CS286 Spring 2016 Lab 1: Multi-lingual Translator MapReduce Program

Exercise 1: Write Translator from English to Multiple Languages

Objective

The objective of this exercise is to write a MapReduce program that merges a set of to-English dictionaries into a single file. The languages include French, German, Italian, Portuguese, and Spanish.

Data

The data you will use for this program is located at the "I Love Languages" Web site: http://www.ilovelanguages.com/IDP/IDPfiles.html

The data files have been provided to you in the DATA directory of your ZIP file.

Output

The output of your program a single text file with lines of the following format:

English-word: [*part of speech*] french:*french-translation*|german:*german-translation*|italian:*italian-translation*|portuguese:*portuguese-translation*|spanish:*spanish-translation*

For example,

```
hello: [Noun] french:bonjour|german:N/A|italian:ciao|portuguese:N/A|spanish:N/A
```

Note that not all words are represented in all the dictionaries. For example, the word "hello" does not have a translation in the given Portuguese dictionary. For all such words, put "N/A" to indicate the word is "not available" in the translator.

Similarly, not all records are valid. A valid record is one that contains the part of speech in addition to the word and translation. Valid parts of speech include the following: adjective, adverb, conjunction, noun, preposition, pronoun, verb.

Conversely, some dictionaries have multiple translations for a given word. For example, the word "abandon" as a noun in the French dictionary has two translations – "abandon" and "laisser-aller". In such cases, include all the translations separated by a comma. Additionally, the word "abandon" has a

translation ("abandonner") as a verb in the French dictionary. This will constitute a different record in your output because it is a different part of speech.

The lines in the output file are to be sorted on the English word, so words beginning with the letter 'a' appear in the beginning of the file, and those words beginning with the letter 'z' appear at the end of the file.

The white spaces in the output file are before and after the part of speech. Besides that, there may be white spaces within a given translation.

Last, the translation of a valid record must include every character in between the word and the part of speech.

Exercise 2: Enrich translation with another language

Objective

The objective of this exercise is to write a MapReduce program that enriches the dictionary you created in Exercise 1 with words in another language.

Data

The data you will use for this program is located in the DATA directory provided to you by the instructor. Note that you are required to use the distributed cache in your mapper to read in the data. You will also be using the output from Exercise 1 as input to this program.

Output

The output of your program is a single text file with lines of the following format:

English-word: [*part of speech*] french:*french-translation*|german:*german-translation*|italian:*italian-translation*|portuguese:*portuguese-translation*|spanish:*spanish-translation*|*newlanguage*:*newlanguage-translation*

For example,

```
hello:     [Noun]
french:bonjour|german:N/A|italian:ciao|portuguese:N/A|spani
sh:N/A|latin:N/A
```

Deliverable

IMPORTANT: I have changed the deliverable below. Please read carefully and post your questions to CANVAS if you have any doubts.

Provide the following artifacts for your solution in a single ZIP file named "`CS286_spring2016_lab1_`*`fname_lname`*`.zip`". Note that I will provide a ZIP file of this format containing starter code for the driver, mapper, and reducer.

Refactor your code such that

1. **it conforms to the following class names.**
2. **all 6 source code files belong to the same Java package called "Dictionary".**

   See the updated rebuild and rerun scripts that I've uploaded to the folder for LAB1. Make sure your drivers, mappers, and reducers conform to these updated rebuild and rerun scripts.

   1. `DictionaryDriver.java` (exercise 1)
   2. `DictionaryMapper.java` (exercise 1)
   3. `DictionaryReducer.java` (exercise 1)
   4. `CacheDriver.java` (exercise 2)
   5. `CacheMapper.java` (exercise 2)
   6. `CacheReducer.java` (exercise 2)

Notes

Do not do any "pre-processing" or "post-processing" of the data files. All your code must run in the driver, mapper, and reducer Java programs for your solution.

Some of you have mentioned you've implemented "map-only" solutions for either exercise 1 or exercise 2 or both. If that is the case, you must still provide a reducer java file for each exercise which functions as an "identity reducer". Such a reducer outputs all key-value pairs it receives as input without changing the keys or values.