Implementación de un sistema de cómputo distribudo Map-Reduce sobre AMQP

Juan Bautista Cabral [1],
[1] IATE-OAC;

El advenimiento de grandes volúmenes de datos (o Big Data) esta generando una necesidad de productos que sirvan para la manipulación y resumen de los mismos. Big Data puede considerarse de manera mas o menos precisa (no es mas que un numero comercial) como una cantidad de información tal que no puede procesarse ni almacenarse en un único ordenador.

Las dificultades más habituales vinculadas a la gestión de estas cantidades de datos se centran en la captura, el alma- cenamiento, búsqueda, compartición, análisis y visualización. La tendencia a manipular ingentes cantidades de datos se debe a la necesidad en muchos casos de incluir los diferentes conjuntos de datos relacionados.

La tendencia actual es el almacenamiento y el procesamiento a través de nodos distribuidos en una red de la manera mas transparente posible para el programador, haciéndolo parecer que esta ejecutando todo localmente; despegándose un poco del modelo propuesto por el ya tradicional modelo distribuido de MPI (del inglés Interfaz de Paso de Mensaje) de hacer evidente la no localidad del computo.

Por el dado del análisis de datos; en la presencia de una cantidad ingente de información como la que planteamos hace necesario el disponer de mecanismos automáticos para el procesamiento de estos volúmenes. Es en este campo donde una herramienta como el aprendizaje automático (o ML) obtiene un valor de piedra angular. El aprendizaje automático es una rama de la de la Inteligencia Artificial que consiste en crear programas que buscan de manera autónoma patrones en la información a partir de ejemplos.

Objetivos

Los objetivos del trabajo fueron reconocer la distribución geográfica y establecer en cada región del país el nivel de inóculo que podría relacionarse con el desencadenamiento de la enfermedad en la campaña agrícola siguiente, así como comparar el comportamiento entre las dos enfermedades.

Arquitectura

Resultados y discusión

Ejemplo: Random Forest Sobre *Iris.arff*

```
from poopy import script

class Script(script.ScriptBase):

    def map(self, k, v, ctx):
        import random

        import numpy as np
        import scipy

        from sklearn import tree

        attrs = ['sepallength', 'sepalwidth', 'petallength', 'petalwidth']
        random.shuffle(attrs)
        attrs.pop()

        data, meta = v

        target = np.array(data['class'])
        train = np.array(data[attrs][:75])
        X = np.asarray(train.tolist(), dtype=np.float32)
        dt = tree.DecisionTreeClassifier(
            criterion='entropy', max_features="auto", min_samples_leaf=10)
        ctx.emit(None, dt)

    def reduce(self, k, v, ctx):
        for vi in v:
            ctx.emit("iris", vi)

    def setup(self, job):
        job.name = "Random Forest"
        job.input_path.append(["poopyFS://iris.arff",
                               self.readers.ARFFReader])
```

Para correr este Script usted debe isntall rabbit-mq y poopy (**pip install poopy**), y luego:

Conclusiones

Contacto: jbc.develop@gmail.com