UNIVERSIDAD TORCUATO DI TELLA

Tecnología Digital V: Diseño de algoritmos
Licenciatura en Tecnología Digital
Primer Semestre, 2023

TP1 - Servicios de movilidad on-demand en tiempo real: estrategias y algoritmos

---

El contexto
La aparición masiva de plataformas durante los últimos años han modificado la forma en la que interactuamos con distintos servicios. Algunos ejemplos incluyen la compra *on-line* y el efecto en el *retail* tradicional; la planificación de vacaciones y reserva de alojamientos; el delivery de comidas y la realización de envíos de pequeña escala; y las nuevas soluciones de movilidad tanto sobre sistemas tradicionales, como los taxis, así como también plataformas *colaborativas* y de *ride sharing*. La masividad lograda por estas aplicaciones y plataformas se justifica no solo por los mejores precios, sino también por una mejora en la *experiencia del usuario* en distintos aspectos como la presencia de reviews, la comodidad en la utilización y la rápida respuesta con las respectivas soluciones. Sin embargo, es importante notar que estas diferencias no se producen solamente con ofrecer una *app*, sino que es el resultado de un proceso más complejo de análisis de distintos aspectos del servicio propuesto.
En este trabajo nos ponemos en el rol de consultores para una empresa que provee servicios de movilidad, cumpliendo el rol de *matchmaker* entre pasajeros y vehículos (taxis o particulares). En efecto, existen en el mercado local distintas alternativas de aplicaciones para conectar pasajeros con vehículos, con diversos niveles de éxito en términos de demanda y utilización. Nuestro objetivo es analizar un aspecto particular del proceso con decisiones que deben tomarse en tiempo real, evaluando el impacto en términos de la experiencia del usuario (tanto pasajeros como conductores) y de los potenciales costos involucrados.

El problema
En un instante dado (o en un intervalo muy pequeño de tiempo, digamos 10-15 segundos) y en una determinada área geográfica la empresa posee un número de pedidos de viajes a realizar y una cierta cantidad de vehículos disponibles para realizar viajes. Para cada pasajero podemos asumir la disponibilidad de un mínimo de información como

- el instante en el que realizó el pedido;
- la localización de origen del viaje (donde debe ser buscado);
- la localización de destino del viaje (donde debe ser llevado) o la distancia estimada del recorrido;
- una estimación de la tarifa total a cobrar por el viaje (que depende en parte de la distancia del recorrido, pero puede contener gastos extra como valijas, cantidad de pasajeros, horas pico, etc.).

Análogamente, también podemos asumir mínimamente conocer en tiempo real la localización de cada uno de los vehículos de nuestra flota, en particular la de aquellos que estén disponibles para realizar viajes. Combinando la localización de pasajeros y vehículos, podemos asumir también que conocemos la distancia a recorrer que le llevaría a cada vehículo llegar a cada posible cliente a fin de poder empezar el viaje. El problema que buscamos resolver es decidir que vehículo debe buscar a cada pasajero.
A fin de formular un modelo para la decisión en cuestión, primero formalizamos el problema. Tenemos n vehiculos disponibles, $i = 1, \ldots, n$ para cubrir n viajes de distintos pasajeros, $j = 1, \ldots, n$. Por simplicidad, asumimos que el problema se encuentra balanceado en términos de

oferta y demanda de viajes.[1] En función de la información geográfica de los conductores y pasajeros, definimos d_{ij} a la distancia que debe recorrer el conductor i para empezar el viaje del pasajero j. Adicionalmente, para un pasajero $j = 1, \ldots, n$ llamamos v_j a la distancia del viaje a realizar por el pasajero j y f_j a la tarifa total a cobrarle por el viaje. Asumimos también que los pasajeros se encuentran ya ordenados de manera creciente en función del instante en el que realizaron el pedido.

Estrategias de resolución
La decisión puede ser abordada con distintos enfoques. Una primera aproximación natural al problema consiste en atender a los pasajeros uno a uno siguiendo el criterio *First Come, First Served* (FCFS), y tomar para cada pasajero una decisión *greedy* asignando el conductor disponible más cercano. En base a nuestras definciones, esta estrategia consiste en los siguientes pasos:

- considerar a los pasajeros por orden de llegada;
- a cada pasajero asignar el vehículo más cercano;
- cada vehículo debe ser usado por exactamente un pasajero.

Esta corresponde a la estrategia aplicada actualmente por la empresa.
Sin embargo, aún con un contexto de toma de decisión en tiempo real, si el problema presenta un volumen de demanda significativo es posible modificar el proceso aplicando la idea de *matching* si se esperan algunos segundos y se arma un *batch*. El concepto detrás de esta idea es simple: dentro de los parámetros permitidos, tratar de agrupar varios pedidos y, en lugar de tomar decisiones locales a cada pasajero, reformular el problema de manera global y tomar una decisión conjunta. En nuestro caso, podemos asumir que los n vehículos disponibles y los n pasajeros solicitan un viaje en un lapso de tiempo pequeño, algunos pocos segundos, en los cuales el correspondiente usuario está esperando una repuesta (ya sea pasajero, con su vehículo asignado, o un vehículo, con su pasajero). Luego, el objetivo es formular un modelo que tome una decisión global, indicando qué vehículo es asignado a cada pasajero.
Inicialmente, consideramos como métrica de éxito la minimización de la distancia recorrida por los vehículos hasta la ubicación de su pasajero asignado. Dado que asumimos la oferta y demanda balanceada, todos los pedidos deben ser cumplidos. Nuestro objetivo es proveer evidencia basándonos en datos y metodología formal respecto a la mejora en la distancia total recorrida por los vehiculos. Esta métrica se puede utilizar como un *proxy* de los costos y tiempos de espera de los pasajeros, bajo la hipótesis que la distancia tiene algún tipo de correlación con estos otros factores.

El entorno
Contamos con un desarrollo básico nuestro problema, incluyendo una clase `TaxiAssignmentInstance` que se encarga de leer la definición de una instancia del problema de un archivo de entrada. La misma contiene los siguientes atributos:

- `n`: cantidad de vehículos/pasajeros.
- `taxis_position`: `vector<pair(double,double)>` de n elementos, donde la posición i tiene la localización en términos de longitud y latitud del taxi i, $i = 1, \ldots, n$.
- `paxs_position`: `vector<pair(double,double)>` de n elementos, donde la posición j tiene la localización en términos de longitud y latitud del pasajero j, $j = 1, \ldots, n$.
- `paxs_trip_dist`: `vector<double>`, donde la posición j tiene la distancia (en kms) del viaje a realizar por el pasajero j, $j = 1, \ldots, n$.
- `paxs_tot_fare`: `vector<double>`, donde la posición j tiene la tarifa total (en USD) del viaje a realizar por el pasajero j, $j = 1, \ldots, n$.

[1] También evitamos algunos otros aspectos como las prioridades de los pasajeros/conductores, así como también lograr un distribución razonable de viajes entre los distintos conductores a lo largo del día.

- `dist`: `vector<vector<double > >` con una matriz que en la posición (i, j) posee la distancia d_{ij} (en kms) que debe recorrer el vehículo i para empezar el viaje del pasajero j, $i, j = 1, \ldots, n$. **Notar que los pasajeros están representados por las columnas (**$j = 1, \ldots, n$**).**

En caso de valores de tipo `double`, se puede asumir que los mismos pueden ser truncados al primer valor decimal en caso de ser necesario. Recordar que los pasajeros se encuentran ya ordenados según instante de llegada a la aplicación siguiendo un orden creciente en función de j. La clase puede obtener la información de un archivo de datos con un formato específico, diseñado por la cátedra, indicando simplemente en el constructor con un `string` la ruta de acceso al archivo de entrada. Por completitud, el formato de los archivos de entrada es el siguiente:

- Una primera línea indicando el valor de n.
- n líneas con la información de longitud y latitud para cada taxi, separados por `','`.
- n líneas con la información de longitud, latitud, distancia del viaje y costo total para cada pasajero, separados por `','`.
- La matriz de distancias $D = (d_{ij})$ de manera explícita, represantada por filas, una por línea. Los elementos se encuentran separados con el delimitador `','`.

Para la evaluación del problema, se cuenta con 4 sets de instancias de distintos tamaño, para modelar escenarios variados de demanda. Las mismas deben ser utilizadas para comparar la efectividad de nuestro enfoque y realizar una comparación extensiva entre el método de *FCFS* y el de *matching*. Las caraceterísticas de las instancias son:

- `small`: $n = 10$;
- `medium`: $n = 100$;
- `large`: $n = 250$;
- `xl`: $n = 500$.

Cada grupo posee 10 instancias distintas, a fin de agregar variabilidad en los escenarios considerados. Respecto a las instancias, las mismas fueron generadas de la siguiente manera:

- La información de pasajeros fue muestreada de forma aleatoria de registros de viajes de taxi de Nueva York para el mes de Diciembre de 2018.[2]
- Las instancias se restringen al área de Manhattan.
- Los taxis fueron generados aleatoriamente utilizando información geoespacial de la zonas geográficas de Mahattan.
- La posición geográfica tanto de los pasajeros como de los vehículos fue generada aleatoriamente usando la información de las zonas en cada caso.[3]
- Teniendo la ubicación geográfica es posible obtener la distancia entre dos ubicaciones usando alguna API (por ejemplo Google Maps). Sin embargo, para evitar realizar estas consultas, en esta primera etapa una solución de compromiso consiste en considerar la *distancia Manahttan* (o norma 1) entre dos puntos. La misma puede se calculada con una cuenta estándar, resulta en una buena aproximación de la distancia a recorrer y elimina el factor de conexión a un servidor para obtener información. La matriz de distancias utilizada se calcula usando esta medida.

[2] `https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page`

[3] Cabe destacar que, por cuestion de resguardo de privacidad de los datos, no se cuenta la información precisa de la geolocalización de los pasajeros.

El trabajo
La resolución del trabajo se compone de varias etapas, incluyendo modelado, programación, experimentación, así también como el reporte detallado de las metodologías aplicadas y los resultados obtenidos que justifican el análisis. Se pide:

1. **(1 punto) Estrategia FCFS.** Implementar la solución que se corresponde con la estrategia FCFS. Para ello, completar la clase `GreedySolver`, que toma una instancia y resuelve el problema con la estrategia FCFS. Esta clase debe guardar la información respecto al valor de la función objetivo, tiempo de resolución, y la solución a implementar usando la clase `TaxiAssignmentSolution`, ya implementada.

2. **(1 puntos) Modelo para estrategia de *batching*.** Modelar el problema usando grafos, incluyendo todos los parámetros y definiciones necesarias. Justificar la elección del mismo y por qué lo consideran apropiado para el problema.

3. **(2 puntos) Implementación estrategia de *batching*.** Realizar una implementación del modelo propuesto en el ítem anterior. Para ello, completar la clase `MinCostFlowSolver`, con un comportamiento análogo a `GreedySolver`. **Detalles sobre cómo realizar esta implemtanción serán dados más adelante.**

4. **(1 punto) Experimentación.** Realizar experimentos sobre todas las instancias comparando el modelo propuesto respecto al de la solución actual de la compañía. En cada caso, medir la mejora porcentual obtenida. Sean z_b y z_g el valor de la función objetivo de una solución del modelo para el *batching* y el de FCFS, respectivamente. definimos la mejora relativa como

$$\%\text{gap} = \frac{z_g - z_b}{z_b}.$$

   En caso de considerarlo conveniente, pueden agregar otras métricas complementarias (respecto al método o a las soluciones) para el análisis de los resultados.
   Sugerencia: se recomienda sistematizar la realización de experimentos, idealmente definiendo la lista de instancias a considerar y reportando en algún formato conveniente (por ejemplo, csv) el resumen de los resultados obtenidos, para ser analizados posteriormente.

5. **(2 puntos) Discusión y análisis de resultados.** Analizar los resultados obtenidos en función de la comparación entre ambos métodos. Incluir como parte de la discusión aquello que considere relevante en relación a las características del modelo propuesto para resolver el problema (por ejemplo, tiempo de ejecución vs. contexto de aplicación, posibles adaptaciones ante limitantes, etc.).

6. **(1 punto) Limitaciones y posibles extensiones.** El nuevo modelo propuesto puede no considerar algunos aspectos importantes a la hora de implementarlo. En base a encuestas realizadas a los conductores en general, las aplicaciones sugieren muchas veces viajes que demandan una distancia considerable para llegar a la ubicación del pasajero, para luego realizar un muy viaje corto en comparación. En este sentido, la sensación de los conductores es que es mucho el costo de buscar el pasajero, ya sea en costo específico o en el tiempo utilizado y que podrían destinar a un viaje más rentable, en relación al beneficio obtenido por el viaje en sí mismo. Se pide analizar las soluciones de los métodos propuestos utilizando las distancias paticulares codificadas en la solución con la información adicional disponible de cada pasajero. Proponer alguna modificación a los modelos para abordar la problemática identificada.

7. **(2 puntos) Modelo alternativo.** Implementar la mejora propuesta y comparar los nuevos resultados con los obtenidos originalmente en términos de la distancia total recorrida. De ser posible, medir el impacto.

Modalidad de entrega

Se pide presentar el modelo y la experimentación en un informe de máximo 15 páginas que contenga:

- introducción al problema y la decisión,
- descripción del modelo propuesto, según corresponda,
- consideraciones generales respecto a la implementación del modelo, incluyendo dificultades que hayan encontrado,
- resumen de resultados obtenidos en la experimentación,
- conclusiones, posibles mejoras y observaciones adicionales que consideren pertinentes.

Junto con el informe debe entregarse el código con la implmenetación del modelo. El mismo debe ser entendible, incluyendo comentarios que faciliten su corrección y ejecución.

Fechas de entrega

Formato Electrónico: **2 de Junio de 2023**, enviando el trabajo (informe + código) vía el campus virtual.
Importante: El horario es estricto. Las entregas recibidas después de la hora indicada serán considerados re-entrega.