UNIVERSIDAD TORCUATO DI TELLA

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

TP2 - Logística centralizada de primera milla

El contexto
Con el desarrollo del e-commerce en los últimos años, marketplaces y empresas logísticas en Latinoamérica y otras regiones (por ejemplo, Asia) recurren a diferentes estrategias para reducir los costos operativos. Para vendedores grandes, con gran volumen de ventas diario, se suele planificar una visita para recolectar las mismas por los negocios. Sin embargo, esta operación es costosa en términos logísticos para vendedores pequeños (*vendedores* de acá en más), que venden sólo unos pocos ítems por día (digamos, menos de 10 o 15). En estos casos, ellos son responsables por la logística necesaria para entregar los paquetes hasta el punto de recepción de la red (usualmente llamada *primera milla*).
Hemos conseguido recientemente un contrato para asistir a la *start-up* **ThunderPack**. La empresa, además de proveer servicios logísticos, gestiona una red de comercios de distinto tipo que, como actividad secundaria, ofrecen de forma limitada la posibilidad de almacenar temporariamente paquetes, para que luego las empresas logísticas realicen la colecta directamente por estos puntos. Así, estos negocios actúan como puntos de consolidación, simplificando la operatoria. Por simplicidad, nos referiremos como *depósitos* a estos puntos de recepción. La Figura 1 muestra un ejemplo, donde se denota con el color azul a los 1100 *vendedores* y en rojo a los 310 *depósitos*.

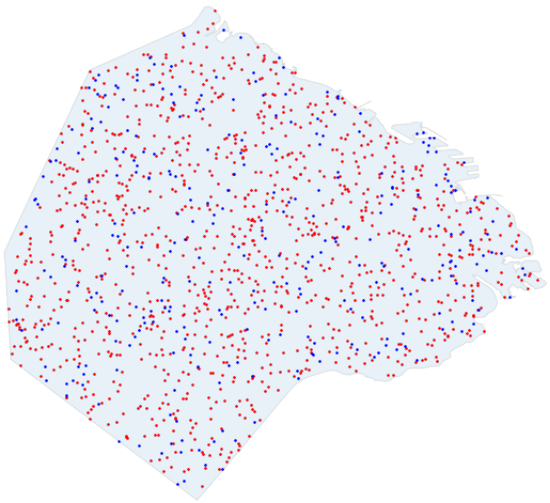

Figure 1: Disposición geográfica de los vendedores (rojo) y depósitos (azul)

Algunos eventos particulares, como el *Black Friday*, *Hot Sale* y campañas similares generan un nivel de actividad por fuera de lo estándar, estresando toda la red logística. En este caso, a fin de garantizar una buena experiencia a los vendedores que utilizan el servicio, es necesario tomar medidas preventivas para poder responder de la manera más efectiva posible.
Actualmente, la modalidad de operación es que los vendedores deciden a qué depósito llevar sus paquetes, pudiendo cambiar sin previo aviso. En base a estudios previos, se nota que esta modalidad genera una utilización ineficiente de la capacidad de almacenamiento de la red, que se traduce en reclamos, costos adicionales y una mala experiencia para los usuarios (i.e., vendedores) ya que en caso de elegir un depósito sin capacidad de almacenamiento, entonces debe recorrer

otros posibles destinos hasta encontrar un con capacidad disponible. Por esta razón, **ThunderPack** está considerando migrar a una modalidad centralizada, donde a cada vendedor se le asignará un depósito para que utilice regularmente buscando cumplir con las capacidades máximas de cada depósito y buscando minimizar la distancia total recorrida por los vendedores. Hoy en día, **ThunderPack** tiene algunas preguntas que busca idealmente responder:

- *¿Es suficiente la capacidad de la red de depósitos, o es necesario expandir la misma para poder dar respuesta a todos los vendedores?*
- *¿Es posible encontrar una asignación donde los vendedores recorran una distancia razonable para entregar sus paquetes?*
- *Es factible desarrollar una herramienta que nos permita experimentar con distintos escenarios y obtener soluciones de buena calidad en unos pocos minutos?*. Esta herramienta podría utilizarse en estos contextos de alta demanda, posiblemente en tiempo real, a medida que se va observando el volumen de ventas real.

El modelo

El problema descripto anteriormente puede ser modelado mediante el *Problema de Asignación Generlizada* (GAP, *Generalized Assignment Problem*) que, en su versión más general, puede ser formulado de la siguiente forma. Sea $N = \{1, \dots, n\}$ el conjunto de *vendedores* y $M = \{1, \dots, m\}$ el conjunto de *depósitos*. Cada depósito $i \in M$ tiene una capacidad máxima de recepción c_i, medida en cantidad de unidades. Dado un vendedor $j \in N$ y un depósito $i \in M$, d_{ij} denota la demanda (también en cantidad de unidades) a utilizar y c_{ij} el costo incurrido si j es asignado a i. Con estas definiciones, podemos representar una solución como una colección de conjuntos $\Gamma_1, \dots, \Gamma_m \subseteq N$, con Γ_i el subconjunto de venededores asignados al depósito $i \in M$. Luego, el GAP consiste en:

1. asignar cada $j \in N$ a exactamente un depósito $i \in M$, es decir, $\Gamma_i \cap \Gamma_k = \emptyset$ si $i \neq k$, $i, k \in M$;
2. la capacidad de cada depósito no debe ser excedida, es decir, para $i \in M$
$$\sum_{j \in \Gamma_i} d_{ij} \leq c_i$$
3. minimizar el costo total de la asignación dada por
$$\sum_{i=1}^{n} \sum_{j \in \Gamma_i} c_{ij}.$$

En el contexto del problema de **ThunderPack**, tomaremos c_{ij} como la distancia que debe recorrer el vendedor j en caso de ser asignado al depósito i. Notar que, dependiendo de las capacidades y las demandas, podría dificultarse encontrar una solución factible que asigne a todos los vendedores y respete las capacidades. Definimos

$$c_{\max} = \max_{i \in M, j \in N} c_{ij}$$

como la máxima distancia posible a recorrer por un vendedor. En caso de tener una solución parcial, donde haya vendedores que no pueden ser asignados a algún depósito, se asumirá una penalización de $3 \times d_{\max}$ por cada uno de ellos.
Para cada vendedor, nuestra estimación de la demanda (en cantidad de unidades) es la misma independientemente del depósito al que sea asignado, y por lo tanto $d_{ij} = d_j$. De todas formas, trabajaremos con la versión general del GAP y, al momento de abordar las instancias con datos reales, serán un caso particular para el algoritmo.

Los datos

Para evaluar los métodos y algoritmos, se considerarán dos fuentes posibles de instancias:

- Instancias de *benchmark* del GAP de la literatura científica, usualmente utilizadas para comparar y dar evidencia de la efectividad de los algoritmos. Junto con el enunciado se adjuntan un conjunto de diversas instancias tomadas del dataset propuesto en el siguiente link. El dataset considera instancias de distinto tamaño, todas ellas definidas en un archivo de input de texto plano siguiendo el formato allí especificado.
- Una instancia de tipo real de gran escala, construída a partir de datos simulados. La instancia posee $n = 1100$ vendedores, $m = 310$ depósitos. El formato del archivo con su definición es el mismo que en el dataset anterior. A su vez, en caso de querer visualizar resultados, también se provee un archivo JSON que, además, incluye información de geolocalización de los vendedores y depósitos.

Estas instancias deben ser utilizadas para evaluar el desempeño de los métodos y algoritmos propuestos. A su vez, la instancia real debe ser utilizada para analizar el problema desde la perspectiva del problema.
A fin de poder evaluar los resultados obtenidos, el programa que ejecute los algoritmos implementados deberán recibir (al menos) como parámetro:

- el nombre del archivo de input con la instancia;
- el nombre del archivo de output donde escribir la solución;
- cualquier otro parámetro que el grupo considere necesario, deberá ser explicado en el informe.

El formato del archivo de salida debe ser el siguiente:

- una línea por depósito, donde la línea i-ésima corresponde al depósito i;
- una lista de números, separados por un espacio, con los índices de los vendedores asignados a cada depósito.

El trabajo

La resolución del trabajo se compone de varias etapas, incluyendo modelado, programación, experimentación, así también como el reporte detallado de la evaluación de los métodos implementados, los resultados obtenidos y la discusión sobre el caso real. El trabajo práctico debe ser implementado en C++. Se pide:

1. **(2 puntos) Heurísticas constructivas.** Proponer e implementar (al menos) dos heurísticas constructivas distintas para el GAP.
2. **(2 puntos) Operadores de búsqueda local.** Proponer e implementar (al menos) dos operadores de búsqueda local para el GAP.
3. **(1.5 puntos) Metaheurística.** Proponer e implementar una metaheurística para el GAP.
4. **(1.5 puntos) Experimentación y discusión.** Realizar una experimentación exhaustiva con las instancias de benchmark provistas, comparando los métodos propuestos y realizando un tunning de parámetros.
5. **(1.5 punto) Análisis del caso real.** Evaluar los métodos propuestos sobre la instancia real, incluyendo una discusión comparando calidad de resultados, tiempo de ejecución e implicancias prácticas en relación al problema que enfrenta **ThunderPack**.
6. **(1.5 puntos) Modelo, informe, presentación de resultados y delivery del código.** El modelo, la descripción de los algoritmos, los operadores, las decisiones de diseño, la implementación, el testing realizado, la presentación de resultados, instrucciones de compilación y ejecución.

Competencia

Para motivar el desarrollo de algoritmos super eficientes, pero sin implicancias en cuanto a la nota, se realizará (al menos) una competencia buscando la mejor solución posible para la instancia real. Para esta competencia, las reglas son las siguientes:

- La métrica de evaluación será la función objetivo antes mencionada.
- No hay limitación de tiempo de ejecución.
- Para ser considerado ganador, se deberá mostrar evidencia respecto a la utilización del algoritmo implementado para el cálculo de la solución propuesta, idealmente indicando los parámetros de compilación y ejecución. La evaluación y decisión del jurado (compuesto por Javier y Juanjo) es final y no apelable.
- No es posible utilizar herramientas y/o liberías por fuera de lo requerido por el TP.
- Los docentes de la materia pueden participar de la competencia.
- Está permitido usar bibliografía científica como insipiración para el desarrollo de los métodos.

Más información respecto a premios y otros posibles formatos de competencia serán detallados en las próximas semanas.

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: **7 de Julio de 2023, 23:59 hs**, 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.