Sophia Parafina
Posted on Dec 25, 2021

Speeding up geodata processing

#geopandas #python #pyogrio #pickle

I've been using the excellent geopandas for working with largish geodata sets and CSV files. While geopandas has been great for working with data, it is slow to ingest geodata. I ran a simple test to time reading a 1.2GB line shapefile into a dataframe.

```
import geopandas as gpd
import time
import pickle

# read shapefile
read_start = time.process_time()
data = gpd.read_file("Streets.shp")
read_end = time.process_time()

read_time = read_end - read_start
print(str(read_time/60)+" minutes")
```

25.437234833333333 minutes

Martin Fleischmann suggested using pyogrio instead of geopandas. The result was quite impressive.

```
# alternate package for reading data
from pyogrio import read_dataframe
import time
import pickle

# read shapefile
read_start = time.process_time()
data = read_dataframe("Streets.shp")
read_end = time.process_time()

read_time = read_end - read_start
print(str(read_time/60)+" minutes")
```

2.9936875333333335 minutes

While going from 25 minutes to 3 minutes is quite an improvement, I'm building out a data processing pipeline and I want to reduce read time even more. My next experiment uses Python's pickle which serializes the data frame into a byte stream and writes it to a file. Here are the results from pickling the dataframe and reading the pickled data.

```
#create a file
picklefile = open('streets', 'wb')

#pickle the dataframe
pickle_write_start = time.process_time()
pickle.dump(data, picklefile)
pickle_write_end = time.process_time()

#close file
picklefile.close()

pickle_write = (pickle_write_end - pickle_write_start)/60
print(str(pickle_write)+" minutes")
```

4.362236583333333 minutes

```
#read the pickle file
picklefile = open('streets', 'rb')

#unpickle the dataframe
pickle_read_start = time.process_time()
df = pickle.load(picklefile)
pickle_read_end = time.process_time()

#close file
picklefile.close()

pickle_read = (pickle_read_end - pickle_read_start)/60
print(str(pickle_read)+" minutes")
```

0.9217719833333339 minutes

Wow! We've gone from 25 minutes to read a 1.2GB shapefile to less than a minute.

Finally, pickling the dataframe reduces the shapefile size from 1.2GB to 984MB. However, pickled data can be efficiently compressed.

```
import gzip
import shutil
with open('streets', 'rb') as f_in:
    with gzip.open('streets.gz', 'wb') as f_out:
        shutil.copyfileobj(f_in, f_out)
```

The compressed file is 78.8MB, a bit more than a 10x reduction in file size.

Conclusion

If you're working with geodata that remains static, you can improve geopanda's read times by using pyogrio and pickling the resulting dataframe. Additionally, pickle files can be compressed efficiently, which can lower costs for data egress when using cloud storage.

> Pickle image by Renee Comet

Top comments (0)