PiCloud: Cloud Computing Simplified

A hands-on quick overview

Amit Saha
PyCon Australia'12
@echorand

August 17, 2012

About Me

- PiCloud (and Python) enthusiast
- Freelance Technical Writer
- Fedora project contributor: Scientific Spin, Google Summer of Code, etc.
- Blog: `http://echorand.me`

Outline

What is PiCloud?

- Cloud Computing solution
- Primarily acessible via Python library: import cloud
- Commercial offering, but has 20 free core hours/month for all

Key Features

- **Automated Deployment:** Your code (along with modules) magically transported

Key Features

- **Automated Deployment:** Your code (along with modules) magically transported
- **Choice of computing power**: Cores of different capabilities
- **Scientific Computing ready**: SciPy and NumPy (others can be installed)
- REST APIs, Environments, Cron Jobs, S3 Storage ..

Easy Setup

- Register at
  `https://www.picloud.com/accounts/register/`
- $sudo pip-python install cloud
- $ picloud setup (Your email address)
- Gives you an API key (keep it safe)

Sanity check

(IPython notebook available in the slide repository)

```
def square(x):
    return x*x

# demonstration of cloud.call()
import cloud
jid = cloud.call(square,3)

print 'Job Id::', jid
print 'Job status',cloud.status(jid)

print 'Result',cloud.result(jid)
```

Sanity Check: Demo

DEMO

Sorting in the Cloud

```
import cloud
import numpy

def sort_num(num):
    sort_num = numpy.sort(num)
    return sort_num

if __name__ == '__main__':
    num=numpy.random.random_integers(10,10000,50000)
    jid=cloud.call(sort_num, num)
    print cloud.result(jid)
```

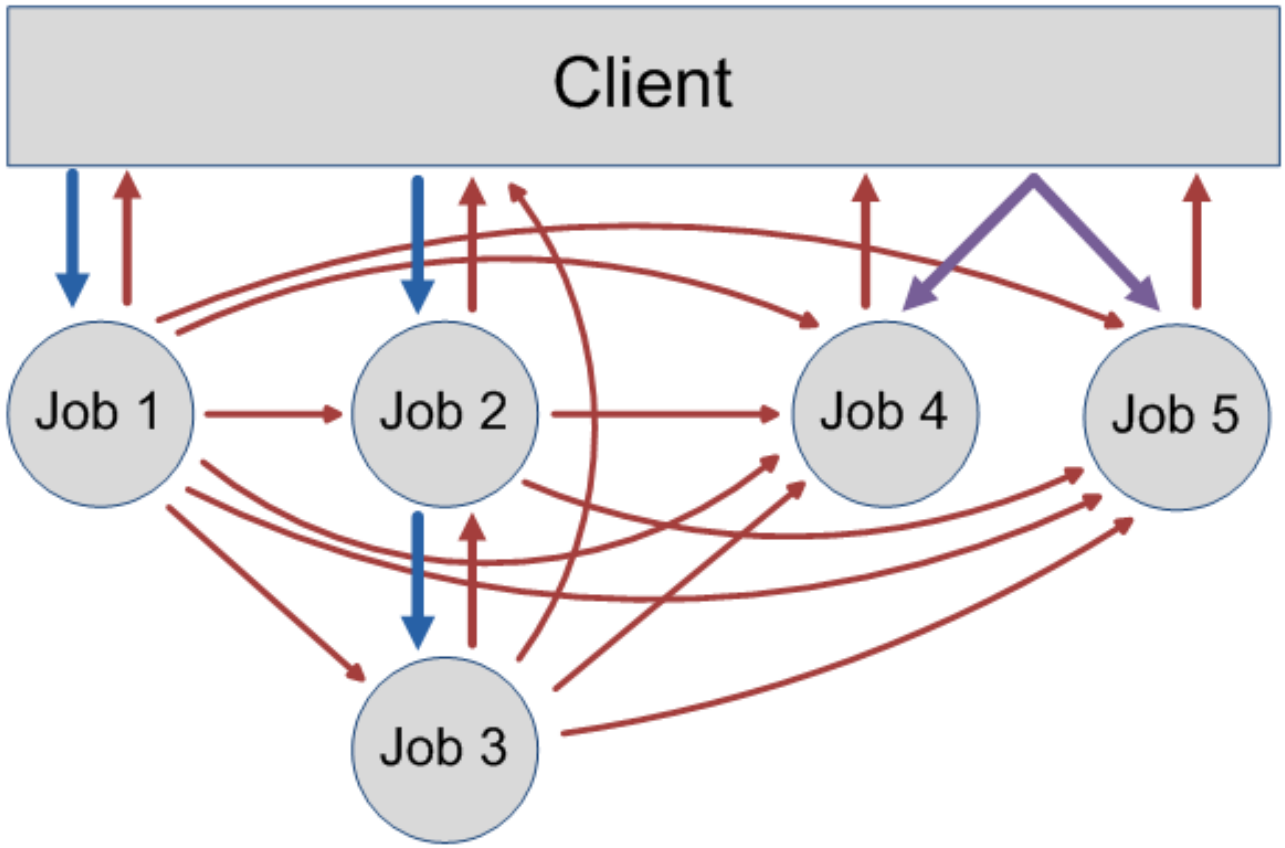
A PiCloud decorator

```
def cloudcall(func):
    def sendtocloud(*args, **kwargs):
        import cloud
        jid = cloud.call(func,*args,**kwargs)
        cloud.join(jid)
        print 'Result:: ', cloud.result(jid)

    return sendtocloud
```

Using the decorator

```
@cloudcall
def anexpensivefunction(x,y):
    return x**3 + y**3

if __name__=='__main__':
    anexpensivefunction(3,3)
```

A messy graphic



Source:

`http://docs.picloud.com/tech_overview.html#take-away`

Moving persistent data

- cloud.files module
- Store a file: cloud.files.put()
- Retrieve a file: cloud.files.get()
- List all files: cloud.files.list()

Persistent data demo

```
import cloud
def savedata():
    f=open('data.txt','w')
    f.write('This is a line of text')
    f.close()

    cloud.files.put('data.txt')

# Interpreter session:
# In [7]: import cloud
# In [8]: cloud.files.list()
# Out[8]: []
# In [9]: from picloud_filedemo import *
# In [10]: cloud.call(savedata)
# Out[10]: 107
# In [11]: cloud.files.list()
# Out[11]: [u'data.txt']
```

Evolutionary Algorithms in the Cloud

- Pyevolve: Python Evolutionary Algorithm library
- More than one ways to parallelize
- Execute the algorithm on PiCloud infrastructure
- For more information: `http://pyevolve.sourceforge.net/`

Pyevolve + PiCloud

```
from pyevolve_rastrigin import *
import cloud

# assuming 10 runs
seed_list=[100*(i+1) for i in range(10)]
runid_list=[i+1 for i in range(10)]

# calls the method defined in pyevolve_rastrigin.py
# which initiates the GA execution.
# Execute the code on PiCloud
jids = cloud.map(run_ga,seed_list,runid_list)

# pull the stat files
cloud.join(jids)
print cloud.files.list()
for i in range(10):
    cloud.files.get('stats_' + str(i+1) + '.csv','stats_' +
        str(i+1)+'.csv')
```

Automatic Deployment + Moving files

DEMO

REST API

- Publish your functions via a REST API
- Language independent access to your Python functions
- Most of the cloud library functions have REST analogs

REST API: Publishing a function

```
>>> def square(x):
        return x*x
>>> import cloud
>>> uri=cloud.rest.publish(square, "square_func")
>>> print uri
https://api.picloud.com/r/3222/square_func
```

REST API: Invoking the published functions

- Using the end-point, first get the job ID
- Use this Job ID to get the result
- Appropriate requests can be made using curl or any other client capable of invoking REST APIs

Environments

- Non-Python software packages
- Python libraries with native-dependencies (not already installed)
- Your personal sandbox in PiCloud (based on Ubuntu Linux)
- Specify environment to execute your code in

Job Monitoring and Management

- Get information about a job: cloud.info()
- Kill, Delete jobs cloud.kill(), cloud.delete()
- Web interface has functions for managing your jobs, crons, analytics, payment ..
- Manage your account using cloud.account module

Summarize

- A truly simple way to harness and play around with cloud computing
- Code is there to explore
- Simulator to help you test your code
- Just go ahead and import cloud!

Resources

- PiCloud Homepage: `http://www.picloud.com`
- Technical Overview: `http://docs.picloud.com/tech_overview.html`
- PiCloud Pitfalls: `http://docs.picloud.com/client_pitfall.html`
- PiCloud Documentation: `http://docs.picloud.com/`
- PiCloud FAQ: `http://www.picloud.com/faq/`
- PiCloud: An Easy Way to the Cloud: `http://bit.ly/GZDxZB`
- Presentation and Code: `https://github.com/amitsaha/picloud-preso`