Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería de Transporte y Logística

Junio, 2020.

TAREA 2

# La ruta más corta

## ICT-2233 Flujo en Redes

Prof. Mathias Klapp
Ayudante: Pablo Seisdedos (pcseisdedos@uc.cl)

En esta tarea su grupo estudiará distintos algoritmos para determinar rutas mínimas entre entre un conjunto de orígenes $S$ y un conjunto de destinos $D$ en un grafo dirigido. Para esto contarán con una base de datos que posee el posicionamiento geográfico de 264.346 intersecciones de calles en la ciudad de Nueva York junto con 730.100 calles que conectan dichas intersecciones, cada una con su respectivo tiempo de traslado y distancia. Esta información le permitirá a su grupo construir un grafo de la red tal como se detalla en la siguiente figura. Específicamente, en la base de datos entregada usted contará con los siguientes tres archivos:

- **coordenadas.txt**: Contiene las coordenadas geográficas $(x, y)$ de todas las intersecciones de calles en la ciudad.
- **distancias.txt**: Contiene las distancias de las calles que unen intersecciones en la red.
- **tiempos.txt**: Contiene los tiempos de traslado por las calles que unen intersecciones de la red.


Imagen 1: Ejemplo grafo dirigido de calles e intersecciones.
Fuente: https://arodriguez.blogs.upv.es/showtime-3-las-obras-y-los-grafos/

Esta tarea se estructura en tres partes descritas a continuación.

## Parte I: Implementación de rutinas de ruta mínima a nivel de usuario (20 puntos):

En esta sección su grupo deberá resolver problemas de ruta mínima utilizando los algoritmos disponibles en la librería NetworkX. Específicamente, el grupo debería resolver dos problemas de ruta mínima (en distancia y tiempo de viaje) desde cada intersección en un conjunto $S$ hacia cada intersección en un conjunto $D$, ambos conjuntos estarán detallados en los archivos **origenes.txt** y **destinos.txt** respectivamente.

Se solicita que el grupo ejecute los siguientes métodos:

1. Bellman-Ford
2. Dijkstra
3. Dijkstra Bidireccional

Se dará un bono por ejecutar el algoritmo A* disponible en Networks desarrollando una estimación adecuada para el costo de ruta hacia adelante que garantice optimalidad.

Analice el desempeño y los resultados de cada método. Debe considerar tiempo de ejecución, uso de memoria y complejidad computacional de su programa completo, incluyendo los algoritmos llamados en NetworkX ¿Qué algoritmo recomendaría para resolver el problema?, ¿por qué?

## Parte II: Diseño de algoritmos (20 puntos)

Ahora, su grupo deberá diseñar y programar su propio algoritmo. Para ello, debe escoger un algoritmo que estime conveniente (RM por OT, Dijkstra, Bellman-Ford, D'Esopo-Pape, Floyd-Warshall, etc..), hacerle los cambios que considere necesarios, programarlo y resolver el problema de la parte I con su propio algoritmo.

Se espera que:

1. El grupo justifique muy bien su elección basándose en argumentos de diseño de algoritmos estudiados en el curso como complejidad computacional y uso de memoria.
2. Sea un algoritmo competitivo con los algoritmos empaquetados de Networkx.
3. El algoritmo entregue la solución óptima.
4. Analice y compare los resultados obtenidos.

Se premiará el uso de soluciones sofisticadas (Heap de Fibonacci o Binarios, por ejemplo).

## Parte III: Rutas Restringidas (20 puntos):

Finalmente, el grupo deberá volver a resolver el problema de ruta mínima anterior, pero considerando las siguientes restricciones:

1. **No se puede doblar a la izquierda en intersecciones.** Cada grupo deberá desarrollar un método que prediga de la mejor manera si es que en una intersección se dobla o no a la izquierda.
2. **No se puede doblar a la izquierda e inmediatamente a la derecha en dos intersecciones seguidas (pero si por separado)**. La idea es evitar que su algoritmo ``culebree''.
3. **Se permite pasar solamente por 500, 650 y 850 intersecciones en la ruta.** Analice lo que ocurre en cada caso para pares OD de los conjuntos $S$ y $D$ que considere interesantes.

Se espera una solución creativa para cada caso, que esté bien explicada y que resuelva el problema a optimalidad considerando las restricciones involucradas. Analice y compare los resultados obtenidos con y sin aplicar las restricciones.

## Entregables

Este trabajo contempla la elaboración de un reporte en pdf (hecho en Word o LATEX) y la entrega del código en Python en dos archivos llamados XX.py y XX.ipynb, donde XX es el número del grupo. También debe incluir un README en formato md o txt en donde se expliquen las distintas librerías ocupadas en el código y se explicite su funcionamiento a nivel usuario.

## Especificaciones para el informe

La extensión máxima de cada informe es de 12 páginas. No es necesario incluir una página de portada, ni resumen, ni índice. El anexo puede tener una extensión máxima de 5 páginas y toda figura, tabla y gráfico debe tener nombre, estar rotulado y ser autoexplicativo. Todos los gráficos deben tener título, nombre de ejes y leyenda. La letra debe ser Calibri o similar tamaño 11, a interlineado simple y justificado. El informe debe abordar, como mínimo, los siguientes puntos:

- Breve introducción: Descripción general del contenido del informe y reporte de lo logrado por parte.
- Desarrollo de cada parte:
    - Descripción técnica de la solución y metodología.
    - Estructura general de código implementado. Basta con un pseudocódigo, un diagrama de flujo, u otra forma de clarificar lo programado. No necesita detallar funciones que llama desde NetworkX.
    - Análisis cualitativo de la solución y recomendaciones (si aplica).
- Conclusión: Breve descripción de los puntos abordados en la tarea junto con sus respectivas conclusiones.

## Especificaciones para el código

Toda herramienta deberá ser programada en Python. Los códigos deben estar adecuadamente comentados e indentados. Cada programa será verificado con herramientas de prevención de plagio. Se permitirá utilizar código encontrado en Internet, pero debe citarse adecuadamente la fuente en el reporte. Los comentarios del código deben ser propios y debe explicar de forma resumida lo que hace el código citado y por qué es útil en la tarea.

El archivo llamando XX.py solo debe contener las funciones sin ejecuciones de estas, mientras que el archivo llamado XX.ipynb si puede contener ejecuciones, comentarios, tablas y lo que estimen conveniente. Ambos archivos deben contener las mismas funciones.

No se permite el uso de librerías externas ni de *solvers* de optimización (Cplex, Gurobi, etc.) sin previa consulta al ayudante mediante el foro.

El uso de *Networkx* se limita a solo la primera parte. Cualquier sección del código que utilice librerías no permitidas no será considerada en la corrección.

Se debe diseñar el código de manera que sea intuitivo su uso y ejecución. Se debe programar para una base de datos genérica con el formato establecido. También tiene que contar con la opción de realizar ejecuciones parciales de las diferentes secciones, por ejemplo, solo ejecutar los algoritmos para encontrar rutas mínimas en determinados pares OD, considerando tiempo o distancia. Esto se debe definir en un bloque ***main*** en la última celda del código.

## Otros aspectos generales

El trabajo se debe realizar en grupos de máximo tres alumnos organizado por los mismos estudiantes en CANVAS.

Se contemplará realizar una evaluación de pares después de la entrega en donde podría subir o bajar la nota de cada integrante del grupo.

Las entregas se pueden enviar hasta el miércoles 1 de julio a las 23:59 vía CANVAS (los retrasos serán fuertemente penalizados). Se debe entregar un archivo `XX.rar` o `XX.zip` (donde `XX` es el número de grupo) que contenga los productos a entregar mencionados anteriormente. No envíe ejecutables.

Cualquier duda sobre el trabajo, hacerla preferentemente a través del foro del curso en CANVAS o a pcseisdedos@uc.cl .