Introduction to MapReduce: Takeaways

Syntax

- MapReduce framework:

```
import math
import functools
from multiprocessing import Pool
def make_chunks(data, num_chunks):
    chunk_size = math.ceil(len(data) / num_chunks)
    return [data[i:i+chunk_size] for i in range(0, len(data), chunk_size)]
def map_reduce(data, num_processes, mapper, reducer):
    chunks = make_chunks(data, num_processes)
    with Pool(num_processes) as pool:
        chunk_results = pool.map(mapper, chunks)
    return functools.reduce(reducer, chunk_results)
```

- Calculating the maximum value in a list with MapReduce with eight processes:

```
maximum = map_reduce(values, 8, max, max)
```

Concepts

- MapReduce is a programming model for parallel data processing.
- In the MapReduce framework, a user provides a mapper and a reducer function. The data breaks into chunks, and the mapper function applies in parallel to each chunk. Then the reduces function takes the processes chunk results and combines them into an overall answer.
- We can't solve all problems with the MapReduce framework since some problems require us to look at the entire dataset at the same time. This approach only works with calculations where we can calculate the answer from the answers of the partial chunks of data.

Resources

- MapReduce
- Hadoop MapReduce
- `Pool` object
- `functools.reduce()` function