

```
require( 'TEMPLATEPATH.DS.'yjscore/yjs_stylew.php');
$render = $document->loadRenderer( 'module' );
$options = array( 'style' => "raw" );
$module = JModuleHelper::getModule( 'mod_menu' );
$stopmenu = false; $subnav = false; $sidenav = false;
Main Menu
if ( $default_menu_style == 1 or $default_menu_style == 2 ) :
    $module->params = "menutype=$menu_name\nshowAllChildren=1\nclass_grow
    $stopmenu = $render->render( $module, $options );
    $menuclass = 'horiznav';
    $stopmenuclass = 'top_menu';
elseif ( $default_menu_style == 3 or $default_menu_style == 4 ) :
    $module->params = "menutype=$menu_name\nshowAllChildren=1\nclass_grow
    $stopmenu = $render->render( $module, $options );
    $menuclass = 'horiznav_d';
    $stopmenuclass = 'top_menu_d';
SPLIT MENU NO SUBS
elseif ( $default_menu_style == 5 ) :
    $module->params = "menutype=$menu_name\nstartLevel=$startLevel
    $stopmenu = $render->render( $module, $options );
    $menuclass = 'horiznav';
    $stopmenuclass = 'top_menu';
```



Solución al TSP con Python

Ing. Juan Camilo Correa Chica

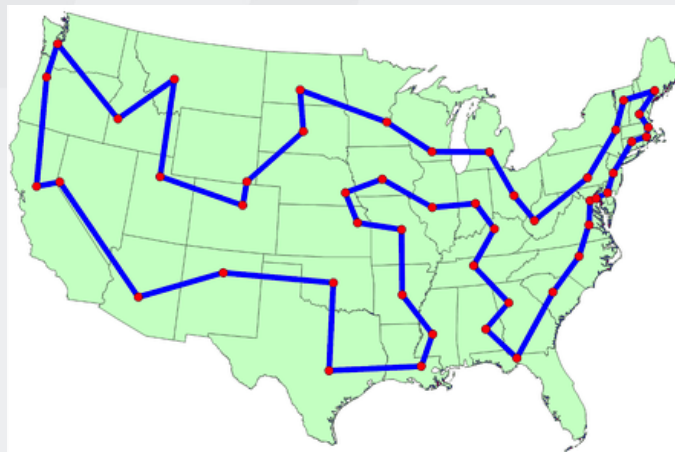
Solución al Travel Salesman Problem con Python

- Travel Salesman Problem - TSP
- Heurística constructiva: Método de inserción
- Heurística de mejoramiento: Movimientos K-opt

Travel Salesman Problem

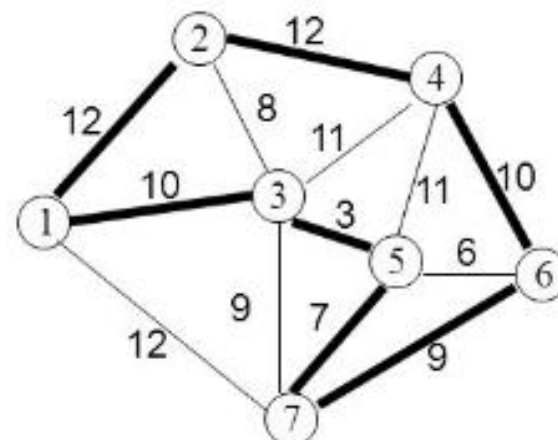
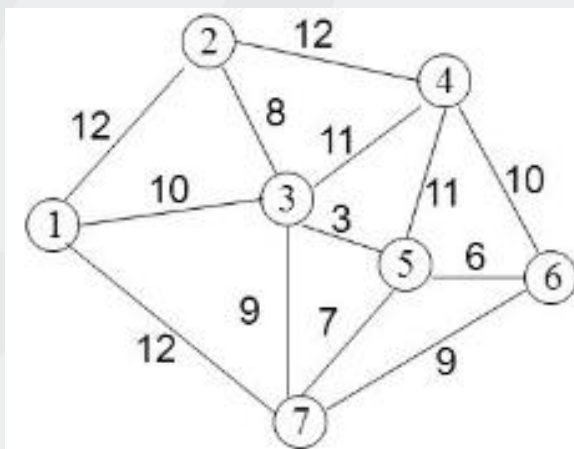
El problema del vendedor viajero (TSP por sus siglas en inglés) es uno de los problemas mejor conocidos en el campo de la investigación de operaciones, ciencias de la computación y matemáticas. La idea básica del problema es la de tratar de encontrar en ciclo más corto en una red tal que todos los nodos de la red sean visitados y que la distancia viajada sea mínima. De manera más técnica se puede enunciar así:

Se tiene una red $G=[N, A, C]$ que está definida por el número N de nodos, el número A de enlaces entre los nodos y $C=[c_{ij}]$ la matriz de costos donde c_{ij} es el costo de moverse desde el nodo i hacia el nodo j . El problema del vendedor viajero se soluciona con un ciclo Hamiltoniano en la red G con el costo mínimo. Un ciclo Hamiltoniano es un ciclo que pasa por cada nodo i de los N nodos de la red exactamente una vez.



Travel Salesman Problem

Empezando desde la ciudad 1 (nodo 1), el vendedor debe viajar a todas las otras ciudades (nodos) antes de retornar de nuevo al punto de partida. La distancia entre cada ciudad con respecto a las otras está dada y se asume que es igual independientemente de la dirección. El objetivo es minimizar el costo o distancia total que se debe viajar, un recorrido optimo es aquel en el que no se crean sub-ciclos cerrados y no hay rutas que se entrecrucen.



Travel Salesman Problem

Muchos algoritmos se han propuesto para la solución de este problema. Un proceso de solución típico tiene varias etapas, donde un “tour” inicial se construye y luego se insertan los puntos que quedaron por fuera. Una vez se tienen todos los nodos en el tour se procede a ir mejorando el tour hasta llegar al valor óptimo de la solución.

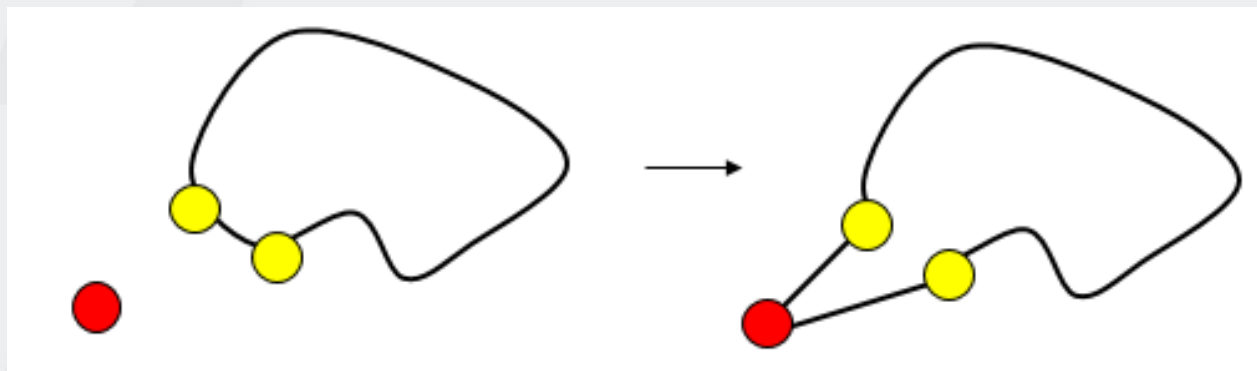
1. Crear un sub-tour inicial: Algoritmos de inserción.
2. Insertar los nodos restantes: Algoritmos de inserción.
3. Mejorar el tour existente: Movimientos 2-opt y 3-opt, y swapping.

Algoritmos de inserción: Inserción del más cercano, inserción del más lejano, inserción del más barato, inserción del más costoso.

Inserción del más cercano: El tour comienza con cualquier nodo de la red, luego se busca el nodo más cercano y se conecta al último nodo que se haya agregado al tour, este paso se repite hasta que todos los nodos se hayan insertado al tour y se conecta el último nodo insertado al primero con el que se comenzó el tour.

Travel Salesman Problem

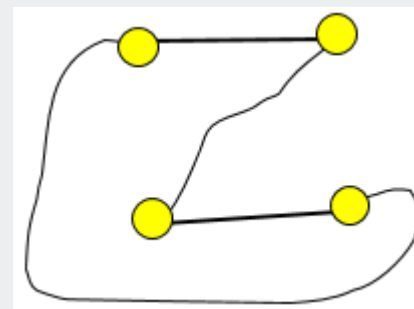
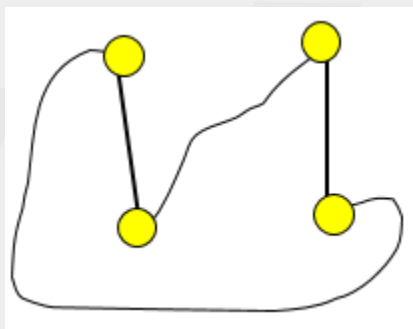
Inserción del más lejano: El tour comienza con cualquier nodo de la red, luego se busca el nodo más lejano y se conecta al último nodo que se haya agregado al tour, este paso se repite hasta que todos los nodos se hayan insertado al tour y se conecta el último nodo insertado al primero con el que se comenzó el tour.



Mejoramiento de la solución

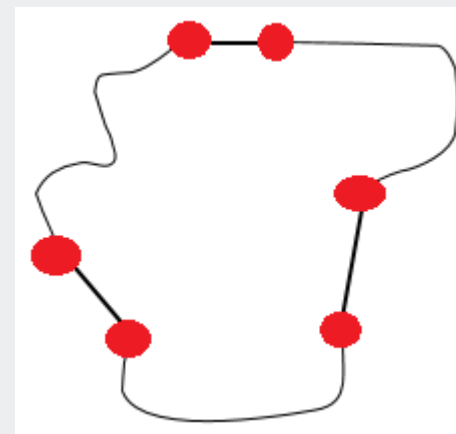
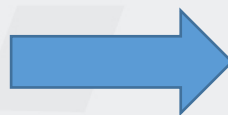
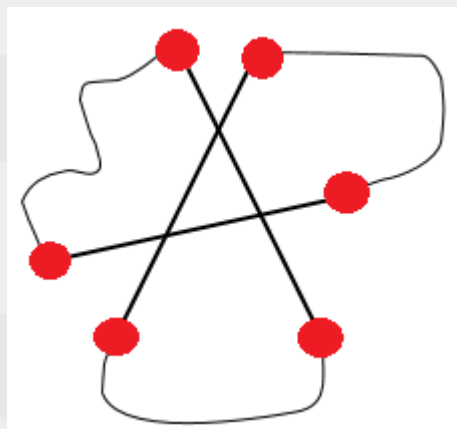
Para mejorar la solución, se pueden emplear movimientos K-opt, con los que se busca minimizar el costo final del tour mediante el rediseño y el intercambio de arcos (vértices) que hacen parte de la solución actual. Los movimientos K-opt ayudan a eliminar arcos que se cruzan y a mejorar la solución ya que encuentra nuevos arcos con costos inferiores a los de la solución original.

Movimiento 2-opt: Se toman dos arcos (vértices), suponga (v, w) y (x, y) y se modifica el tour formando nuevos arcos, sea (v, x) y (w, y) o bien (v, y) y (w, x) .



Mejoramiento de la solución

Movimiento 3-opt: En este caso se escogen 3 arcos (vértices) del tour y los nodos pertenecientes a ellos se reconectan para formar nuevos arcos que mejoren el costo de la solución original



Problema

En GitHub se encuentra un archivo de Python (TSP.py) con 3 instancias del problema del vendedor viajero, utilice una heurística constructiva (método de inserción) y una heurística de mejoramiento (movimiento K-opt) para encontrar una solución lo más óptima posible (tour con la menor distancia).

Enlaces útiles TSP

https://www2.isye.gatech.edu/~mgoetsch/cali/VEHICLE/TSP/TSP003__.HTM

<https://docs.mosek.com/6.0/capi/node015.html>