-t %%v **$form**" to DMP; if **$neg** is non-empty, prints " (**$neg**)" to DMP, and finally prints "\n" to DMP.
- when **$base** equals "dare": sets **$varform** to what follows **variant=** and then sets **$form** to what precedes a semicolon and **$tense** to what follows the semicolon; if **$tense** matches "pres", sets **$neg** to what follows this match and if **$neg** matches "negative", sets **$neg** to "neg"; prints "\t%%6 pres-t %%v $form" to DMP; if **$neg** is non-empty, prints " (**$neg**)" to DMP, and finally prints "\n" to DMP.
- when **$base** equals "need": sets **$varform** to what follows **variant=** and then sets **$form** to what precedes a semicolon and **$tense** to what follows the semicolon; if **$tense** matches "pres", sets **$neg** to what follows this match and if **$neg** matches "negative", sets **$neg** to "neg"; prints "\t%%6 pres-t %%v $form" to DMP; if **$neg** is non-empty, prints " (**$neg**)" to DMP, and finally prints "\n" to DMP.

Preposition and Conjunction Processing

Unit
spec2dmp.pl (# preposition processing, # conjunction processing)

Description
Handles lines in preposition and coordinating conjunction entries (see **Category processing**), a few of which are acronyms or abbreviations.

Implementation
Handles entry lines when **$pos** equal "prp" (**cat=prep**) or "ccj" (**cat=conj**) and also any entries that have specifically:

- Acronyms (matching "**acronym_of=**"): sets **$acr** to the remainder, resets it to whatever precedes "**\|**", and pushes **$acr** onto **@acrs**. (If **@acrs** is non-empty, it is handled in End of Entry Processing.)
- Abbreviations (matching "**abbreviation_of=**"): sets **$abb** to the remainder, resets it to whatever precedes "**\|**", and pushes **$abb** onto **@abbs**. (If **@abbs** is non-empty, it is handled in End of Entry Processing.)

GetPos

Unit
spec2dmp.pl

Description
Returns the DIMAP three-letter code for each part of speech when getting the part of speech (**cat=**) for an entry

Implementation
Changes the argument **$pos** with the 3-letter code used in converting these entries into DIMAP, as follows::

- noun (nou): common and proper nouns
- verb (vrb): verbs
- adj (adj): adjectives
- adv (adv): adverbs
- det (det): determiners (e.g., *a* and *the*)
- prep (prp): prepositions
- pron (pro): pronouns
- conj (ccj): conjunctions
- modal (aux): modal verbs (e.g., *might* and *would*)
- aux (aux): auxiliary verbs (e.g., *be*)
- compl (rel): complementizers (e.g., *that*)

If none of these is matched, sets **$pos** to "nil" and prints a message that there is an unrecognized part of speech.

GrecoLatinPlural

GrecoLatinPlural($base);
Unit
spec2dmp.pl

Description
Creates the plural of the argument, which is the singular form, following Greco-Latin inflection rules, when an entry has the property **glreg**.

Implementation
Sets **$sing** to the argument (the base of an entry) and **$plural** to an empty string. Matches the end of **$sing** (the final characters of the base entry), with the order of the tests being important, and replaces that end with a different string to create **$plural**, which is returned.

- "us" with "i"

- "ma" with "mata"
- "a" with "ae"
- "um" with "a"
- "on" with "a"
- "sis" with "ses"
- "is" with "ides"
- "men" with "mina"
- "ex" with "ices"
- "x" with "ces"

If none of these end characters was match, the returned value of **$plural** is an empty string.

CreateHeadsAndKinds

CreateHeadsAndKinds($base,$pos)

Unit
spec2dmp.pl (called from End of Entry Processing)

Description
For noun phrases (i.e., containing a space), prints two DIMAP entry lines to the variant file (VAR), one with the last word as an entry with the full entry as a DIMAP instance and one with the first word as an entry and the remainder of the phrase as a DIMAP feature **kind** with a value consisting of a tilde, a space, and the remainder of the phrase.

Implementation
Sets **$mybase** and **$mypos** to the arguments. Performs further processing only if **$mypos** is equal to "nou". Passes over all intermediate spaces, to reach the final one, setting **$head** to the last word. Prints "%%1 **$head** %%2 %%4 **$mybase**\n" to VAR. Matches a space in **$mybase**, setting **$first** to the first word and **$last** to the remainder. Sets **$eqn** to the concatenation of "~ " and **$last** (with the tilde corresponding to the base entry). Prints "%%1 $first %%2 %%6 kind %%v $eqn\n" to VAR.

AnalyzeComp

AnalyzeComp($pos,$comp)

Unit
spec2dmp.pl

Description
Extracts preposition complements to the file **comp.dmp** (COMP), a text file containing preposition complements to verbs, nouns, and adjectives. This text file is a tab-separated file, with each line consisting of a preposition, the part of speech of the governing word ("nou" for noun, "vrb" for verb, and "adj" for adjective), the phrase type of the preposition complement (almost always "np"), the base form of the entry where the complement appears, and if present, a particle associated with the base entry (usually a verb). Also, counts the number of entries with complements and the number that have preposition complements that are printed to the file. Preposition complements that are part of idiomatic phrases are not included as "general" complements.

Implementation
Sets **$mypos** and **$mycomp** to the arguments and declares empty strings at **$p2** and **$c2** (to hold secondary preposition complements). Increments **$compsAll** (counting the number of complements recognized in processing the file).

Matches **$mycomp** to "pphr\((.*?)\,(.*?)\)$", setting **$p** (the preposition) to **$1** and **$c** (the phrase type of the complement) to **$2** if successful, to identify preposition complements. If **$c** matches "\)\,pphr\(" (indicating the presence of a second preposition complement), sets **$c** to what precedes the match (to capture the first preposition) and **$second** to what follows the match. If **$second** matches a comma, sets **$p2** to what precedes the comma (to capture the second preposition) and **$c2** to what follows the comma (the phrase type of the second complement). Sets **$particle** to an empty string. If **$c** matches "np\);part\((.*?)$" (indicating that the base also has a particle), sets **$c** to "np" and **$particle** to **$1**.

If **$c** does not contain "\|" (which otherwise would indicate that the complement is actually part of an idiomatic expression or phrase, such as *take account of* or *cirrhosis of liver*, where *account* or *liver* are parts of phrases, i.e., not a general complement) and **$base** is not empty, prints "**$p**\t**$mypos**\t**$c**\t**$base**" to COMP (the file **comp.dmp**). If **$particle** is not empty, prints "\t**$particle**" to COMP. Prints a line return to COMP. Increments **$compsPreps**. If **$p2** and **$c2** are non-empty and **$c2** does not contain "\|", prints "**$p2**\t**$mypos**\t**$c2**\t**$base**\n" to COMP (the second complement is not distinguished from the first).

collect.pl

Unit
collect.pl

Description
This script collects variant information generated in various places in **spec2dmp.pl** that should be entered under a single entry. This includes:

- DIMAP features: **plOf** (plural of), **varOf** (variant of), **compOf** (comparative form of), **suplOf** (superlative form of), **presOf** (3rd person singular present of), **pastOf** (past tense of), **presPartOf** (present participle of), **pastPartOf** (past participle of), and **kind** (the second and subsequent words in a multiword noun entry, entered under the first word)
- DIMAP instances: the head noun of a multiword noun entry (i.e., entered under the last word of such an entry)

To collect "variants" data generated in converting the UMLS SPECIALIST lexicon to DIMAP, several steps in the command prompt are required before using **collect.pl**, as follows:

- Running **spec2dmp.pl** produces the initial file **specvar.dmp**
- Rename **specvar.dmp** to **specvar0.dmp**
- Sort **specvar0.dmp** into **specvar1.dmp** ("sort specvar0.dmp > specvar1.dmp")
- Open **specvar1.dmp** and remove initial lines beginning with non-ASCII characters and digits. (The last such line in 2020 begins with "%%1 9q %%2 %%4 trisomy 9q**".**)
- Run **collect.pl** with **specvar1.dmp** as argument to create the final **specvar.dmp** ("collect.pl specvar1.dmp > specvar.dmp").
- Upload **specvar.dmp** to the variant dictionary **specvar** in DIMAP

Implemenation
Initializes **$oldhw** and **$oldline** to empty strings. Reads each **$line** with the following steps:

- If **$line** is the same as **$oldine**, goes to the next line; otherwise, setting **$oldline** to **$line**
- Sets **$newhw** to the string after "%%1 " and before " %%2" (getting the lowercased headword).

- If **$newhw** is not equal to **$oldhw**, (checking for lines with "%%5 " and either "rootOf" or "derivs", neither of which will now do not occur), prints **$newline** (to the command prompt or to **specvar.dmp**), and then sets **$oldhw** to **$newhw** and **$newline** to **$line** (essentially getting ready to process a new headword)
- Else If **$newhw** is equal to **$oldhw**, matches " %%2", appending what occurs after this match to the current **$newline** (the string after the match is either a DIMAP feature or a DIMAP instance, thus collecting all the features or instances that will be concatenated to the appropriate headword characterization)

(The subroutines **combineRootOf** and **combineDerivs** are no longer applicable.)