Project concept

Dawid Białka

Kamil Burkiewicz

About lithops

Lithops is a Python multi-cloud distributed computing framework. It allows you to run unmodified local python code at massive scale in the main serverless computing platforms. Lithops delivers the user's code into the cloud without requiring knowledge of how it is deployed and run. Moreover, its multicloud-agnostic architecture ensures portability across cloud providers. Lithops is specially suited for highly-parallel programs with little or no need for communication between processes, but it also supports parallel applications that need to share state among processes. Examples of applications that run with Lithops include Monte Carlo simulations, deep learning and machine learning processes, metabolomics computations, and geospatial analytics, to name a few.

Topic

- AWS deployment automation
- App management and scaling

The main purpose of the project is to test how well lithops performs with large and parallel data analysis using map-reduce concept.

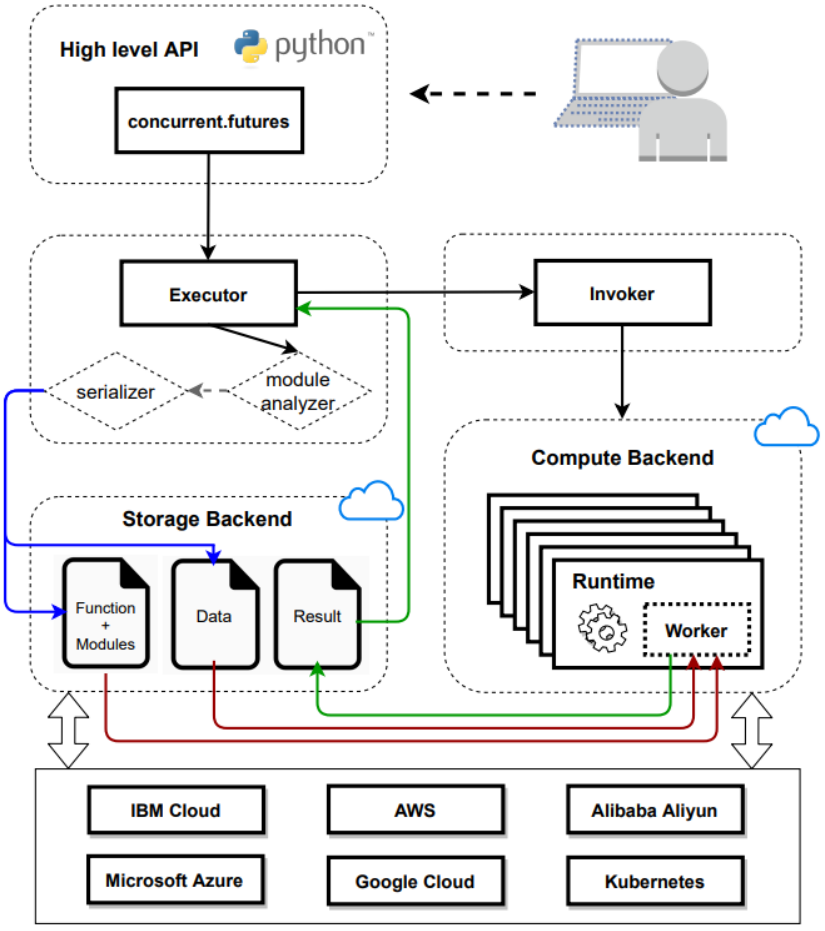
Challenges

- Large dataset processing

Main problems to solve

- Setting up the whole infrastructure
- Find and download large amount of appropriate data to analyze
- Decide what analysis to perform
- Measuring how well the system performs under various conditions

Architecture scheme



Planned tests

Infrastructure:

Compute backend: AWS Lambda

- 2048 MB memory
- cpu type: `Intel(R) Xeon(R) Processor @ 2.50GHz model 63`
- timeout: 900s
- vCPUs: 2 (scaled to 1.15)*
- lithops invokes lambdas asynchronously by design

Storage: S3

Dataset:

- World air pollution data. Source: https://registry.opendata.aws/openaq/
- Around 50 GB of data, divided into files ~ 25 MB each and uploaded to S3 storage

	date	parameter	value	unit	averagingPeriod	location	city	country	coordinates	attribution	sourceName	sourceType	mobile
0	{'utc': '2020-10-31T08:30:00.000Z', 'local': '...	pm25	100.000000	µg/m³	{'value': 1, 'unit': 'hours'}	US Diplomatic Post: Kabul	Kabul	AF	{'latitude': 34.535812, 'longitude': 69.190514}	[{'name': 'EPA AirNow DOS', 'url': 'http://air...	StateAir_Kabul	government	False
1	{'utc': '2020-10-31T07:30:00.000Z', 'local': '...	pm25	45.000000	µg/m³	{'value': 1, 'unit': 'hours'}	US Diplomatic Post: Kabul	Kabul	AF	{'latitude': 34.535812, 'longitude': 69.190514}	[{'name': 'EPA AirNow DOS', 'url': 'http://air...	StateAir_Kabul	government	False
2	{'utc': '2020-10-31T08:30:00.000Z', 'local': '...	pm25	46.000000	µg/m³	{'value': 1, 'unit': 'hours'}	US Diplomatic Post: Kabul	Kabul	AF	{'latitude': 34.535812, 'longitude': 69.190514}	[{'name': 'EPA AirNow DOS', 'url': 'http://air...	StateAir_Kabul	government	False
3	{'utc': '2020-10-31T09:30:00.000Z', 'local': '...	pm25	48.000000	µg/m³	{'value': 1, 'unit': 'hours'}	US Diplomatic Post: Kabul	Kabul	AF	{'latitude': 34.535812, 'longitude': 69.190514}	[{'name': 'EPA AirNow DOS', 'url': 'http://air...	StateAir_Kabul	government	False
4	{'utc': '2020-10-31T10:30:00.000Z', 'local': '...	pm25	39.000000	µg/m³	{'value': 1, 'unit': 'hours'}	US Diplomatic Post: Kabul	Kabul	AF	{'latitude': 34.535812, 'longitude': 69.190514}	[{'name': 'EPA AirNow DOS', 'url': 'http://air...	StateAir_Kabul	government	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
4995	{'utc': '2020-10-31T12:00:00.000Z', 'local': '...	so2	6.000000	µg/m³	{'unit': 'hours', 'value': 1}	GR0020A	KENTPIKH MAKEΔONIA	GR	{'latitude': 40.67354965, 'longitude': 22.8934...	[{'name': 'EEA', 'url': 'http://www.eea.europa...	EEA Greece	government	False
4996	{'utc': '2020-11-01T04:00:00.000Z', 'local': '...	pm10	12.500000	µg/m³	{'unit': 'hours', 'value': 1}	FR05090	Seine-Maritime	FR	{'latitude': 49.5146953797856, 'longitude': 0....	[{'name': 'EEA', 'url': 'http://www.eea.europa...	EEA France	government	False
4997	{'utc': '2020-11-01T04:00:00.000Z', 'local': '...	o3	44.113987	µg/m³	{'unit': 'hours', 'value': 1}	FI00357	Lapland	FI	{'latitude': 68.477009999634, 'longitude': 28....	[{'name': 'EEA', 'url': 'http://www.eea.europa...	EEA Finland	government	False
4998	{'utc': '2020-11-01T06:00:00.000Z', 'local': '...	pm25	2.137920	µg/m³	{'value': 24, 'unit': 'hours'}	Heerlen-Jamboreepad	Heerlen	NL	{'latitude': 50.9003, 'longitude': 5.986850000...	[{'name': 'RIVM', 'url': 'http://www.lml.rivm....	Netherlands	government	False
4999	{'utc': '2020-11-01T06:00:00.000Z', 'local': '...	no2	0.005300	ppm	{'unit': 'hours', 'value': 1}	Renwu	埔里市	TW	{'latitude': 22.689056, 'longitude': 120.332631}	[{'name': 'http://opendata.epa.gov.tw/', 'url':...	Taiwan	government	False

5000 rows × 13 columns

Analysis:

- Computing average day values of the pollutants, grouping by every station.

```
[25] def day_average_map(x):
       if isinstance(x, str):
         df = pd.DataFrame(loads_recursive(json.loads(download_bytes(os.path.basename(x), input_bucket).decode(encoding='utf-8'))))
       else:
         df = x
       df_to_process = preprocess(df)
       df_sum = df_to_process.groupby(['country', 'city', 'date', 'parameter', 'latitude', 'longitude'])['value'].agg(['sum','count'])
       df_to_process = df_to_process.drop(columns=['value'], axis=1)
       df_merged = pd.merge(df_sum, df_to_process, on=['country', 'city', 'date',  'parameter', 'latitude', 'longitude']).drop_duplicates()
       return df_merged.to_json()
```

```
[26] def day_average_reduce(results):
       dfs = [pd.read_json(result) for result in results]
       df_concatenated = pd.concat(dfs)
       df_concatenated.reset_index(drop=True, inplace=True)

       df_avg = df_concatenated.groupby(['country', 'city', 'date', 'parameter', 'latitude', 'longitude']).sum(['count', 'sum'])
       df_avg['average'] = df_avg.apply(lambda row: row['sum'] / row['count'], axis=1)

       df_avg = df_avg.drop(columns=['count', 'sum'], axis=1).dropna()
       df_concatenated = df_concatenated.drop(columns=['count', 'sum'], axis=1).dropna()

       df_merged = pd.merge(df_avg, df_concatenated, on=['country', 'city', 'date', 'parameter', 'latitude', 'longitude']).drop_duplicates()
       return df_merged.to_json()
```

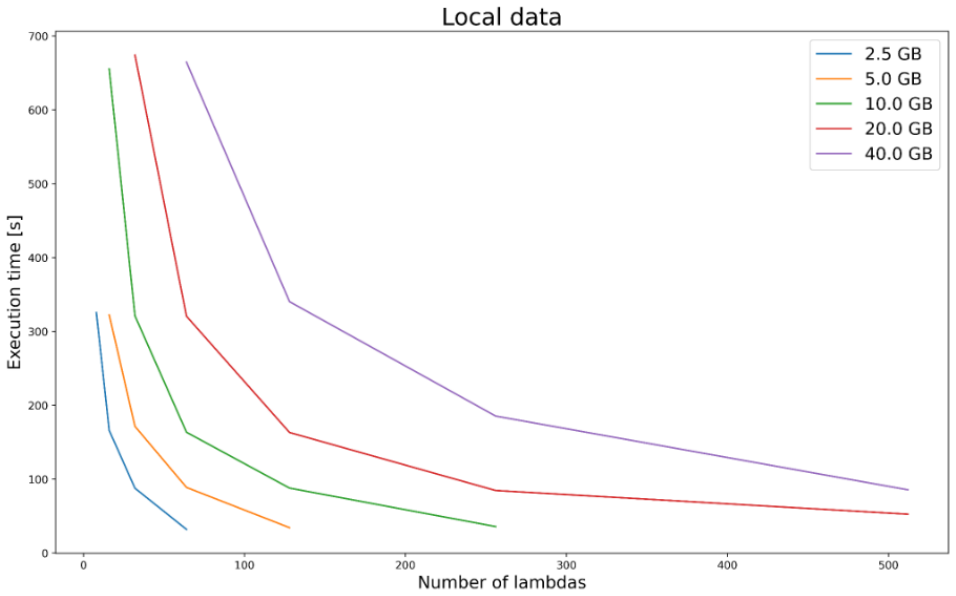
Metrics:

- Scalability
- Utilization
- Network usage
- Time from running analysis to task distribution
- Influence of data locality on system efficiency - S3 bucket located on the same region as the computations and on other one

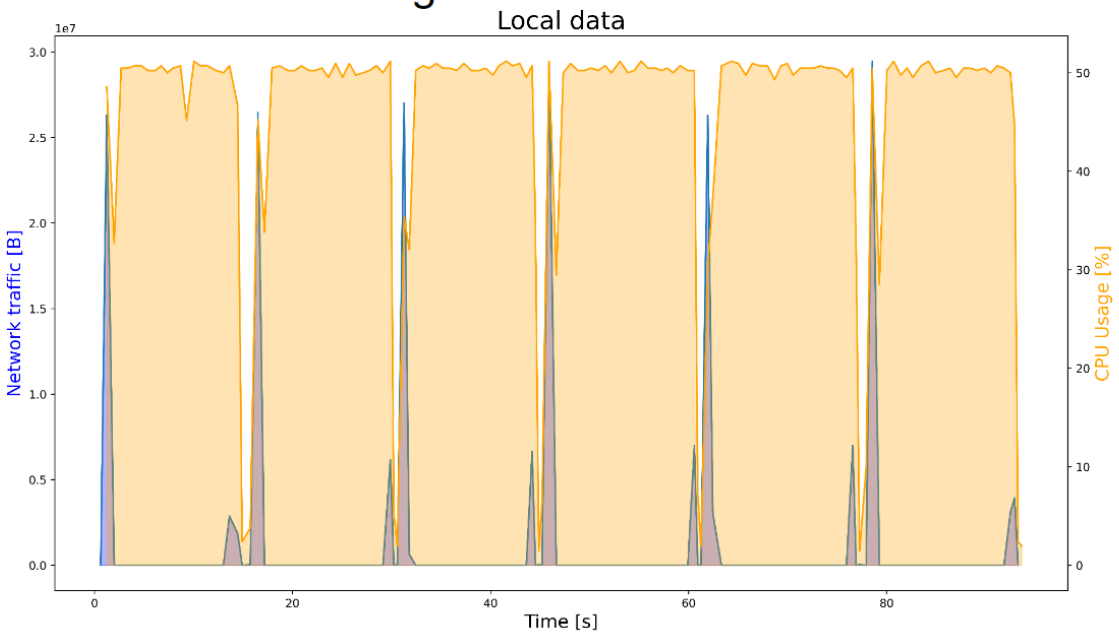
Performed tests:

8, 16, 32, 64, 128, 256, 512 lambdas calculating pollutant averages on 2.5, 5, 10, 20, 40 GB of data.

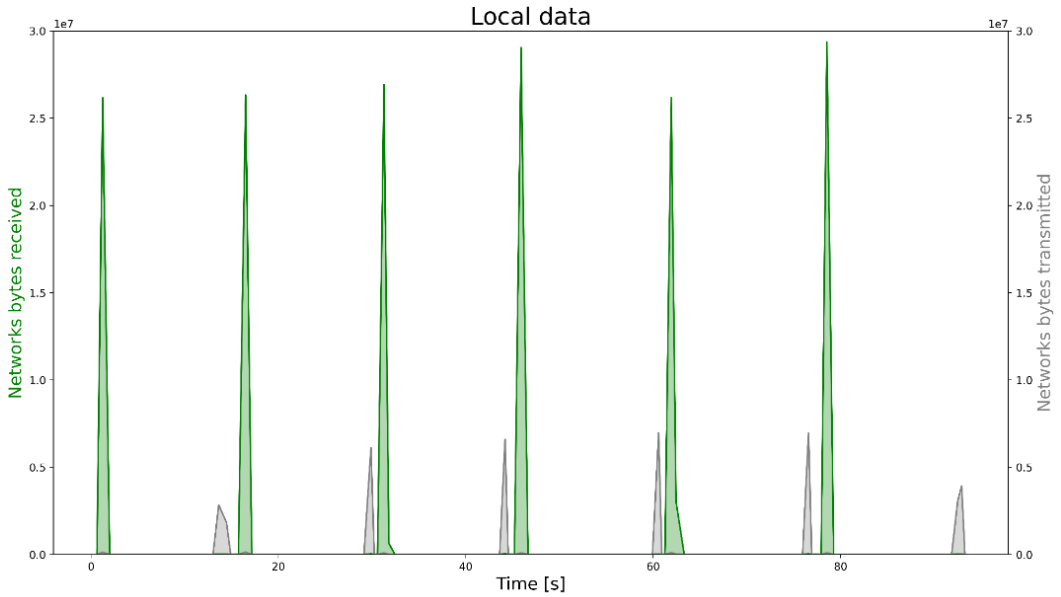
Scalability



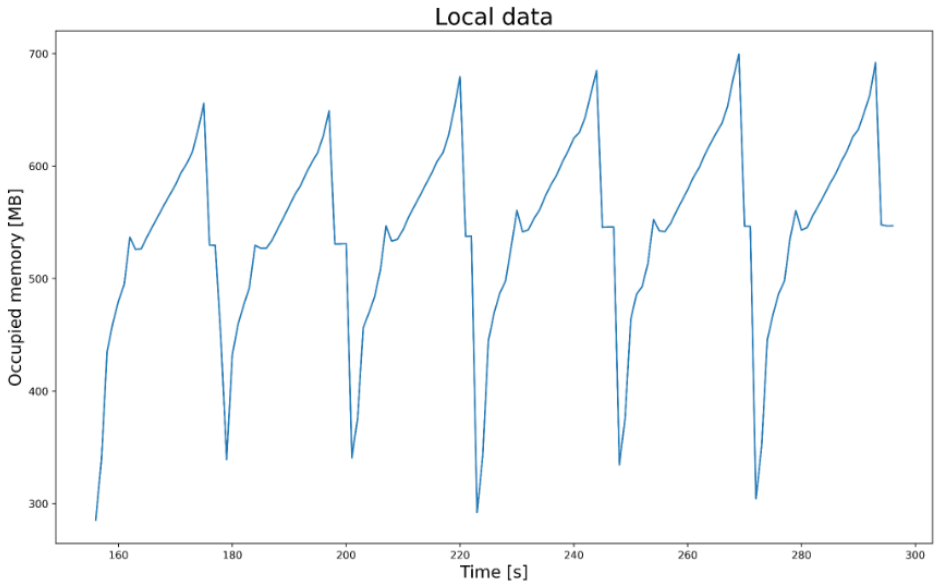
CPU and network usage

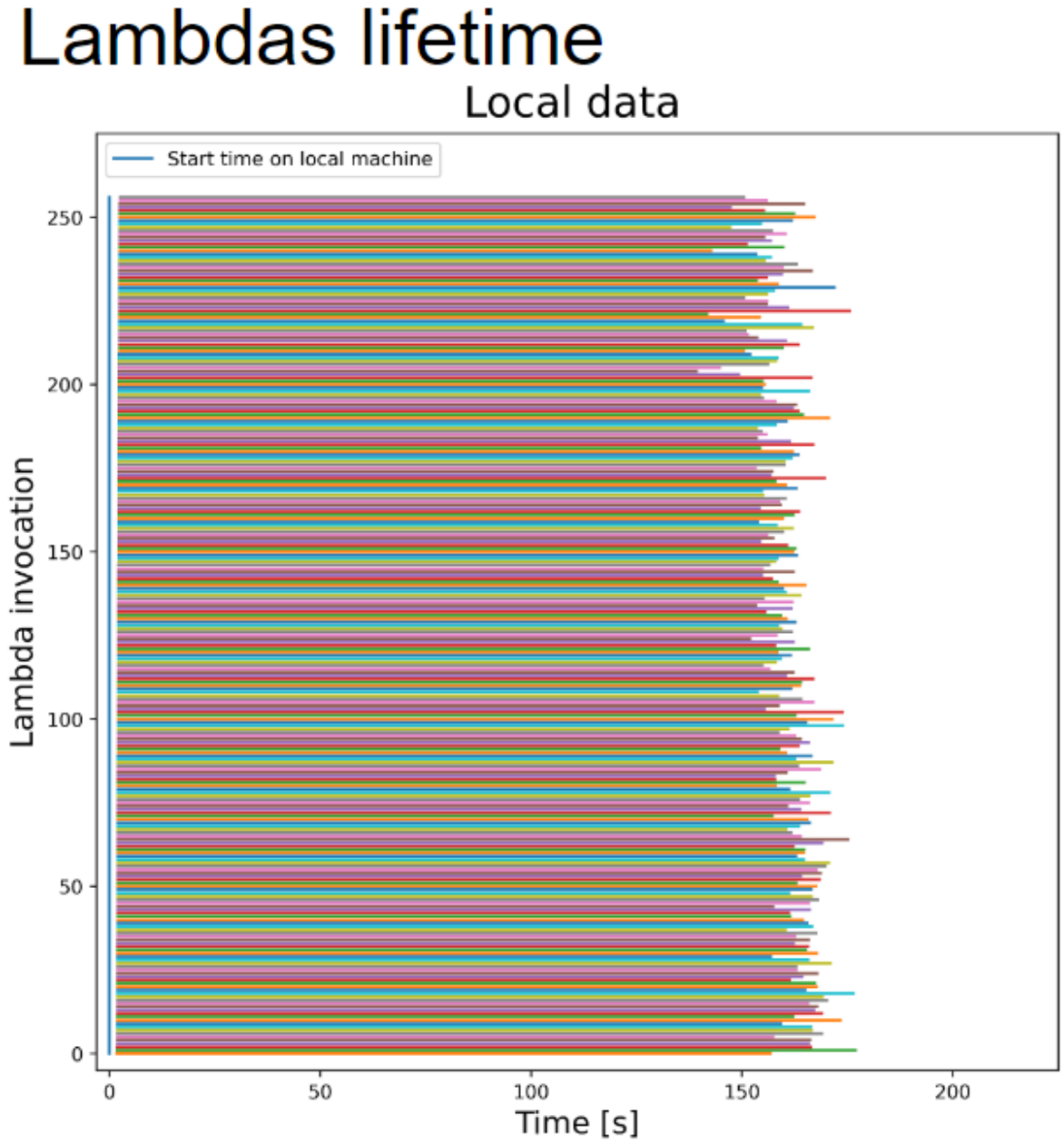


Network transmission



Occupied memory





The cumulative cost of the whole testing was about 150 $.