BONOBO TUTORIAL

Week 1

By Beidan Huang

ABOUT ME

Beidan Huang

Master of Computer Science, UCSD

Master of Finance, Rady (graduated in 2018)

Email: b5huang@eng.ucsd.edu

PIAZZA!

Ask the questions in piazza so that questions can be shared among the class!

Professor and TA are there to help you!

A good question will get a good fast response!

http://piazza.com/rady.ucsd/winter2020/mgta49599910

OR Login To CANVAS --- PIAZZA --- ENROLL

BONOBO

- A lightweight Extract-Transform-Load (ETL) framework for Python 3.5+
  - Building Data Processing Pipelines
  - Executing them in Parallel

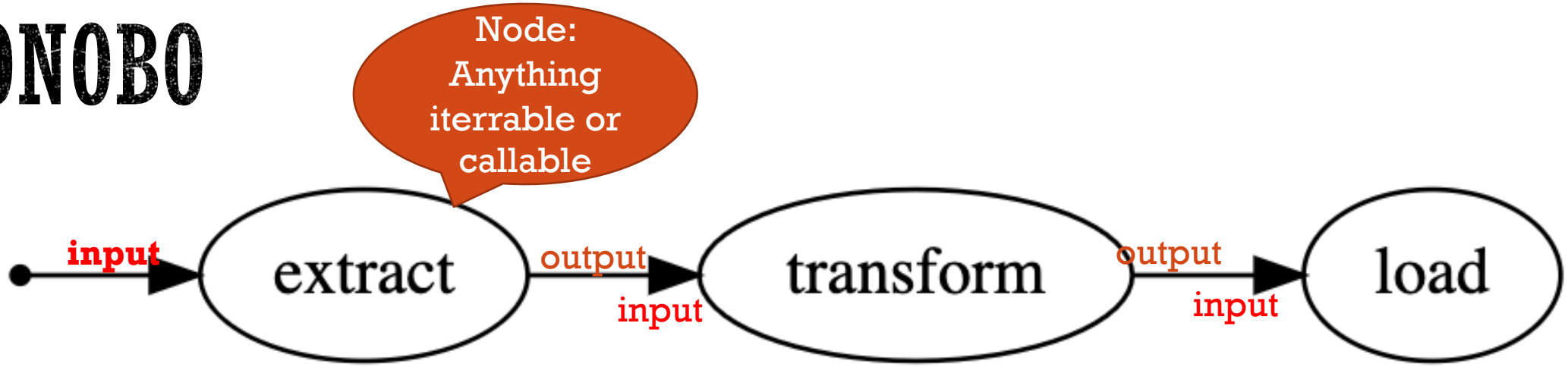
Keyword:

Pipelining

Efficient!

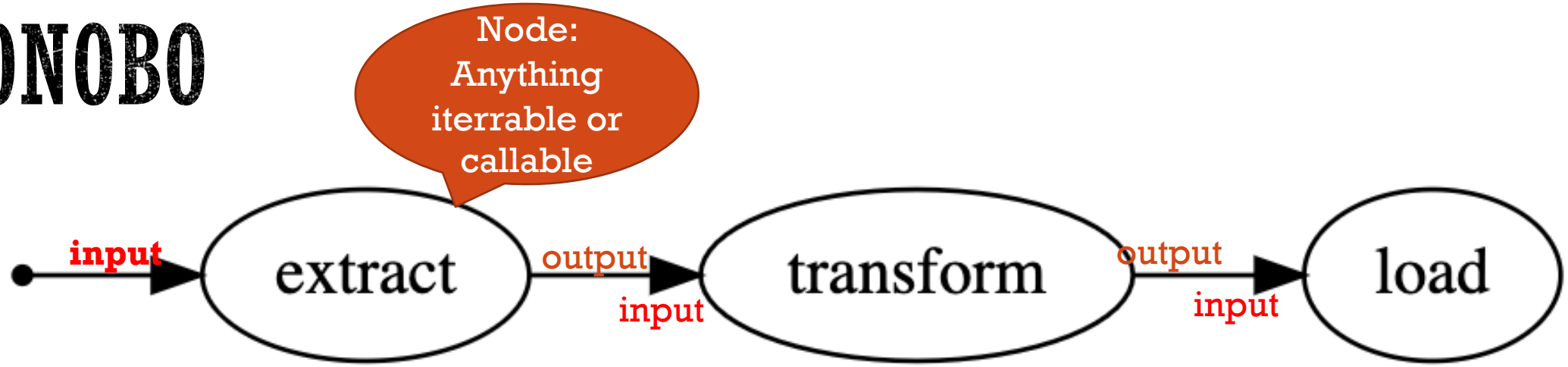
PIPELINING

BONOBO



- Write atomic data transformation using Python (IMPROTANT)
- Combine the nodes in directed graphs as pipelining
- Execute in parallel

BONOBO



- First in, first out – the order of rows will be preserved
- Parallelism
- Independence

BONOBO EXAMPLES

- GitHub Link to my codes:
- https://github.com/huangbeidan/bonobo_tutorial/tree/master

A SIMPLE EXAMPLE

```python
import bonobo

def extract():
    yield 'hello'
    yield 'world'

def transform(*args):
    yield tuple(map(str.capitalize, args))

def load(*args):
    print(*args)

def get_graph(**options):
    graph = bonobo.Graph()
    graph.add_chain(extract, transform, load)
    return graph
```

```
Beidans-MacBook-Pro:bonobo_tutorial beidan$ python3 bonobo_basic.py
Hello
World
 - extract in=1 out=2 [done]
 - transform in=2 out=2 [done]
 - load in=2 [done]
```

THE OUTPUTS

```python
def extract():
    """Placeholder, change, rename, remove... """
    time.sleep(5)
    yield 'hello'
    time.sleep(5)
    yield 'world'

def transform(*args):
    """Placeholder, change, rename, remove... """
    time.sleep(5)
    yield tuple(
        map(str.capitalize, args)
    )

def load(*args):
    """Placeholder, change, rename, remove... """
    time.sleep(5)
    print(*args)

def get_graph(**options):
    graph = bonobo.Graph()
    graph.add_chain(extract, transform, load)
    return graph
```

ADD DELAYS

- (Run)

1

Get the price column from csv

2

Convert to “Transform: [price]” format

3

Write the outputs to file

READ FROM EXCEL

READ FROM EXCEL

- **Reading columns from Excel**

```
def get_price():
    data = pd.read_csv('train.csv')
    price = data['SalePrice'].tolist()
    yield from price

def transform(*args):
    # args always return with tuples!
    # print(args)
    data = str(args[0])
    # print(data)
    yield "Transform: " + data + ", data2:" + da

def with_opened_file(self, context):
    with open('output_csv.txt', 'w+') as f:
        yield f
```

```
/usr/local/bin/python3.7 /U
(208500,)
(181500,)
(223500,)
(140000,)
(250000,)
(143000,)
(307000,)
(200000,)
(129900,)
(118000,)
(129500,)
(345000,)
```

```
_opened_file)
row):
")
```

READ FROM EXCEL - MORE

- **Reading rows from Excel**

```
# prepare for the data
data = pd.read_csv('train.csv', encoding='ISO-8859-1')

def extract():
    yield from data.itertuples()
```

READ FROM EXCEL - MORE

- **Reading tuples from Excel**

`DataFrame.`**`itertuples`**(*self*, *index=True*, *name='Pandas'*) [source]

Iterate over DataFrame rows as namedtuples.

Parameters:	**index** : *bool, default True* If True, return the index as the first element of the tuple. **name** : *str or None, default "Pandas"* The name of the returned namedtuples or None to return regular tuples.
Returns:	iterator An object to iterate over namedtuples for each row in the DataFrame with the first field possibly being the index and following fields being the column values.

READ FROM EXCEL - MORE

- **Reading tuples from Excel**
- **How to access the 5th element of a particular row?**

```
# filter out the rows (yearbuilt < 1980)
idx0 = category_dict['YearBuilt'] + 1
if int(args[idx0]) < 1980:
    return None
```

GET DATA FROM JSON

- Dataset


```
{
   "nhits":995,
   "parameters":{
      "dataset":"fablabs",
      "timezone":"UTC",
      "rows":1000,
      "format":"json"
   },
   "records":[
      {
         "datasetid":"fablabs",
         "recordid":"e1b3d0f2dbe6a25c95269ef11b80209ea87a1f42",
         "fields":{
            "city":"Sau\u00f0\u00e1rkr\u00f3kur",
            "name":"FabLab Saudarkrokur, Innovation Center Iceland",
            "links":"[{\"url\": \"http://www.fablab.is/w/index.php/Main_Page/%C3%8Dslenska\", \"id\": 69}]",
            "url":"https://www.fablabs.io/labs/fablabsaudarkrokurinnovationcentericeland",
            "capabilities":"three_d_printing;cnc_milling;circuit_production;laser;precision_milling;vinyl_cutting",
            "county":"Northwest",
            "longitude":-19.6406055251313,
            "country_code":"is",
            "latitude":65.7416349296638,
            "coordinates":[
               65.7416349297,
               -19.6406055251
            ],
            "kind_name":"fab_lab"
         },
         "geometry":{
            "type":"Point",
            "coordinates":[
               -19.6406055251,
               65.7416349297
            ]
         },
         "record_timestamp":"2017-03-22T16:28:59.051000+00:00"
      },
```


First, crawl content from website

Then, extract the parts we want

Finally, write the outputs into files

GET DATA FROM JSON

- Extractor

```
import requests

FABLABS_API_URL = 'https://public-us.opendatasoft.com/api/records/1.0/

def extract_fablabs():
    yield from requests.get(FABLABS_API_URL).json().get('records')
```

GET DATA FROM JSON

- Transform and Load step

```
import bonobo

def get_graph(**options):
    graph = bonobo.Graph()
    graph.add_chain(
        extract_fablabs,
        bonobo.Limit(10),
        bonobo.PrettyPrinter(),
    )
    return graph
```

GET DATA FROM JSON

- Output (Run first)

```
- extract_fablabs in=1 out=995 [done]
- Limit in=995 out=5 [done]
- PrettyPrinter in=5 out=5 [done]
```

GET DATA FROM JSON

- Another way of using Request Library

```
def get_services():
    http = requests.Session()
    http.headers = {'User-Agent': 'Monkeys!'}
    return {
        'http': http
    }

@use('http')
def extract_fablabs(http):
    yield from http.get(FABLABS_API_URL).json().get('records')
```

GET DATA FROM JSON

- How to get the first element of records? (e.g. datasetid)

- How to get an element inside an array (e.g. county)
  - What if no county in the fields?


```
{
   "nhits":995,
   "parameters":{
      "dataset":"fablabs",
      "timezone":"UTC",
      "rows":1000,
      "format":"json"
   },
   "records":[
      {
         "datasetid":"fablabs",
         "recordid":"e1b3d0f2dbe6a25c95269ef11b80209ea87a1f42",
         "fields":{
            "city":"Sau\u00f0\u00e1rkr\u00f3kur",
            "name":"FabLab Saudarkrokur, Innovation Center Iceland",
            "links":"[{\"url\": \"http://www.fablab.is/w/index.php/Main_Page/%
            "url":"https://www.fablabs.io/labs/fablabsaudarkrokurinnovationcent
            "capabilities":"three_d_printing;cnc_milling;circuit_production;las
            "county":"Northwest",
            "longitude":-19.6406055251313,
            "country_code":"is",
            "latitude":65.7416349296638,
            "coordinates":[
               65.7416349297,
               -19.6406055251
            ],
            "kind_name":"fab_lab"
         },
         "geometry":{
            "type":"Point",
            "coordinates":[
               -19.6406055251,
               65.7416349297
            ]
         },
         "record_timestamp":"2017-03-22T16:28:59.051000+00:00"
      },
```

GET DATA FROM JSON - MORE

- **Getting first element of “records”**

```
@use('http')
def extract_fablabs(http):
    yield from http.get(FABLABS_API_URL).json().get("records")

def transform(*args):
    first_element = args[0]['datasetid']
    return first_element
```

GET DATA FROM JSON - OUTPUT

- **Getting first element of “records”**

```
/usr/local/bin/python3.7 /Users/beidan/PycharmProjects/bonobo_tutorial/bonobo_tutorial2.py
- 0 : fablabs
- 0 : fablabs
- 0 : fablabs
- 0 : fablabs
- 0 : fablabs
```

GET DATA FROM JSON - MORE

- **Getting element inside an array**

```
@use('http')
def extract_fablabs(http):
    yield from http.get(FABLABS_API_URL).json().get("records")

def transform(*args):
    fields = args[0]['fields']
    try:
      return fields['county']
    except Exception as e:
        print("no county in fields available")
```

GET DATA FROM JSON - OUTPUT

- **Getting element inside an array**

```
- 0 : Northwest
- 0 : Canton of Zurich
no county in fields available
- 0 : Catalonia
- 0 : Principality of Asturias
```

TIPS ON HOW TO DEBUG

- Simplest way: just print out
- Don’t be afraid! You will and must learn from debugging.

HELPFUL FUNCTIONS FOR YOUR HW

- Helpful functions are all there in the previous slides
- Try first! I'll publish my example codes later.
- In which, I will provide complete steps for writing one table.
- You task is to produce all three required tables!

USEFUL LINKS

- Reference to Python Decorators (Chinese)
- https://blog.csdn.net/catwan/article/details/84975817

- Reference to yield generator
- https://www.geeksforgeeks.org/use-yield-keyword-instead-return-keyword-python/

- Reference to Python Decorators (English)
- https://www.geeksforgeeks.org/decorators-in-python/