CS 638 - Fall 2016
Team 20 - Project Stage 3

Group Members:
Daniel Kaczmarek <dkaczmarek@wisc.edu>,
Sahit Mandala, <mandala@wisc.edu>,
Zhilin Jiang <zjiang62@wisc.edu>

- **How did you develop the final blocker? What blocker did you start with? What problems did you see? Then how did you revise it to come up with the next blocker? In short, explain the *development process*, from the first blocker all the way to the final blocker (that you submit in the Jupyter file).**

We started with using an Attribute Equivalence Blocker on attribute “year”. Blocking on tables A and B gives us table C0 with 251475 tuples.

Then we pipe C0 into an Overlap Blocker on attribute “name”, tokenizing on each word (word_level=True), and filter with minimum overlap of 1 word (overlap_size=1). The resulting table C1 has 5567 tuples.

We also experimented with running Overlap Blocking on attribute “name” directly on the original tables A and B, tokenizing on each word (word_level=True), and filter with minimum overlap of 1 word (overlap_size=1). We are able to get table C4 that has 2120506 tuples. However due to the unintended behavior of Magellan (see descriptions below), we decided not to continue with this table.

- **If you use Magellan, then did you use the debugger? If so, where in the process? And what did you find? Was it useful, in what way? If you do not use Magellan, you can skip this question.**

Yes. We used it after doing our blocking steps on year and name to see if we were incorrectly excluding tuples that we shouldn’t have been. We did not see any that were incorrectly excluded. It was useful in assuring us that we did not accidentally miss matching tuples when blocking.

- **How much time did it take for you to do the whole blocking process?**

Implementation time: 3 h
Blocking running time: ~2 mins including unused 3-gram blocking. ~15 seconds for the blockers we ended up using.

- **Report the size of table A, the size of table B, the total number of tuple pairs in the Catresian product of A and B, and the total number of tuple pairs in the table C.**

Size of table A: 6991
Size of table B: 10298
Size of the Cartesian product of A and B: 6991 * 10298 = 71993318
Total number of tuple pairs in table C: 5567

- **Did you have to do any cleaning or additional information extraction on tables A and B?**

No.

- **Did you run into any issues using Magellan (such as scalability?). Provide feedback on Magellan. Is there anything you want to see in Magellan (and is not there)? If you do not use Magellan, you can skip this question.**

We have found that **py_entitymatching.OverlapBlocker().block_tables()** produces unintended output as following:

Running

```
ob = em.OverlapBlocker()

# block using name, tokenize by q-gram of 3
C3 = ob.block_tables(A, B, 'name', 'name',
                     l_output_attrs=['name', 'year'],
              r_output_attrs=['name', 'year'],word_level=False, q_val=3, overlap_size=1,)
```

Or

```
# block using name, tokenize by word
C4 = ob.block_tables(A, B, 'name', 'name',
                     l_output_attrs=['name', 'year'],
              r_output_attrs=['name', 'year'],word_level=True, overlap_size=1,)
```

The resulting table should contain “ltable_name” and “rtable_name” attributes as their respective original values, but are incorrectly populated with tokenized words instead:

In [17]:
```
# Take a look at how blocking using just the name works for fun
ob = em.OverlapBlocker()

# block using name
C4 = ob.block_tables(A, B, 'name', 'name',
                                    l_output_attrs=['name', 'year'],
                     r_output_attrs=['name', 'year'],word_level=True, overlap_size=1,)
C4
```

```
0%                          100%
[##############################] | ETA: 00:00:00
Total time elapsed: 00:00:07
```

Out[17]:

	_id	ltable_id	rtable_id	ltable_name	ltable_year	rtable_name	rtable_year
0	0	3715	1	kontor player expansion 4 3	2002.0	front 3 rhine panzer drive 2nd exp	NaN
1	1	516	1	liberation national front	1983.0	front 3 rhine panzer drive 2nd exp	NaN
2	2	5509	1	panzer armee afrika	1973.0	front 3 rhine panzer drive 2nd exp	NaN
3	3	3906	1	panzer pranks	1980.0	front 3 rhine panzer drive 2nd exp	NaN
4	4	5641	1	up 3	1972.0	front 3 rhine panzer drive 2nd exp	NaN
5	5	3722	1	russian front	1985.0	front 3 rhine panzer drive 2nd exp	NaN
6	6	2572	1	quiet front all western	1997.0	front 3 rhine panzer drive 2nd exp	NaN
7	7	549	1	panzer grenadier korps afrika	2002.0	front 3 rhine panzer drive 2nd exp	NaN
8	8	6670	1	3 horizons set of duel vast ages	2003.0	front 3 rhine panzer drive 2nd exp	NaN
9	9	4879	1	maze 3 worlds expansion of tyrants card hell heroes	2003.0	front 3 rhine panzer drive 2nd exp	NaN
10	10	145	1	conflict front set battlecards starter russian world	2003.0	front 3 rhine panzer drive 2nd exp	NaN

Here the name value in both tables (e.g. “Panzer Exp. #3: Drive to the Rhine - The 2nd Front” in the right table) are converted to tokenized words in arbitrary order (e.g. “front exp rhine 3 panzer drive 2nd”), which we assume is due to an implementation error in the Magellan module.