Hw2

Steps

Repo link - https://github.com/ArkashJ/hw2_mini_internets

- Generate the mini_internet on my local machine and stored the html files in a folder
- Logged into google console using `gcloud auth login`
- Set a project: `gcloud config set project $courseid_on_personal_gcloud`
- Generated a storage bucket with the link `gs://hw2-arkjain-mini-internet/ --location=US-EAST1`

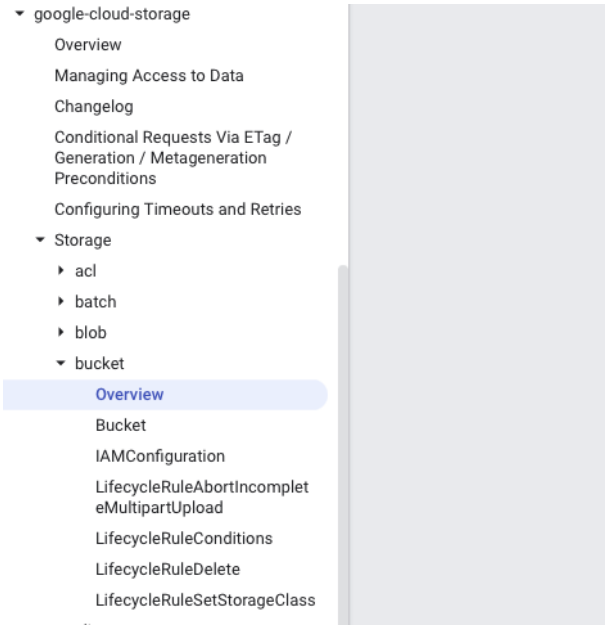
- Useful information

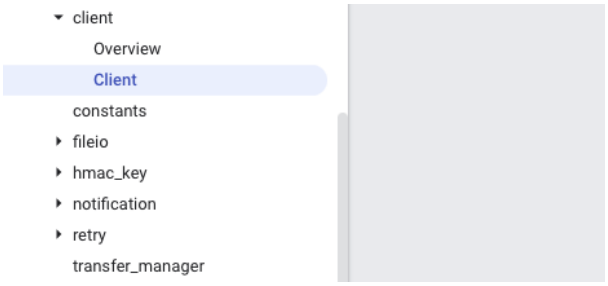
```
    project: str = "CloudComputingCourse",
    bucket_name: str = "hw2-arkjain-mini-internet",
    blob_name: "mini_internet_test"
```

- `gcloud storage buckets describe $LINK` explains information about the bucket

```
hw2 on  main via  v3.11.4 on ☁  arkjain@bu.edu
❯ gcloud storage buckets describe gs://hw2-arkjain-mini-internet
acl:
- entity: project-owners-820402225330
  projectTeam:
    projectNumber: '820402225330'
    team: owners
  role: OWNER
- entity: project-editors-820402225330
  projectTeam:
    projectNumber: '820402225330'
    team: editors
  role: OWNER
- entity: project-viewers-820402225330
  projectTeam:
    projectNumber: '820402225330'
    team: viewers
  role: READER
creation_time: 2023-09-21T23:13:25+0000
default_acl:
- entity: project-owners-820402225330
  projectTeam:
    projectNumber: '820402225330'
    team: owners
  role: OWNER
- entity: project-editors-820402225330
  projectTeam:
    projectNumber: '820402225330'
    team: editors
  role: OWNER
- entity: project-viewers-820402225330
  projectTeam:
    projectNumber: '820402225330'
    team: viewers
  role: READER
default_storage_class: STANDARD
location: US-EAST4
location_type: region
metageneration: 1
name: hw2-arkjain-mini-internet
public_access_prevention: inherited
storage_url: gs://hw2-arkjain-mini-internet/
uniform_bucket_level_access: false
update_time: 2023-09-21T23:13:25+0000
```

```
# count file
ls | wc -l
# make bucket
gcloud storage buckets create gs://hw2-arkjain-mini-internet --location=US-EAST4
# work on the mini internet folder
gsutil ls gs://hw2-arkjain-mini-internet/
```





- Documentation Images (used to setting up buckets on the cloud in python and moving files

- Used gsutil to move files from local machine as blobs in the bucket

  `gsutil -m cp -r mini_internet gs://hw2-arkjain-mini-internet/`

  `gsutil ls`

Running on the cloud with multi threading

- Started threads from the python `from concurrent.futures import ThreadPoolExecutor, as_completed` library and ran the blobs on threads

```
with ThreadPoolExecutor(max_workers=os.cpu_count() * 5) as executor:
        futures = [
            executor.submit(extract_numbers_from_html, blob.name)
            for blob in blobs
        ]
        total_len = len(futures)
        for future in tqdm(
            as_completed(futures),
            total=total_len,
            desc="Extracting numbers from html files",
        ):
            arr, file_path = future.result()
            if arr is []:
                continue
            iter = int(file_path.split("/")[-1].split(".")[0])
            links_dict[iter] = arr
    return links_dict
```

Page rank logic

- For each file first use regex to get the tags from the html readers
  - Parse these headers to store the tags in an array

- Takes in a file path - could be a blob or a local file path and returns an array with the correct “page links” along with the name of the file which can be used to construct a dictionary

```
def extract_numbers_from_html(
    html_file_path: str,
    pattern: str = r'HREF="(\d+)\.html"',
) -> Optional[list[int]]:
```

- Make an adjacency matrix of size 10000×10000 where we iterate through the dictionary and for each page, whichever page it is linked to, we set the value to be 1 for that page in the row. Meaning that for page 1, we will set the value of columns 62, 240 etc. to be 1

```
def construct_adjacency_matrix(links_dict: dict) -> list[list[int]]
# The rows in the matrix are the outgoing links and
# The columns are the incoming links
# if the row has a 1, it means that the column has a
# link to the row otherwise 0
    adj_mat = np.array(
        [[0 for i in range(MAX_NUM_FILES)] for j in range(MAX_NUM_FILES)]
    )
    for key, value in links_dict.items():
        curr_row = adj_mat[key]
        for link in value:
            curr_row[int(link)] = 1
    return adj_mat
```

- Once the matrix is made we run the page rank function where in a while loop, for each iteration, we update the pagerank value for every single page in the 10000 sized page rank list that we generated with the default value of $1/10000$ for each of the elements (I assumed that initially we can have every pagerank we the *expected value* of the page being linked to another one

```
@jit(nopython=True)
def pagerank(
    adj_mat: list[list[int]],
    pr_addition_const: float = 0.15,
    pr_damping_factor: float = 0.85,
    epsilon: float = 0.005,
) -> list[float]:
    """
    ## Description:
    In a while loop, for each incoming edge i in the 10000,
    you pick the outgoing edge j and sum the number for this edge,
    adding it to get the total number of outgoing edges for the page.
    #  PR(A) = 0.15 + 0.85 * (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))

    Explanation:
    - For each "iteration" of the while loop, we update the pagerank
      matrix for all MAX_NUM_FILES pages
      and then we check if the percent change between the old and
      new pagerank matrix is less than epsilon
    - If the percent change is less than epsilon, we return
      the pagerank matrix
    """
```

- Pagerank calculation:

```
for i in range(MAX_NUM_FILES):
    sum_rows = 0.0
    for elem in elems_dict[i]:
    # For each column, we get the page
    #rank and divide it by the sum of the outgoing links for that page
         sum_rows += page_rank_mat[elem] / sums_dict[elem]
```

```
    # Page rank calculation.
    page_rank_mat[i] = pr_addition_const + (pr_damping_factor * sum_rows)
```

- Statistics calculation:
  - we know that the incoming links are the columns in the adjacency matrix
  - and the outgoing links are the rows in this matrix
  - Performing math operations on them is trivial using numpy inbuilt classes

I’ve made a class called Pagerank where all these helper functions are called

- There’s flags I’ve defined in the code which allow on to run the code locally, on the cloud and also test it by running the local and cloud code together

- THE CODE HAS DETALED EXPLANATIONS FOR EVERYTHING
  - Some of the instructions and a brief overview of the code is as follows:

````
# hw2_mini_internet

https://cdn-uploads.piazza.com/paste/l78dcjpoj615ed/59d96dd159ba1094a7b22833e823fcbca915783586cb05a1bb38fc1b9c4a649f/DS_561-HW_2.pdf

## Run on cloud
```
python3 page_rank_alg.py --cloud
```
## Steps to run the code
1. Create a virtual environment
```
python3 -m venv env
```
2. Activate the virtual environment
```
source venv/bin/activate
```
(use activate.fish if using fish shell)
3. Install the requirements
```
pip3 install -r requirements.txt
```
4. Make the mini_internet
```
python3 generate_content.py
```
5. Move .html files to a folder called "mini_internet"
```
mkdir mini_internet
mv *.html mini_internet
```
Note: if you use another directory name, you will need to
 change the directory name in the code.
Go the iter_files_and_get_links function in pagerank.py
and change the directory name

6. Run the code
```
python3 pagerank.py --local
```

## Options for running the code
RUN LOCALLY
```
python3 pagerank.py --local
```
RUN ON CLOUD
````

```
python3 pagerank.py --cloud
```
TEST LOCALLY
Note: you may need to change the epsilon value
in (pagerank function) the code to 0.0001
```
python3 pagerank.py --test
```

Python Commands
Running Python lint to check style
```
pylint pagerank.py
```

Lint code
```
black pagerank.py
```

Regex
Running regex.compile to prevent making regex objects again anad again
```
re.compile(str)
```

Numba
Use numba only on static classes and do not pass in
 complex data types into the function or call them in the array

Logic
- Make an adjacency matrix of the graphs where the rows
are the nodes and the columns are the edges
- Compute the outgoing and incoming edges of each node
- in each iteration, get the pagerank for each node
- check if the pagerank is converging

Cost Calculation:

Service	Cost	Discounts	Promotions and others	Subtotal	% Change
Cloud Storage	$0.08	$0.00	-$0.08	$0.00	0%
Networking	$0.01	-$0.01	$0.00	$0.00	0%
Compute Engine	$0.00	$0.00	$0.00	$0.00	0%