A Light Weight Queue System

Yihan Song[1], Ali Luo[1], Yongheng Zhao[1]

[1]National Astronomical Observatories, Chinese Academy of Sciences

Introduction

With the developing of the technology, we have more compute resource. We can run our pipeline on a cloud or several computers. We develop a light weight package for distributed computation in python. To use this package, you need not to install any other software. Just after some tiny modification on your code, you will have a distributed version.

User should do:

- Job generator
- A function returns results depending on incoming job as parameter.
- Modify configure files which including server IP and port.
- Start server and lunch clients.

Package does:

- Log messages when job fails.
- Start a web server for monitoring how many job finished and failed.
- Transfer jobs from job generator to each clients.
- One clients can lunch several fork to consume the jobs.
- All transfer are threading safe.

Web Monitoring

Message Queue Web Statistics

Forks	Finished	Machines	Failed	Working	Server Time	Job Time
8	40 | 3560	1	4	20	0:00:15	0:00:05

#	IP	Finished	Failed	Working	CPU (%)	Mem Used(MB)/Free(GB)
1	[1]127.0.0.1/1	4	0	2	32.3	10.25MB | 45.66G
2	[2]127.0.0.1/2	4	0	2	32.3	10.43MB | 45.66G
3	[3]127.0.0.1/3	4	0	2	32.3	10.60MB | 45.66G
4	[4]127.0.0.1/4	4	2	2	32.3	10.48MB | 45.66G
5	[5]127.0.0.1/5	4	0	2	32.3	10.47MB | 45.66G
6	[6]127.0.0.1/6	4	2	2	28.0	10.43MB | 45.66G
7	[7]127.0.0.1/7	4	0	2	43.8	10.34MB | 45.66G
8	[8]127.0.0.1/8	4	0	2	33.3	10.35MB | 45.66G

Job Generator

```
class seed:
    def __init__(self):
        self.i = 0
        self.total = 3560
        pass
    def __iter__(self):
        return self
    def __next__(self):
        self.i += 1
        if self.i >self.total:
            return None
        return self.i
    def __len__(self):
        return self.total
```

Job Process Function

```
import time
class user_main():
    def __init__(self,logger=None):
        self.logger = logger
        self.logging("Hello World")
        pass
    def logging(self,msg):
        if self.logger:
            self.logger(msg)
    def run(self,msg):
        if int(msg) % 11 ==0:
            a = 1/0.0
        time.sleep(1)
        return 'done'
    def final(self):
        pass
```

Flow Diagram