MetaMap Migration to Lexical Tools Java APIs – Inflectional Issues

I. Fundamental issues (differences):

The inflection function is used to get categories, inflections, base forms, spelling variants, citation forms, etc. in Lexical tool. There are three fundamental issues between "C-Code" and Java version that results in difference in test_bff.log, test_vf.log, and test_vcf.log. They are describes as below:

Issue 1. The facts in Java version is a complete version of Lexicon

Inflections/uninflections are generated by facts (known to Lexicon) and rules (if not known in Lexicon) in the Lexical Tools. The fact in C-code is a very primitive version; it only includes irreg inflections in C version Lexical Tool. C version Lexical Tool uses suffix rules for inflections and uninflections except for irreg. A new feature (requested by the designed requirements of Lexical tools) was implemented in the Java version to load all known inflectional variables (in LEXICON) to the Database as facts. All facts are stored in database for retrieving to achieve faster performance and more accurate results. For the 2010 release, there are 1,488,144 inflectional variants known as facts in Java (while only less than 10% facts in C). This difference in inflectional facts results in difference in retrieving base forms, spelling variants, and inflectional variants.

Issue 2. Should inflectional variants be case sensitive (of the input)?

Usually, case does not contribute meaning in terms. The inflectional variants are designed to be aggressive to ignore cases (case insensitive) when they are known by LEXICON (fact). In other words, the output should be the same for the same inputs with difference cases. For example, the set of inflectional variants should be the same for inputs of "see", "SEE", and "See". Accordingly, the lowercase forms of inflectional variants are used for indexing in the database table. On the other hand, inflectional variants keep the case information when they are generated by suffix rules to have better guesses (precision). Inflectional variants might have different results depends on they are generated by facts or rules. C-code uses suffix rules for inflections/uninflections results in case sensitive (except for irreg and those know in C facts). C-code does have some algorithm to convert the outputs to make them case-insensitive. However, the results are not consistent and bug prone to result in missing some records as discussed below.

Issue 3. The ASCII conversion issue

Please see ASCII issues report.

II. Difference in of test_bff.log:

The difference of above three issues results in different outputs in test_bff.log. test_bff.log is to log of testing the difference in retrieving the citation forms and spelling variants (with base inflection) of input terms. ASCII issue will be discussed in a separated report and won't be addressed here. Let's go through problems caused by the first two issues from above. C-code finds less variants (than in Java version) in test_bff.log.

Type I: C-code does not find some spelling variants in test_bff.log

C-code does not find some spelling variants (IMHO, it's a bug). Such as for terms of "AAAS", "AADS", "BADS", etc..

Example walks through in ("AAAS", where C-code does not find "A.A.A."):

- Step 1: AAAS is uninflected and found two uninflected (base) forms
  shell> lvg –f:b
  AAAS
  AAAS|**AAA|128|1**|b|1|
  AAAS|**AAAS|128|1**|b|1|

- Step 2: find all spelling variants (with inflection is base) for these two base forms
  shell> lvg –f:s –cf:2 –if:3
  AAA|128|1
  AAA|128|1|A.A.A.|128|1|s|1|
  AAA|128|1|AAA|128|1|s|1|
  AAA|128|1|AAA|128|1|s|1|

  AAAS|128|1
  AAAS|128|1|AAAS|128|1|s|1|

  Three spelling variants are found: “A.A.A.”, “AAA”, and “AAAS”. As shown below, there are three lexical records related to these two base forms. “**A.A.A.” is a spelling variant of “AAA” from 2nd record and C-code does not find it.**

  {base=AAA
  entry=E0000049
  cat=noun
  variants=metareg
  variants=uncount
  acronym_of=abdominal aortic aneurysmectomy|E0429482
  acronym_of=acne-associated arthritis|E0429483
  acronym_of=acquired aplastic anemia|E0429484
  acronym_of=acute anxiety attack|E0429485
  acronym_of=androgenic anabolic agent|E0429486
  acronym_of=aneurysm of ascending aorta
  acronym_of=aromatic amino acid|E0356310
  acronym_of=acute apical abscess|E0356309
  acronym_of=abdominal aortic aneurysm|E0006446
  acronym_of=alpha-aminoadipic acid|E0598118
  acronym_of=anti-albumin antibody|E0588272
  acronym_of=acetylaminoantipyrine|E0561250
  acronym_of=anti-actin antibody|E0598119
  acronym_of=Aleuria aurantia agglutinin|E0598120
  acronym_of=Anguilla anguilla agglutinin|E0598123
  acronym_of=amino acid analysis|E0598121
  acronym_of=aryl acylamidase|E0598122
  }
  {base=A.A.A.
  spelling_variant=AAA
  entry=E0000546

cat=noun
variants=sing
variants=groupuncount
acronym_of=American Academy of Allergy|E0000250
acronym_of=American Association of Anatomists|E0000265
acronym_of=Area Agency on Aging|E0429481
}
{base=AAAS
entry=E0598131
cat=noun
variants=groupuncount
acronym_of=American Association for the Advancement of Science|E0000264
}

Type II: C-code does not find some records in test_bff.log due to it's case sensitive algorithm:

Inflections should be case insensitive. For example, the inflectional results of inputs of "bacteria", "BACTERIA", "Bacteria", or "BacTeria" should be the same. Java version generates more aggressive spelling variants of inflections (uninflections) when they are multiple records for the same term in difference cases. This results in difference cases in test_bff.log, such as terms of "bacteria", "boss", "dive", "letters", "this", etc..

Example walks through in bff ("bacteria", where C-code does not find "Bacterium"):

- Step 1: Bacteria is uninflected and found two uninflected (base) forms
  shell> lvg –f:b
  bacteria
  bacteria|**bacteria|128|1**|b|1|
  bacteria|**bacterium|128|1**|b|1|

- Step 2: find all spelling variants (with inflection is base) for these two base forms
  shell> lvg –f:s –cf:2 –if:3
  bacteria|128|1
  bacteria|128|1|bacteria|128|1|s|1|

  bacterium|128|1
  bacterium|128|1|Bacterium|128|1|s|1|
  bacterium|128|1|bacterium|128|1|s|1|

  Three spelling variants are found: "bacteria", "Bacterium", and "bacterium". As shown below, there are three lexical records related to these two base forms. C-code does not find "Bacterium" from the 3rd record.

  {base=bacterium
  entry=E0011763
  cat=noun
  variants=glreg
  }

{base=bacteria
entry=E0446596
cat=noun
variants=reg
}
{base=Bacterium
entry=E0456333
cat=noun
variants=inv
variants=uncount
}

In conclusion, java version finds more complete set of spelling variants of base forms for the input term than C-code.

III. Differences in the result of test_vf.log:

test_vf.log is the test log for testing retrieving all inflectional variants (including spelling variants). We found following two problems in C-code.

Type I. Bug in C-code for spvar

By definitions, spelling variants need to have different spelling than the input term. In C-code , there is a bug to mark a term as spelling variant. For example, "mannikin" is marked as noun:[base] and nouon:[spvar] at the same time when the spelling is identical in C-code. Other examples, such as manoeuvre, week-long, bail, etc..

Type II. Difference results from different facts

When the plural form is the same as singular form for a noun, C-code does not generate the inflectional variant of the plural form. This bug is fixed in the Java version by using the complete set of inflectional facts. Such case happens when a noun is marked with "variants=inv" (191,459), "variants=plur" (1,422), "variants=groupuncount" (851), etc.. This problem happens for the citation form and all spelling variants in such lexical records and results in many different instances.

Example 1: variants=inv

"variants=inv" is used for nouns have the same form in the singular and the plural, but remain countable. Such as terms of "sheep", "deer", "fish", "bear", etc.

For example, the lexical record of "sheep" is:

{base=sheep
entry=E0055595
cat=noun
variants=inv
}

Accordingly, the inflections of sheep are:

```
>lvg –f:i
sheep
sheep|sheep|128|1|i|1|
sheep|sheep|128|8|i|1|
sheep|sheep|128|512|i|1|
```

C-code does not find sheep as plural form (as shown on the 2nd results).

Example 2: variants=plur

"variants=plur" is used for fixed plural nouns, count nouns that lack a singular form. Such as terms of "cattle", "police", "suds", "surroundings", etc.

For example, the lexical record of "cattle" is:

```
{base=cattle
entry=E0015618
        cat=noun
        variants=plur
}
```

Accordingly, the inflections of cattle are:

```
>lvg –f:i
cattle
cattle|cattle|128|1|i|1|
cattle|cattle|128|8|i|1|
```

C-code does not find cattle as plural form (as shown on the 2nd results).

Example 3: variants=groupuncount

"variants=groupuncount" is used for uncount nouns, and fixed singular nouns which are group. Such as terms of "mankind", "intelligentsia", "laity", etc.

For example, the lexical record of "mankind" is:

```
{base=mankind
entry=E0038825
        cat=noun
        variants=groupuncount
}
```

Accordingly, the inflections of mankind are:

```
>lvg –f:i
mankind
mankind|mankind|128|1|i|1|
mankind|mankind|128|8|i|1|
mankind|mankind|128|512|i|1|
```

C-code does not find mankind as plural form (as shown on the 2nd results).