Merge DENUE-ORBIS

General Description

We would be working with administrative data from Directorio Estadístico Nacional de Unidades Económicas (*DENUE*) of Mexico and with an international dataset called *ORBIS*.

- The main objective is to take each firm from Orbis and find the "llave_denue" in the Mexican data for that firm. Put differently, we are not interested in all the firms in DENUE, only in those that are the same as the firms in Orbis that we are after.
- The main variable that would tell us whether we have found a firm from ORBIS in DENUE would be the name of the firm.
- There are two main challenges:
    1. We have a bit more than 17,000 firms in Orbis and more than 7 million firms in DENUE.
    2. The same name could be written in different ways in each dataset. So an exact merge may not be possible for many firms. We are not referring only to differences such as capital letters vs. lower case letters or accents. The names might be different in other ways too (say, one includes "sociedad" and the other doesn't).
- Both datasets contain some geographic information that could be useful. There are 32 "states" in Mexico. In DENUE, states are called "entidad" and in ORBIS "region." Furthermore, we have zipcodes, municipalities, etc. Please use this information to speed up the process or to make the matches more accurate. For instance, one can restrict the search to firms within the same municipality, etc.
- While most of the work should involve no manual steps, some algorithms provide a "confidence" number for every match. As the end, we assume that we won't be able to escape some manual work when checking whether some of the matches are actually correct or not.

Main Steps

Step 1: Standardizing Geographic Units

The first step would be to standardize the names of the geographic units (states, municipalities, etc.). Remember, our goal is to use the location of a firm in each dataset to improve the match rate and quality.

First, let's start with the "states". There are 32 "states" in Mexico. In DENUE, states are called "entidad" and in ORBIS "region." In one dataset one region is called "ciudad de mexico" whereas in the other one, the entidad is called "ciudad_de_mexico." Both refer to the same geography. Please create a code that fixes those issues and leaves a unique state name in each dataset. We believe it is better if you use **Stata** for this Step.

Then, once you've standardized the "state", within each "state" check

1. What is the relation between "city" (in ORBIS) and "municipio" or "localidad" (in DENUE)? Can we standardize those geographic units as well? Or are they referring to different geographic units?
2. Do the zipcodes make sense? Are they the same across datasets (same postcode say in the same state)? Would it be correct to say that a zipcode in one dataset corresponds to the same zipcode in

the other dataset? Please take a look at the file `zip_codes.txt` in case it helps.

Step 2: Merging Names: Part I

Before starting this, please make sure that all variables are in lower-case and try to remove "unrecognized" characters to improve the quality of the match. Dealing with accents might also be valuable, like replacing all accented characters by the version without accent. Otherwise, if one version has the accent and the other doesn't, the merge will not be perfect. Feel free to use any software or programming language for this Step and the following one.

How to proceed next depends on the output of **Step 1**. In the fortunate scenario in which we can standardize geographic units (or the zipcodes make sense), we can restrict the search within each zipcode. The sub-steps would be the following:

1. First, note that some names in ORBIS are repeated (despite the ID of the firm, called bvidnumber being different). For instance, there are many exactly same-named Walmarts in Mexico. Treat them as separate case, because they are likely to be in different geographic units.
2. Take one firm from the ORBIS data. Then search across DENUE firms within the same zipcode for the firm having the closest match between the variables "name_1" and "companyname1"
3. Save the firm and the 5 closest matches (together with the "score" of the quality of the match) in a different dataset, where we will keep all the matches. Keep all variables at this point.
4. Repeat the previous two steps (in a loop across all firms in ORBIS). Save the output file.

In the case that we cannot standardize well the zipcodes, then we will proceed with 1 and 2 above restricting to the same state. Restricting the match to the same state would also be the way to go if the firm misses the postcode information, but has the state information.

We guess that this process might take around one second per firm. So the full code would take almost 5 hours to run.

Note: Not all firms have geographic information, the geographic units can only be used for those firms with the relevant geographic unit. If a firm in Orbis has no useful geographic information, the match quality will only be based on the firm name variable.

Step 3: Merging Names: Part II

- Here we take the input from the previous part.
- The idea is to use all the information possible (other names available, other geographic information) to compare the previous 5 closest matches and to re-assess which is the best match for each firm.
- Our hope is that a large share of the matches at this point would have a high degree of confidence.
- Put those "good matches" aside in a different dataset.
- Analyze the not "good matches." Can we identify a pattern in what went wrong? If yes, modify Step 2 accordingly and iterate the process once again.

Working on the server

Accessing the server

Please follow the instructions from Berkeley's EML. My username is jpvasquez. I generated a password for you: hidapa. I access it using Xming and Putty. Filezila is also very useful. The host server we will use is fargo.berkeley.edu. Port 22.

The server is like a command prompt from Windows but it uses Linux. The files we need are in:

```
cd scratch/public/jpvasquez/MNCs_informality/Raw_data
```

Important files

There are two important folders: DENUE and ORBIS.

- Within DENUE, there are several folders. The important file is in the folder `output` and the name of the file is `denue_version_d.dta`. We will call this file the "DENUE file." It is a heavy file (≈ 20 GB). There is a smaller file in the same folder called `denue_subsample.dta`. This is the "sample file" (that you can use to test the codes before running them on the full data).
- Within ORBIS, the main file is `Orbis_for_Denue_merge.dta`. This one would be the "Orbis file." I also placed there the file txt file with the zipcodes.

Using the server

- Once you connect to the server using Putty and Xming, you can type `xstata` to get a very user-friendly window with Stata. Unfortunately, there is not such interface for Python or R. For this reason, we believe that you should use Stata for the Step 1.
- Given the size of the data, the most efficient way to run codes is to run the full code on the server in `batch` mode. For this, you will first need to create a `name.sh` file. Then, use Putty to get to the folder where the file is. Finally type in the server `sbatch name.sh`.
- To check whether the file is running, you can type `squeue` in the server.
- When working with R or Python, the most efficient way is to test the code with small subsamples in your computer. Once you are sure that the code runs well on the subsample, then run the code on the full data using the server.