**Punto 2.** Para este punto, primero se planteó el grafo como un problema de flujo en redes de ruta más corta para encontrar el menor costo posible entre la bodega y los clientes para que solo se pase una sola vez por cada cliente. Ahora bien, como era necesario tener en cuenta que cada cliente tenía una demanda, también era necesario formular un problema de flujo de costo mínimo donde la se tuvieran en cuenta las ofertas como cada vehículo y las demandas en cada cliente. Como estos dos problemas son diferentes, pero ambos de flujo en redes, se utilizó una formulación de flujo en redes con “comodities” utilizando la descomposición de Dantzig-Wolfe para resolverlo utilizando el algoritmo de Simplex en vez de un problema entero MIP, gracias a que Simplex es mucho más eficiente computacionalmente.

En primer lugar, se planteó el grafo del problema. Para esto, se creó un nodo por cada vehículo, con el motivo de representar de una mejor manera que se pueden generar varias rutas diferentes. Adicionalmente, cada uno de estos nodos oferta una cantidad de helados igual a la capacidad máxima del vehículo. Adicionalmente, se creó un nodo por cada cliente, donde cada uno tiene una demanda (valor negativo). Por último, y dado que la suma de la capacidad de todos los vehículos disponibles es mayor a la demanda total, se crea un nodo final a donde se dirigen todas las unidades que sobran. Cabe resaltar que ningún arco sale de este nodo y que para todo nodo existe un arco hacia este último.

Ahora bien, en este grafo es necesario considerar dos problemas diferentes. Uno, en el que un solo vehículo puede pasar por cada nodo, y que con la menor cantidad de vehículos posible se deben recorrer todos los clientes con el mínimo costo posible. Por lo tanto, se definieron los costos entre cada arco del grafo como: . Por otro lado, para todo grafo entre un vehículo y un cliente se adiciona un costo fijo: . Puesto que esto simula que se incurre en un costo fijo para el vehículo i, en caso de que este sea utilizado en el problema. Por otro lado, esto implica que para los arcos entre los vehículos y el nodo de sobra el costo es 0, debido a que si hay flujo por este arco significa que el vehículo no fue utilizado:   
 . Ahora bien, este costo solo está asociado al transporte de vehículos, puesto que no existe un costo asociado a transportar un número específico de helados. Por último, el costo desde cualquier cliente hasta el nodo de unidades de sobra es igual al costo de ir desde cualquier cliente hacia la bodega, puesto que este arco representa que se acaba la ruta para el vehículo y retorna a la bodega. Este grafo, se muestra en el Anexo 4 (Diagrama 2). Adicionalmente, los parámetros de este problema brindados por MOPTA, fueron procesados en el proyecto de Eclipse llamado “proyecto1FlujoEnRedes”. Al ejecutar el archivo Solucion2.java, se generan los parámetros que recibe Xpress para resolver el problema.

Ahora bien, se deben formular dos problemas sobre este grafo. Uno, el problema de suplir la demanda para todos los nodos, puesto que es un requisito para el problema, aunque esto no cambia los costos de la función objetivo. Además, se debe cumplir que para todo cliente solo debe haber un vehículo que lo visite, excepto el nodo final a donde llegan todos los vehículos. Por otro lado, se debe enviar una cantidad determinada de helados hacia cada cliente, tomando en cuenta la capacidad de cada vehículo. Se debe cumplir que lo que llega a cada cliente, menos lo que sale debe ser igual a lo que este demande de helados. Por lo tanto, tenemos una formulación de flujo en redes con dos “comodities” una, los vehículos y otra, los helados que se mueven por cada arco. Por lo tanto, tenemos la siguiente formulación:

**Conjuntos:**

**Parámetros:**

**Variable de decisión**

**Restricciones:**

1. Restricción de balance para todos los nodos:
2. Restricción para vehículos. Solo puede entrar y salir un arco de cada nodo. Es decir, solo debe pasar un vehículo por cada cliente
3. Restricción para que solo haya flujo de helados por un arco sí y sólo sí hay un vehículo que pasa por el mismo arco
4. Naturaleza de las variables. Para un problema de flujo en redes, la unimodularidad y triangularidad de la matriz A, no es necesario definir como enteras las variables. Sin embargo, solo las restricciones 1 y 2 conforman matrices de este tipo, mientras que la restricción 3 arruina la estructura de la matriz. Por lo tanto, con esta formulación general MIP, en vez de tener:

Tenemos que:

**Función objetivo**

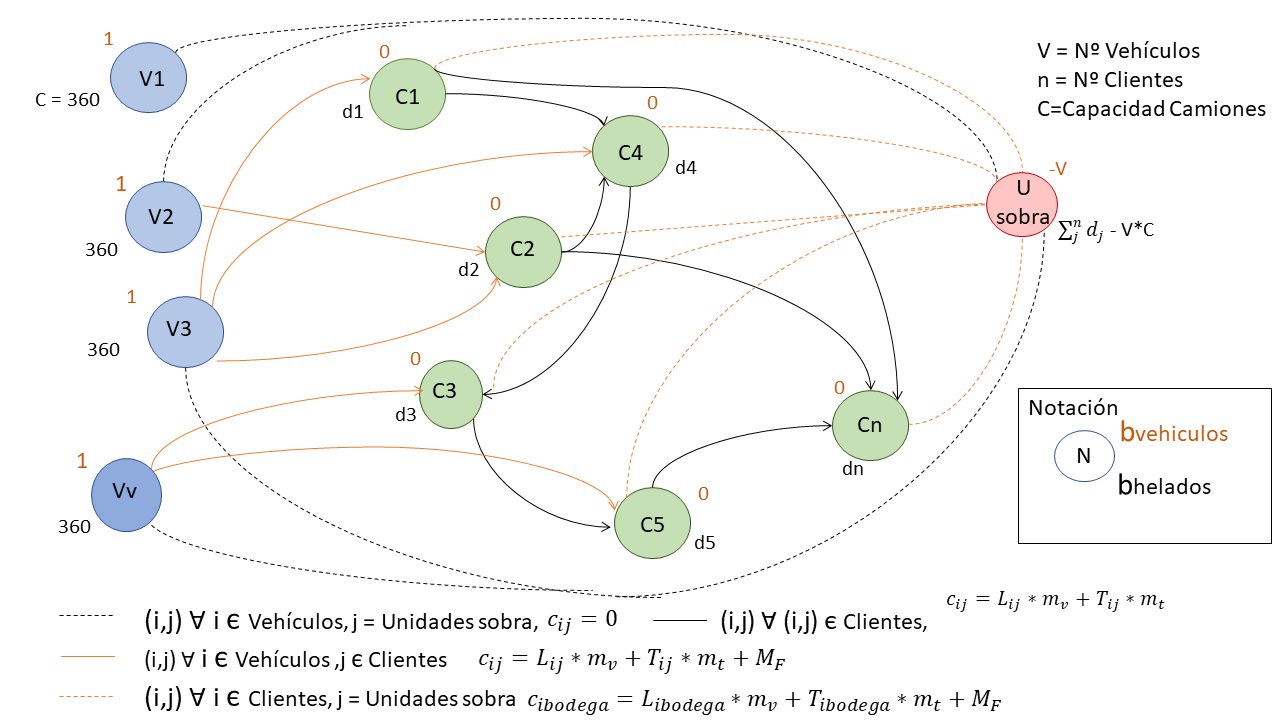
Para conservar la estructura de flujo en redes, es posible realizar la descomposición Dantzig-Wolfe. Donde se conserva la estructura de flujo en redes para problemas auxiliares más pequeños que se resuelven de manera iterativa, y generan columnas de la matriz A del problema general para encontrar una solución óptima para problemas escalables. Este problema, se soluciona utilizando los puntos extremos de la región convexa que forma el espacio de soluciones factibles de cada problema.

Por último, se encontró la solución óptima para el problema con los parámetros dados:

* Sale camión 1 hasta cliente 3, Se devuelve hacia la bodega desde cliente 3
* No se utiliza el camión 2
* Sale camión 3 hasta cliente 6, Va de cliente 6 hasta cliente 5, Se devuelve hacia la bodega desde cliente 5
* Sale camión 4 hasta cliente 9, Va de cliente 9 hasta cliente 1, Se devuelve hacia la bodega desde cliente 1
* No se utiliza el camión 5
* No se utiliza el camión 6
* Sale camión 7 hasta cliente 4, Va de cliente 4 hasta cliente 7, Va de cliente 7 hasta cliente 2, Se devuelve hacia la bodega desde cliente 2
* Sale camión 8 hasta cliente 8, Va de cliente 8 hasta cliente 10, Se devuelve hacia la bodega desde cliente 10  
  **El costo total de transporte sin incluir demoras es de $5931.33**

Finalmente, la formulación de flujo en redes tiene muchas ventajas en el momento de resolver problemas de este tipo, gracias a que, por medio de una formulación simple, es posible obtener soluciones de manera eficiente, con un número de iteraciones bajo y que no crece tanto a medida que crece el problema (es decir, es escalable). Las soluciones con esta formulación suelen ser mucho más rápidas que problemas de optimización lineal utilizando variables enteras que resuelven el problema utilizando Branch and bound, y todo gracias a una propiedad en la matriz de restricciones que permite relajar la naturaleza de las variables. Ahora bien, para un problema de un grafo, el número de nodos y arcos siempre va a crecer de manera exponencial, a medida que crezca el problema, sin embargo, el tiempo computacional no se verá tan afectado como en un problema de optimización lineal utilizando variables enteras. Adicionalmente, en caso de que se tengan redes con flujos de diferentes tipos, se pueden descomponer en diferentes problemas que conserven la formulación en Simplex, y de esta manera conservar la solución de manera eficiente. Sin embargo, una de las desventajas de la formulación de este tipo, es que no en todos los casos se puede formular, por lo que puede ser una solución muy útil, pero no se garantiza que resuelva todos los problemas.

**Anexos**

****