Resolución problema 3

## Punto 3

## Comparación de las corridas

### TMZ

El primer modelo, el TMZ es el modelo del problema del viajante común que conocemos y vemos en la materia. En el problema se plantean actividades o cosas de las cuales no se conoce el orden en el cual se realizan. La función objetivo (directa o indirectamente) depende del orden que indique el modelo para esas actividades (a distinto orden, distinto resultado).

Yij si el tour va directo desde la ciudad i a la j, 0 en otro caso

El objetivo es minimizar donde Cij es la dist. entre sucursales i y j

2 condiciones obvias:

* Exactamente una sucursal debe ser visitada inmediatamente después de la ciudad i
* Exactamente una sucursal debe ser visitada inmediatamente antes de la ciudad j
* Eso se expresa:
* Yij = 1 Para todo i = 1, ..., 100 i <> j
* Yij = 1 Para todo j = 1, ..., 100 i <> j

Este va probando todas las posibilidades una por una de forma secuencial. Entonces podemos sacar de conclusión que a medida que el número de sucursales vaya en aumento el tiempo de solución crece mucho más rápido, lo que hace prever que esta heurística es muy buena cuando el número de sucursales es pequeño, pero no así cuando el número de nodos es grande. Por lo tanto, ***no es recomendable el uso modelo MTZ cuando el número de nodos es muy grande pues podría perder eficacia en el tiempo de solución.***

### Eliminación Subtours

El segundo modelo, es un modelo que elimina subtours y no va probando de forma secuencial cada caso, si no que tiene "ordered" esto hace que solo tenga Xij cuando i < j, limitando muchísimo la cantidad de casos a probar para encontrar la solución óptima.

Esto también genera que no importa qué tipo de lista le pasemos, la performance del algoritmo va a ser siempre muy similar. Esto se verá más adelante con la heurística de mejora.

El número de restricciones del modelo es 2|N| (siendo N=100 sucursales) más el número de subtours que es ***exponencial***. Por esta razón, el modelo no puede ser

utilizado directamente, a menos que el número de nodos sea pequeño. La idea básica sería agregar al modelo solo aquellas restricciones de eliminación de subtours que son realmente necesarias. Por esto, es que en este caso puede usarse este modelo.

Los beneficios que trae esto es que, supongamos que eliminamos todas las restricciones de eliminación de subtours, el problema podría ser resuelto de forma eficiente. Pero, la solución que obtenemos podría ser ***inservible***, ya que podría contener subtours (si no tuviera, sería la solución óptima). En resumen, si NO eliminamos a los subtours, la solución podría ser la ***óptima*** y **NO** contener ningún subtour, ó podría contener subtours y ***no servir***.

Este modelo te ofrece que la solución encontrada va a ser necesariamente la óptima.

## Gráfico de las corridas

Acá tenemos el gráfico de la corrida del TMZ que tardó 5:53 minutos en total para encontrar la solución.

Gráfico

Descripción generada automáticamente

Por otro lado, este es el gráfico del modelo de eliminación de subtours que tardó 19:85 segundos! Mucho más rápido.

Gráfico

Descripción generada automáticamente

Podemos ver que el tiempo es muy distinto entre uno y el otro. Esto es porque el modelo de eliminación de subtours es más optimizante a la hora de encontrar una solución.

## Punto 5

## Gráfico de las corridas con la heurística de mejora

**Gráfico TMZ con heurística de mejora**

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Le tomó 3:33 minutos para encontrar la solución. Podemos apreciar que tardó muchísimo menos con esta heurística de mejora. Esto se debe a que le estamos proporcionando el mejor caso posible, es decir, le estamos entregando una pre-solucion al problema, entonces le cuesta menos encontrarlo. Tanto menos que me atrevo a decir que le cuesta **aproximadamente** la mitad del tiempo.

**Gráfico eliminación subtours con heurística de mejora**

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Le tomó 19:24 segundos. En este caso vemos que la diferencia no es considerable. Esto se debe a la forma en la que encuentra la solución. Al ser tan óptima e ir buscando por subtours, si le proporcionamos una lista con una pre-solución o una lista con el peor caso posible, al algoritmo no le va a cambiar mucho en la performance, ya que va a ir buscando los mismos subtours y los va a ir eliminando de la misma manera.

En este caso fue encontrando soluciones intermedias (puntos amarillos), pero su performance se mantuvo.