Unidad 06
Análisis y geoprocesamiento con Python

Curso de geoprocesamiento de datos con Python
2016

Cayetano Benavent Viñuales
Analista GIS en Geographica
cayetano.benavent@geographica.gs

Unidad 06 - Sumario de contenidos

GEOS - Geometry Engine, Open Source

GEOS Geometry Engine Open Source

- GEOS permite realizar análisis espacial complejo sobre datos vectoriales. Implementa los predicados espaciales del Simple Features Specification for SQL definido por la OGC, además de otras funciones topológicas avanzadas.

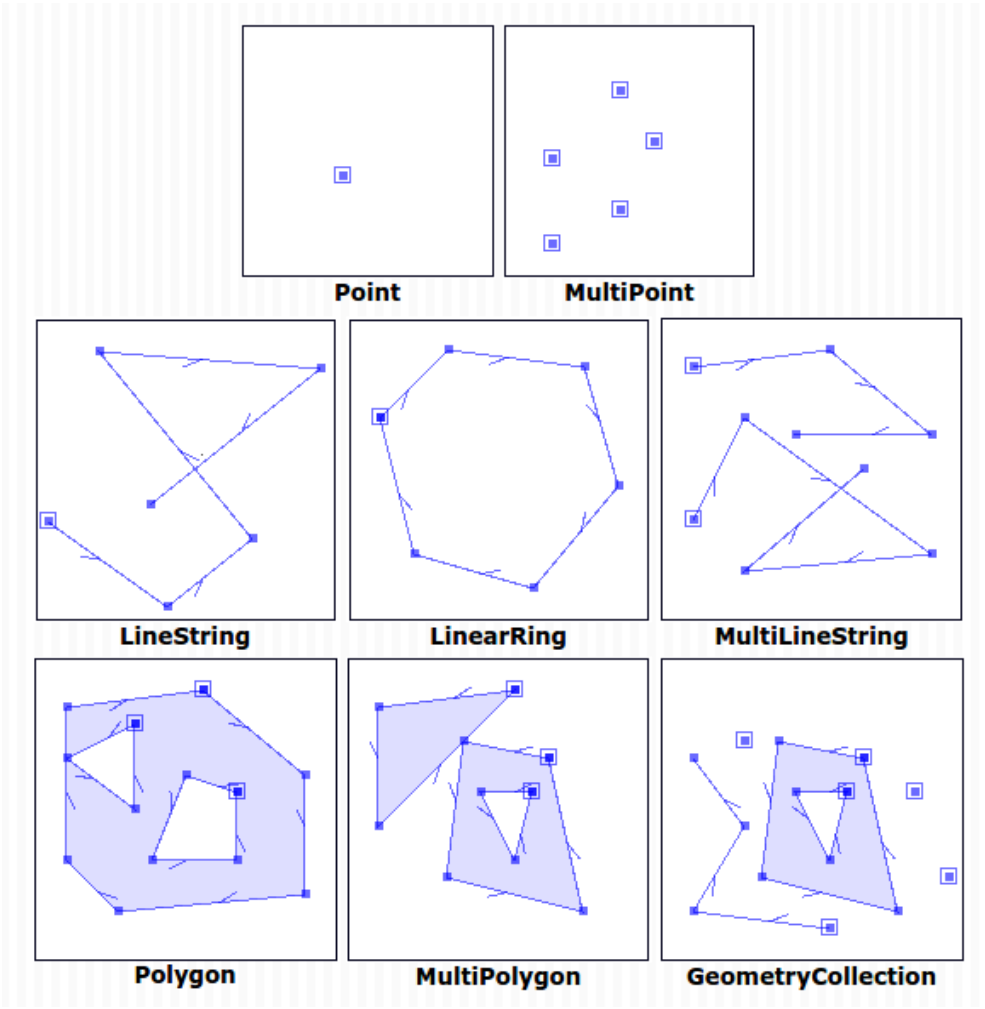
- Es un port a C++ de la librería JTS (Java Topology Suite).

- GEOS es uno de los proyectos principales de OSGeo, y está liberado bajo la licencia GNU LGPL.

- La página del proyecto: https://trac.osgeo.org/geos/

Análisis espacial vectorial: GEOS

Spatial Data Model



Fuente: Vividsolutions.

GEOS Geometry Engine Open Source

Capacidades principales de GEOS

- *Geometries: Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, GeometryCollection*
- *Predicates: Intersects, Touches, Disjoint, Crosses, Within, Contains, Overlaps, Equals, Covers*
- *Operations: Union, Distance, Intersection, Symmetric Difference, Convex Hull, Envelope, Buffer, Simplify, Polygon Assembly, Valid, Area, Length,*
- *Prepared geometries (pre-spatially indexed)*
- *STR spatial index*
- *OGC Well Known Text (WKT) and Well Known Binary (WKB) encoders and decoders.*
- *C and C++ API (C API gives long term ABI stability)*
- *Thread safe (using the reentrant API)*

GEOS - Geometry Engine, Open Source

GEOS Geometry Engine Open Source

- GEOS está actualmente (marzo 2016) en su versión 3.5.

- Es multiplataforma, y tienes dos API oficiales: C (recomendada) y C++.

- GEOS es un pilar clave del geo open source, formando parte de muchas aplicaciones:

  QGIS, GDAL, PostGIS, GRASS GIS, MapServer, SAGA GIS, etc.

¿Python Binding?

GEOS Geometry Engine Open Source

• No hay actualmente (antes sí lo había) un binding de Python dentro de GEOS. Desde el proyecto indican cómo usar GEOS desde Python:

Since version 3.0 the Python bindings are no longer supported.
Options:

1) Become or recruit a new maintainer.
2) Use Shapely (http://pypi.python.org/pypi/Shapely) on Python 2.4+
3) Simply call functions from libgeos_c via Python ctypes.

https://trac.osgeo.org/geos/ticket/228

Shapely

- La opción preferente para usar GEOS en Python es la librería Shapely.

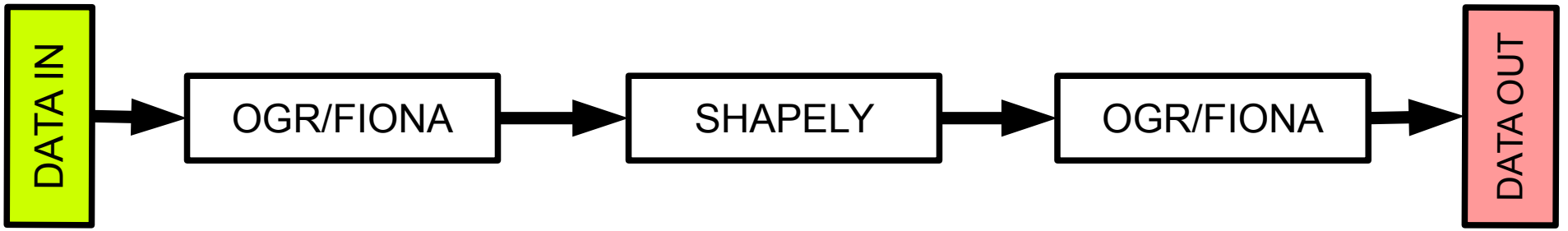
- Es una interfaz madura, estable, muy optimizada (con Cython) y muy Pythónica.

- Nos permite hacer operaciones “PostGIS-like” sin tener que tener una RDBMS y lo más importante, combinar GEOS con el stack científico de Python (Numpy, Scipy, etc.).

Análisis espacial vectorial: Shapely

- Shapely no maneja (ni se preocupa) de formatos de datos; únicamente trabaja con objetos geométricos.

- Es por ello que para diseñar flujos de geoprocesamiento, necesitamos que los data input y output (a distintos formatos) los gestione GDAL (OGR) y/o Fiona.



GDAL (OGR)

- Desde OGR también podemos realizar operaciones de análisis vectorial.

- Realmente, las funciones de análisis avanzado las proporciona en OGR la librería GEOS.

- Podemos, por tanto, realizar también desde OGR muchas de las operaciones clásicas de GEOS.

Análisis espacial vectorial: GDAL (OGR)

Para entender mejor el análisis vectorial con GDAL (OGR), realizaremos un ejercicio práctico siguiendo el Notebook unit06_01.

Análisis espacial vectorial: Fiona

Para entender la reproyección de datos vectoriales con Fiona, realizaremos un ejercicio práctico siguiendo el Notebook unit06_02.

Análisis espacial vectorial: Shapely

Para entender el análisis vectorial con Shapely, realizaremos un ejercicio práctico siguiendo el Notebook unit06_03.

Análisis espacial vectorial: Shapely

Para entender el análisis vectorial con Shapely y NumPy, realizaremos un ejercicio práctico siguiendo el Notebook unit06_04.

Álgebra de mapas

Es el lenguaje de análisis ráster por excelencia.

Permite realizar análisis y modelado de la información geográfica a través de expresiones algebraicas (no hay que olvidar que una capa ráster está formada por celdas, cada una de las cuales tiene un valor numérico con el que podemos operar algebraicamente).

Álgebra de mapas

GDAL y/o Rasterio + NumPy

Una vez hemos abierto un fichero raster como Dataset en GDAL y/o Rasterio, lo que hace tremendamente potente a Python, es poder leerlo como Numpy Array.

Álgebra de mapas

GDAL y/o Rasterio + NumPy

Todo el poder de Numpy en nuestras manos para hacer cómputos vectorizados muy rápidos con el Dataset raster.

Obviamente, ello hace que el Dataset esté expuesto a todo el Stack científico de Python (Scipy, Matplotlib, Scikits, etc.).

Tras operar con Numpy con el Dataset, podemos escribir a un formato raster con GDAL el resultado.

Análisis espacial raster: GDAL

Para entender mejor el análisis raster con GDAL y NumPy, realizaremos un ejercicio práctico siguiendo el Notebook unit06_05.

Análisis espacial raster: Rasterio

Para entender mejor el análisis raster con Rasterio y NumPy, realizaremos un ejercicio práctico siguiendo el Notebook unit06_06.

Para entender mejor la reproyección de datos raster con Rasterio, realizaremos un ejercicio práctico siguiendo el Notebook unit06_07.

Bibliografía

Selección bibliográfica:

- Westra, Eric (2013): “Python Geospatial Development (Second Edition)”. Packt publishing.

- Garrard, Chris (2016): “Geoprocessing with Python”. Manning.