Colorado Department of Transportation: Bid Info Extraction

Abstract

This document is meant to provide guidance on the steps necessary to i) locate the bid summaries, ii) download and save the relative .pdf files, iii) extract the relevant information (i.e., the number of bidders and the winning bid amount), and iv) save the info in tabular format. These guidelines are written in python (3.5.x), but the general tips could be applied with whatever language you are able to code with.

Overview

With pyhton, we proceed by getting the main page - figure **??**, panel A - through the REQUESTS module, and by parsing it with the BEAUTIFULSOUP module. The goal of this subtask is to extract the links underlying the "Bid Tab" buttons: these lead to .pdf files containing the bid summary.

Panel A) Panel B)

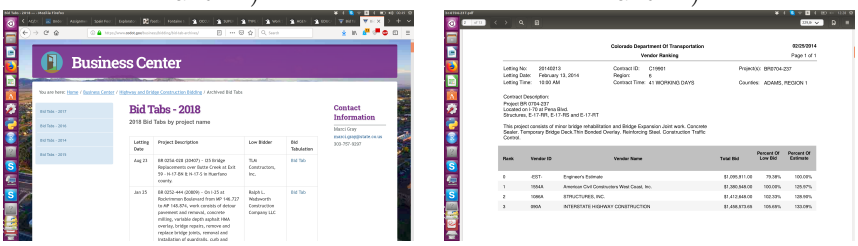

Figure 1: Colorado DoT: Main Result Page (panel A) and bid summary .pdf file (panel B)

The .pdf files - generally 30 to 40 pages long - report a table with a short summary of the auction results (page 1) featuring the contract id, the winning firm name and the winning bid; on page 2 (see figure **??**, panel B) it is reported the *vendor ranking*, with the identity, bid and rank of *all* auction participants. The final goal of the spider will be to parse just two elements of these tables - i.e., the highest rank (*# bidders*) and the bid of the winner (*lowest bid*).

Once done, we will have to save them in machine-readable format (*csv*).

Step 1: Identify the Bid Summaries

The main page contains, year by year, all auctions. Looking at the HTML code beneath the page (figure **??**) one can immediately see that all hrefs are stored in "a" elements within a table in the "content-core" element.

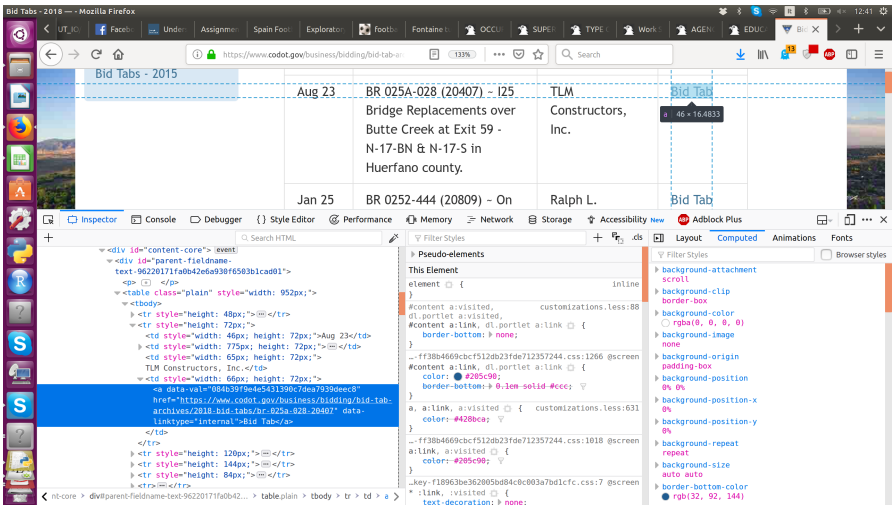

Figure 2: Colorado DoT: main page html structure

Once identified the html structure, the code should reflect it: a simple GET request will allow to take the page, and through BeautifulSoup will be possible to save a list containing all the hrefs (i..e, links) in the table.

```
# Imports #
import requests # module to run GET requests
from bs4 import BeautifulSoup # module to parse html-structured pages

# Function to parse the page and extract the links (hrefs) #
def _getUrls_(res):
    # save the links and names of every auction in the page
    hrefs = []
    soup = BeautifulSoup(res.text, 'lxml')
    main_content = soup.find('div',{'id' : 'content-core'})
    table = main_content.find("table")
    for a in table.findAll('a', href=True):
        hrefs.append(a['href'])
    return(hrefs)

# Main section #
bidurl = 'https://www.codot.gov/business/bidding/bid-tab-archives' # main page for 2018
r = requests.get(bidurl) # GET the page with the hrefs
hrefs = _getUrls_(r) # save the list with the hrefs
```

Python code 1: Extract links from the page.

Step 2: Download and Save the .pdf Files

The hrefs lead to the .pdf files, auction by auction. Save the file in python is particularly easy: after having downloaded the file (GET request), it is possible to save a .pdf file by writing down the content of the requested page. Some error handling may help during the process - there are hrefs not working.

```python
# Function to save the pdf, provided a list of hrefs #
def _getPdfs_(hrefs, basedir):
    # donwload the pdf files (from the hrefs)
    for i in range(len(hrefs)):
        print(hrefs[i])
        respdf = requests.get(hrefs[i])
        pdffile = basedir + "/pdf_dot/" + hrefs[i].split("/")[-1] + ".pdf"
        try:
            with open(pdffile, 'wb') as p:
                p.write(respdf.content)
                p.close()
        except FileNotFoundError:
            print("No PDF produced")
```

Python code 2: Function to save pdfs from links.

Step 3: Extract the Relevant Info

To extract the info, it is mandatory to parse the .pdf files, one by one. The python libraries TABULA and PANDAS are meant to extract info stored in tabular format within .pdf files. The general structure of the code is the following: open the .pdf file, identify the relevant table (i.e., page 2), and store the table of ranked bids. Once done, the number of bids will be the max number of ranked firms, and the lowest bid will be the bid of the firm ranked first - i.e., the winning firm.

In python, the READ_PDF() function of TABULA is able to identify the table in a given .pdf page, and store it in a PANDAS-type dataframe - pay attention to the rows and columns that the function identifies by default.

Step 4: Save the Data in .csv format

In order to save the data in usable format, every row of the dataset should contain the number of bids, the lowest bid and the auction name (optional). To do that, in a *for* loop - i.e., pdf by pdf - one should save the saved elements row by row through the CSV python module.

```
# read the .pdf files in the directory, extract the first line with # bids and winning bid, save it
import tabula
import pandas as pd
import re
import csv

outfile = basedir + '/bids.csv'

pdfdir = basedir + '/pdf_dot/'
with open(outfile, 'a') as c:
    writer = csv.writer(c)
    writer.writerow(['Auction','Num Bids','Lowest Bid'])
    for pdf in os.listdir(pdfdir):
        print(pdf)
        auction_name = pdf.split(".")[0]
        pdffile = pdfdir + pdf
        try:
            df = tabula.read_pdf(pdffile, pages = '2', multiple_tables = True) # read the bid summary table - at p
            df_clean = df[0][pd.notnull(df[0][0])].loc[2:] # the first 2 columns are just the table headers
            num_bids = df_clean[0].max() # the max number in the "Rank" column: it is the number of bids
            winning_bid = df_clean[4][df_clean[0] == "1"].item().replace("$","") # print the bid of the bid ranke
            if re.search("%", winning_bid): # if the table format is different from standard (low bid == column 4)
                winning_bid = df_clean[3][df_clean[0] == "1"].item().replace("$","")
            row = [auction_name,num_bids, winning_bid] # this is the outcome info: name, # bids, winning bid
            writer.writerow(row)
        except:
            continue
```

Python code 3: Code to extract info from .pdf files and save it in tabular format.