CAPÍTULO 2

OBJETIVO, DISEÑO E IMPLEMENTACIÓN DEL ALGORITMO.

2.1. Objetivo del Algoritmo.

2.2. Datos de partida.

2.3. Restricciones del Algoritmo.

2.4. ¿Cómo comenzar el algoritmo? Introducción al algoritmo.

2.5. Resolución del Algoritmo. Funciones y Programación.

2.6. Test del algoritmo sobre diferentes configuraciones.

2.1 OBJETIVO DEL ALGORITMO:

En primer lugar se va a precisar el objetivo del algoritmo a implementar:

Se pretende diseñar un algoritmo que sea capaz de ofrecer una "preplanificación" sobre la trayectoria aproximada que deben de seguir un conjunto de rutas, tratando siempre de maximizar el flujo de viajeros o mercancías, etc. que es capaz de "cargar" el conjunto de rutas; y de minimizar la longitud de las mismas, que es un objetivo clásico del diseño de rutas (*shortest path*), puesto que cuanto menor sea la longitud de las mismas, menor será el coste del operador, y más atractivo será para el cliente o usuario de esa ruta, puesto que implica menores tiempos de viaje.

Será, además, introducida la posibilidad de establecer de antemano una línea o ruta que puede ser llamada "principal" y que como tal, producirá una atracción sobre el resto de rutas. Pensemos en el caso de una autopista o vía rápida que una varias ciudades, puede provocar que carreteras secundarias converjan hacia esta vía. O el caso de una línea de metro en una ciudad, que como es normal atraerá a un número elevado de pasajeros, y por lo tanto, inducirá una atracción de las líneas de autobús hacia la de metro.

2.2 DATOS DE PARTIDA:

Conocido el objetivo, otro aspecto fundamental es conocer cuáles son los datos de partida:

- Se tendrán una seria de nodos distribuidos geográficamente, y que pueden representar pueblos y ciudades, o zonas de transporte dentro de un área urbana (como será el caso en la aplicación práctica que se verá como conclusión de este proyecto).
- Se conocerá, además la matriz de conectividad, es decir, el modo en que estos nodos están unidos; y la distancia entre ellos.

- Otro dato será la matriz de demanda Origen-Destino, que representa el número de mercancías o viajeros que desean ser transportados entre cada uno de los nodos antes señalados.

2.3 RESTRICCIONES DEL ALGORITMO:

Las rutas que se vayan creando deberán cumplir una serie de restricciones, éstas son:

- Ninguna ruta deberá sobrepasar una *Longitud máxima de ruta*.

  ¿Por qué esta restricción?

  En los estudios realizados con anterioridad, la limitación de la longitud de ruta es una restricción clásica, puesto que una de las preocupaciones principales es la de intentar minimizar el coste del operador, y no cabe duda que cuanto mayor sea la longitud a cubrir, mayor será el gasto del operador. Se podría entender de manera más matemática como el hecho de asignar un coste a cada una de las "ramas" que unen los diferentes nodos (en nuestro caso identificamos *coste* con *Longitud de ruta)*, y la restricción sería que la suma de los costes de las ramas por donde pasan las rutas no superara un coste máximo.

- Otro condicionante importante del algoritmo que se pretende programar es la de la posibilidad de transbordos. A la hora de computar el número de viajeros (si asimilamos nuestro modelo al caso de transporte público) que utilizan en un determinado momento una ruta concreta, se computarán también aquellos que realicen un transbordo en cualquiera de las paradas de esa ruta hacia cualquier parada de otras rutas que intersecciones con la ruta en estudio. Si bien, el número total de pasajeros que viajan de un punto a otro mediante un transbordo intermedio se verá reducido. En este algoritmo esta disminución de "carga" vendrá representada por un parámetro *alfa*, que tendrá valores diferentes si el transbordo se produce entre rutas secundarias (por ejemplo, dos líneas de autobús) *alfa_bus* , o entre una ruta secundaria y

otra principal (por ejemplo, entre una línea de autobús y otra de metro) *alfa_metro*.

No se contabilizará el flujo de "carga" entre dos nodos que puedan estar unidos mediante la realización de más de un transbordo. Es decir, esto podría ser expresado como una nueva restricción que diga que la demanda de viajes entre dos puntos solo se considerará "satisfecha" si esos dos nodos no se encuentran unidos por más de un transbordo ("satisfecho": porque el flujo total se verá afectado por el factor *alfa* si se realizan transbordos).

- Se limitará el número de nodos que pueden ser "insertados" entre los dos nodos que producen un mayor flujo de "carga", y que como se verá más tarde son los que provocan que se inicie la ruta.

¿Cuál es el motivo de esta restricción?

A la hora de diseñar el algoritmo, había conciencia de que de algún modo era necesario introducir una "utilidad temporal", es decir, el hecho de que la satisfacción o no de la demanda dependiera de la calidad del servicio. Parecía lógico pensar que cuanto menor fuera el tiempo de viaje (o la longitud de ruta, si suponemos una velocidad media constante) entre dos nodos, la demanda de viajes entre esos dos puntos se vería más satisfecha, y por lo tanto sería mayor la "carga" transportada entre ese par de nodos. Es decir, la estructura o topología de la red de rutas debería ser importante a la hora de satisfacer la demanda, que además es uno de los objetivos principales:"maximizar el flujo de viajeros o mercancías que soporten las rutas".

Sin embargo, la variación de la demanda satisfecha en función de la estructura de las rutas, suponía una complicación enorme en la implementación del algoritmo, puesto que hay que recordar que éste va trazando varias rutas a la vez. Rutas que pueden interseccionar en uno o varios puntos; varias rutas que pueden unir un mismo par de nodos, etc. En definitiva, el mallado que podía formarse era demasiado complejo como para introducir la "utilidad temporal" en la programación del algoritmo.

Sin embargo, había que introducir de algún modo algún elemento que tuviera de alguna manera presente la utilidad temporal. Así pues, se recurrió a una solución sencilla, pero que podía en cierta manera ayudar en nuestra intención. Ésta fue la de introducir una limitación en el número de nodos máximos permitidos entre el par de nodos de mayor flujo (que son los que "inician" la creación de una ruta). De este modo, se podría prácticamente asegurar que la demanda entre los dos nodos de mayor flujo queda completamente satisfecha. Lógicamente, el número de "nodos intermedios" permitidos, dependerá de las características de la red o zona geográfica a estudiar. Por ejemplo, si se pretende hacer un estudio sobre las rutas que deben ser trazadas por las líneas de autobuses en una red de carreteras del ámbito autonómico andaluz, podría obtenerse como un par de ciudades donde la demanda de transporte de viajeros es elevada, las localidades de Sevilla y Cádiz. Por lo tanto, debería de trazarse una ruta de autobús que uniera estas dos ciudades, pero seguro que conseguiríamos aumentar la carga de viajeros en esa ruta si ésta tuviera una serie de paradas intermedias que pudieran ser: Las Cabezas de San Juan, Jerez de la Frontera, Puerto de Santa María, Puerto Real y San Fernando. Pero la realidad nos dice que muy probablemente el número de personas que demandan este servicio entre Sevilla y Cádiz, y que como recordamos fueron las que provocaron que se creara esta ruta, se vería notablemente disminuido por el elevado número de paradas intermedias (y por lo tanto, tiempo elevado de viaje). Así pues, se trata de llegar a una solución de compromiso, tal que se introduzcan una serie de paradas intermedias (aumentando así la captación de viajeros) pero sin que afecte a la demanda de viajeros entre Sevilla y Cádiz. Una solución podría ser una ruta Sevilla-Jerez-Puerto Real-Cádiz.

Si en lugar de situar el ejemplo en una red de carreteras, lo hacemos sobre el transporte público de una ciudad, el número de paradas intermedias será diferente, en función de las peculiaridades de la misma. Adelantamos en este punto, que la aplicación práctica se realizará sobre la ciudad de Sevilla, en nuestro intento de preplanificar

un conjunto de rutas de autobús, y esto nos lleva a la convicción de que el hecho de no introducir una "utilidad temporal" más realista no va a suponer grandes variaciones en el uso del transporte público, por tratarse de una urbe de dimensiones no muy elevadas, y en el que tampoco existe una gran variedad de sistemas de transporte público, prácticamente solo el autobús (si exceptuamos el taxi, que no es un transporte de uso diario, sino más bien de uso puntual en unas condiciones especiales). Es por esto, que nos inclinamos a pensar que la demanda del transporte público no debería variar demasiado si se crea una red de autobuses con una cierta lógica; y que por lo tanto, la restricción mencionada de limitar el número de nodos intermedios nos parece una decisión válida para el nivel de exactitud y complejidad que deseamos.

2.4 ¿CÓMO COMENZAR EL ALGORITMO? INTRODUCCIÓN AL ALGORITMO.

La primera dificultad que se presenta es la de cómo iniciar la creación de rutas, ¿qué criterio se debe tomar para elegir los pares de nodos que inicien las rutas?

Acertar en este primer paso es importante pues de ello va a depender que las rutas que se empiecen a crear tengan desde el inicio una cierta lógica, pues la estructura de la ruta va a depender en gran medida de este primer par de nodos. Sin embargo, no hay ninguna fórmula exacta que nos dé de forma eficiente la solución a la pregunta anteriormente formulada, y por lo tanto debemos recurrir a un heurístico sencillo. Éste es el de analizar la demanda de "carga" entre todos los pares de nodos y seleccionar aquel par de valor mayor. Serán pues estos dos nodos los que comiencen a crear la ruta, si bien, es importante tener en cuenta que aunque en el primer paso se pueden considerar como nodos origen y destino, y de hecho ese nombre van a seguir manteniendo durante todo el algoritmo, puede ser que en algún momento del mismo pierdan su condición de nodos inicial y final, para pasar a estar en un punto intermedio de la ruta. Lo que no perderán nunca será la importancia que

tienen como “nodos de flujo máximo” y por lo tanto como originadores de la ruta.

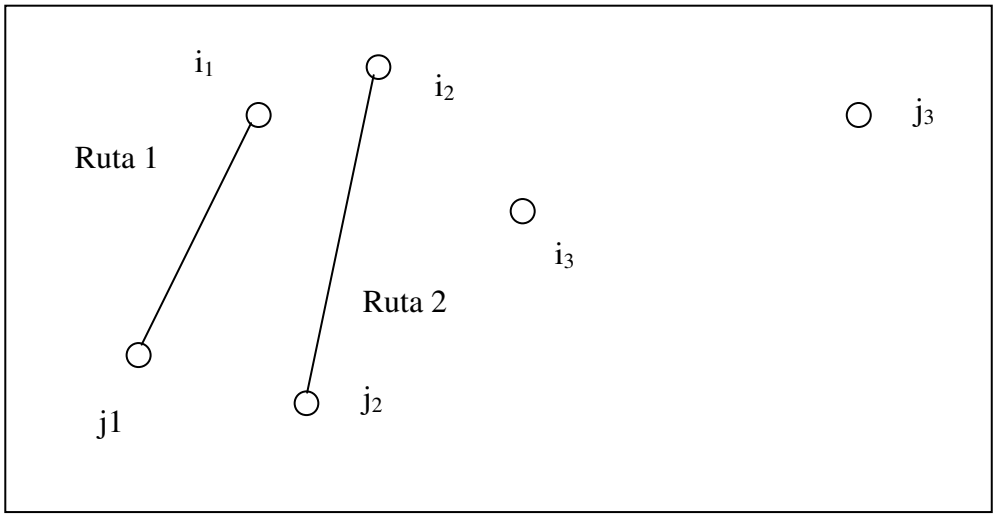
Lo que es evidente, es que no todos los pares de nodos podrán ser “nodos originadores” de rutas. En primer lugar, deberá cumplirse la primera de las restricciones antes enunciadas, es decir, que este par de nodos se encuentren a una distancia tal, que la ruta que los una no supere la longitud máxima de ruta permitida. En segundo lugar, y como consecuencia de que el programa permite la introducción de una ruta principal que no puede ser alterada y que se supone de características superiores al resto de alternativas, se considerará que un par de nodos que ha sido obtenido como “nodos de flujo máximo” y que se encuentran situados sobre la ruta principal, no serán iniciadores de ninguna ruta. Supongamos el caso de dos paradas situadas en una ciudad, entre las que se produce una demanda de viajes muy elevadas, según nuestro algoritmo, esta situación provocará que entre ambas se empiece a crear una ruta de autobús. Esto sería así, si ambas paradas no estuviesen unidas de antemano por una línea de metro, que es un medio de transporte más rápido y fiable que el autobús, por lo que se pierde la necesidad de crear una ruta de autobús entre esos dos nodos, pues la demanda ya está plenamente satisfecha con el metro.

Por supuesto, no es necesario hablar de la relatividad que engloba este método para iniciar rutas; pudiera ocurrir que la primera ruta que comienza a crearse entre los dos nodos con mayor demanda de viajes, finalmente sea una ruta sin mucha “carga” porque los demás nodos que fueron unidos a ella apenas aportaron “carga” a la misma.

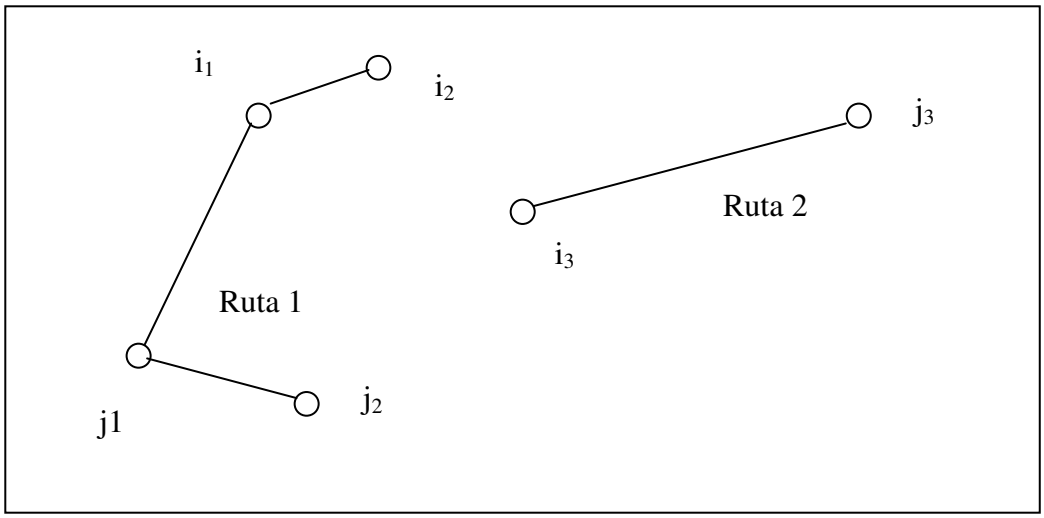
Sin embargo, este método debía ser mejorado pues era posible que al tomar los N pares de nodos con demanda de viajes máxima (tantos, como rutas deseen ser creadas), algunos de ellos se encontrasen muy próximos entre si, y por lo tanto se iniciasen dos rutas que es muy probable que discurrieran casi en paralelo, cuando, a lo mejor, una misma ruta pudiera haber englobado a los cuatro nodos, y haber continuado la creación de la misma con estos dos pares de nodos pertenecientes a la misma ruta.

Esto se entenderá muy fácilmente con el ejemplo visual siguiente:

Supongamos que deseamos crear dos rutas (N=2), y que los dos primeros pares de nodos con una demanda de viajes mayor entre ellos son (i_1,j_1), (i_2,j_2). Según el algoritmo, las dos rutas se empezarán a crear: una entre (i_1,j_1), y la otra entre (i_2,j_2):



A partir de aquí, se empezarán a introducir nodos intermedios, pero es muy probable que estas dos rutas salgan bastante similares en estructura. Es esto lo que nos lleva a pensar en la posibilidad de incluir los cuatro nodos en una misma ruta, e iniciar la segunda ruta con el tercer par de nodos en la lista de aquellos que tienen una mayor demanda de viajes entre ellos: (i_3,j_3). Esto se podrá hacer siempre que el conjunto (i_2, i_1, j_1, j_2) no supere la longitud máxima de ruta especificada.



Una vez obtenidos los pares de nodos entre los que se produce un flujo máximo de viajeros (si estamos hablando de un transporte público de personas), se comienzan a crear la rutas (a partir de ahora hablaremos de rutas de autobuses, y por lo tanto de transporte o flujo de viajeros).

El modo de proceder es sencillo. Hay que recordar que ésta será una forma de resolver este problema, que no tiene por qué dar las mejores soluciones, pero que sí nos puede dar una aproximación a la solución óptima.

Si la intención es la de crear N rutas (tras haber obtenido los N pares de nodos generadores de cada una de las rutas), en cada iteración se analizará qué nodo integrado en qué ruta produce un valor mayor de la función objetivo. Como la intención será siempre la de maximizar el número de viajeros captados, intentado reducir al máximo la longitud de la ruta, se recurrió a una función típicamente matemática y muy clara conceptualmente, ésta es:

$$F.O = \frac{\Delta F_{nodo',i}}{\Delta L_{nodo',i}}$$

siendo:

$\Delta F_{nodo',i}$ el incremento de viajeros que se produce al introducir 'nodo' en la ruta i.

$\Delta L_{nodo',i}$ el incremento en la longitud de la ruta i, al integrar 'nodo' en la misma.

De este modo, las rutas se irán creando simultáneamente, añadiéndose los nodos indistintamente a una u otra ruta, en función del valor que tome la función objetivo.

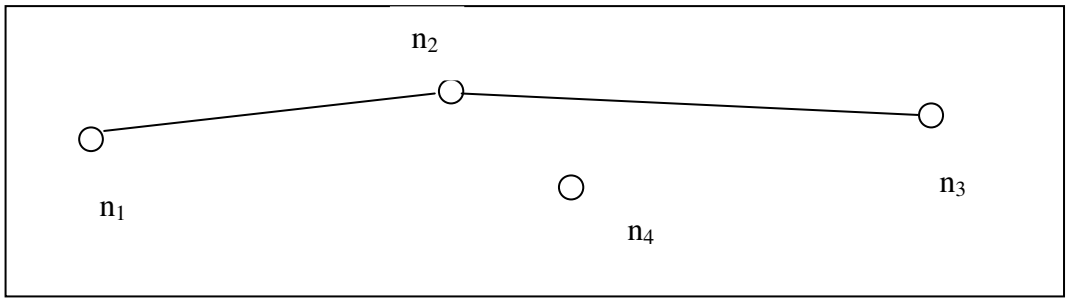
Posteriormente explicaremos la función con más detalle, pues aquí entrarán en juego también el cómputo de los viajeros que transbordan hacia

otras rutas, que merece ser explicado con más detenimiento. Pero eso será en el próximo apartado.

Puesto que, como ya se comentó, la carga de las rutas no va a ser una limitación en la creación de las mismas, puesto que hemos supuesto que no tenemos límite en la cantidad de autobuses de los que podemos disponer, y por lo tanto la frecuencia de paso de los mismos no nos plantea ningún problema; el factor que va a provocar que se "cierren" las rutas, va a ser la longitud de ruta máxima permitida. Por lo tanto conocer con la mayor exactitud posible la longitud de las líneas es importante.

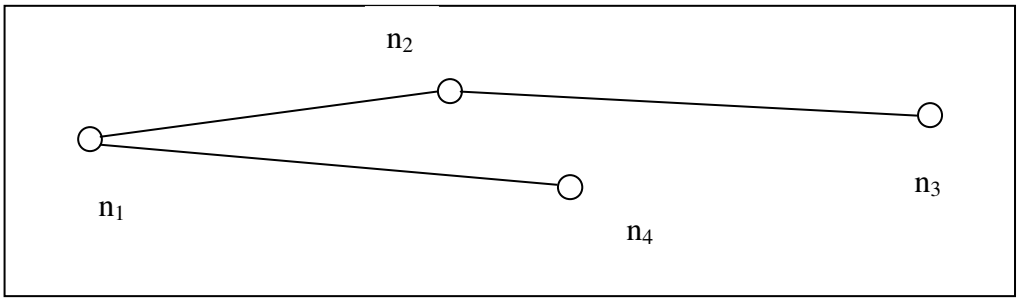
La integración de los nodos en las diferentes rutas deberá hacerse de una manera lógica, es decir, tratando que la longitud de las mismas sea siempre lo menor posible, y por lo tanto siendo fieles al objetivo de minimizar la longitud de las rutas.

Por ejemplo: si el nodo n_4, ha sido hallado como el nodo que produce un mayor valor de la función objetivo al ser unido a la ruta i, será importante que su colocación en la ruta sea la correcta. Además, no hay que olvidar que la propia función objetivo requiere que calculemos con exactitud el incremento de longitud que se produce en la ruta.

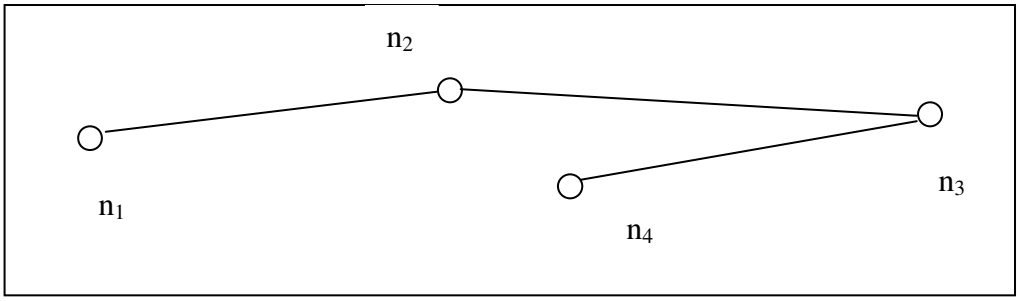


Las soluciones de unir n_4 al primero o al último nodo de la ruta, muy típicas en algunos heurísticos voraces, no proporcionan una opción que satisfaga nuestros intereses:

- Uniendo n_4 al primer nodo de la ruta, n_1.

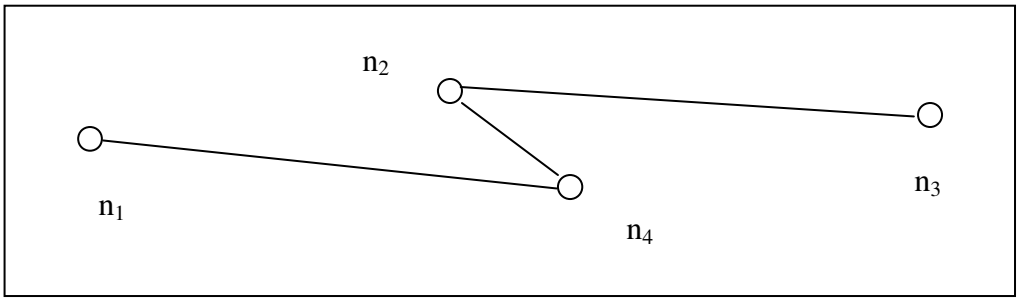


- Uniendo n_4 al último nodo de la ruta, n_3.

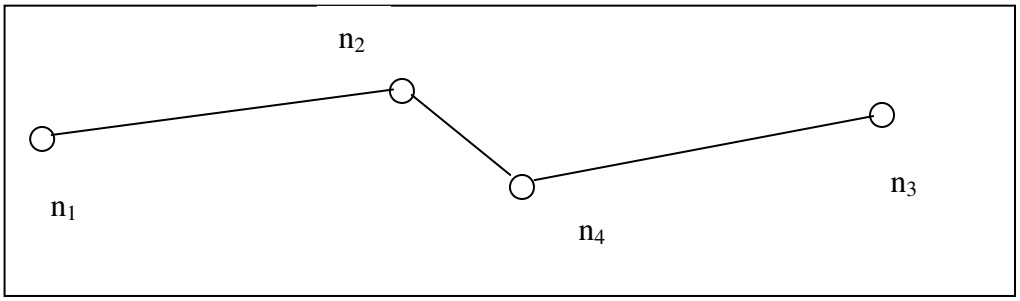


Es evidente que la longitud mínima de ruta se obtendrá uniendo n_4 a n_2, pero habrá que averiguar en que orden.

- Ruta n_1-n_4-n_2-n_3.



- Ruta n_1-n_2-n_4-n_3.



A simple vista, se observa como la longitud de ruta del segundo caso es menor que la del primero, y por lo tanto ésta será la nueva estructura de la línea.

En esta mera introducción, no hemos mencionado la dificultad que supone el hecho de no permitir que entre los nodos "originadores" de la ruta haya más de **n** nodos. Éste, también deberá ser un detalle tenido en cuenta a la hora de insertar los nodos en la ruta, como se verá en el siguiente apartado.

2.5 RESOLUCIÓN DEL ALGORITMO. FUNCIONES Y PROGRAMACIÓN.

Siguiendo la estructura planteada en los apartados anteriores, es el momento de analizar con más detalle el algoritmo. Las funciones en que ha sido dividido y la programación de las mismas.

Los datos que obligatoriamente han de ser conocidos para poder proceder a la ejecución del algoritmo implementado son:

- Situación geográfica de cada uno de los nodos. Estos serán identificados por un número.
- Número de rutas que se desean crear.
- Matriz de demanda de viajes Origen-Destino.
- Matriz de distancias entre todos los nodos.
- Longitud máxima de ruta permitida.
- Número máximo de nodos permitidos entre los nodos "originadores" de la ruta.
- Porcentaje de viajeros que estarían dispuestos a viajar a su destino, realizando un transbordo entre rutas secundarias (rutas de autobuses).
- Porcentaje de viajeros que estarían dispuestos a viajar a su destino, realizando un transbordo entre una ruta secundaria (ruta de autobús), y otra primaria (línea de metro).
- Nodos que componen la línea principal (de metro).

Una vez introducidos estos datos (más adelante se indicará el modo), la primera función en ejecutarse es '*general'.* Examinémosla con más detenimiento.

Función 'general' (ver diagrama sinóptico en el anexo).

Esta función actúa como controladora de todo el proceso. Recibirá como argumentos todas las variables expuestas anteriormente, y el nombre de los archivos donde se encuentran las matrices de distancias y flujos entre nodos; y será la encargada de devolver los valores que nos interesan que son:

- 'ruta': Matriz de vectores compuestos por los nodos que componen cada ruta, y en el orden en que son unidos; lo que nos dará en si la estructura de las rutas.
- 'carga': Vector que devuelve la carga de cada una de las rutas creadas.
- 'Longitud_rutas': Vector que devuelve la longitud de cada una de las rutas creadas.

La matriz de flujos entre los diferentes nodos no es simétrica, es decir, no tiene porque ser igual la demanda de viajes con origen en el nodo **i** y destino en el nodo **j**, que los viajes con origen en **j** y destino en **i**:

$$f_{i,j} \neq f_{j,i}$$

siendo $f_{i,j}$,la demanda de viajes con origen en el nodo i y destino en el nodo j.

Sin embargo, resulta más cómodo trabajar con una matriz simétrica:

$$F_{i,j} = f_{i,j} + f_{j,i}$$

Puesto que al integrar los nodos **i** y **j** dentro de una misma ruta, se verán absorbidos tanto los pasajeros que desean ir de **i** a **j**, como los que demandan el servicio de **j** a **i**. Es por esto que trabajar con la matriz $\mathbf{F_{i,j}}$ puede aliviar, en cierta medida, el número de operaciones a realizar durante la ejecución del programa. Así pues lo primera operación que realiza esta función es la de hallar la matriz simétrica $\mathbf{F_{i,j}}$. Por supuesto, se sobreentiende que la diagonal de la misma será nula, puesto que se considerará que no existe demanda de viajes de un nodo a si mismo.

A continuación se hará una llamada a la función *'máximos'*, que calculará cuales son los pares de nodos que van a iniciar la creación de las rutas.

Es importante destacar la creación del vector *estado*. Éste, inicialmente será un vector de 'unos', tantos como rutas pretendan ser creadas. Si en la ruta **i** pueden seguir añadiéndose nodos, porque no se ha alcanzado la longitud de ruta máxima, diremos que la ruta **i** está "abierta", y por lo tanto vendrá representada por un 'uno' en la posición **i** del vector *estado*. Si fuera imposible añadir más nodos a la ruta, porque se supera la longitud de ruta máxima permitida, el correspondiente elemento que identifica a la ruta en el vector de estados, pasaría a ser un 'cero'. Diremos que la ruta **i** está "cerrada".

Mientras que haya rutas "abiertas", la presente función estará llamando a la función 'cálculo' que es la encargada de calcular qué nodo debe ser unido a qué ruta.

Una vez que todas las rutas hayan sido "cerradas", se dará por concluido el cálculo de rutas y se pasará al cálculo de la carga que soporta cada una de ellas.

Pasemos a continuación a explicar un poco más en profundidad el resto de funciones programadas, y que ya han sido mencionadas.

Función 'máximos'. (ver diagrama sinóptico en el anexo).

Esta función es llamada desde la función 'general'.

Sus argumentos de entrada son:

- La matriz de flujos: $\mathbf{F_{i,j}}$.
- El vector que define la ruta de metro: **ruta_metro**.
- La matriz de distancias entre los nodos: $\mathbf{L_{i,j}}$.
- El número de rutas a crear: **N**.
- La longitud máxima de ruta, permitida: **L**.
- El número máximo de nodos permitidos entre el par de nodos originadores de la ruta: **n_entre**.

Esta función será la encargada de encontrar los pares de nodos entre los que se produce un mayor flujo de viajeros. Y por lo tanto, según nuestro heurístico, la pareja de nodos entre los que se iniciará la creación de las rutas.

Los pasos que sigue la función persiguen los objetivos marcados para esta función que ya fueron explicados extensamente en el apartado 2.4.

Es interesante, la constante creada en las primeras líneas de la función: **n_posible_rutas**.

$$n\_posible\_rutas = \frac{f \cdot (f-1)}{2},$$

que indica el número de rutas que pudieran ser iniciadas, en principio, teniendo en cuenta el número de nodos existente (f).

Así, la función continuará buscando parejas de nodos entre los que se produzca un flujo máximo, mientras no se hayan alcanzado los **N** pares principales, y no se haya llegado al límite ***n_posible_rutas*** de rutas que teóricamente podrían ser creadas. Este límite vendrá representado por el índice **i** que irá aumentando cada vez que es desechado un par como generador de rutas.

Mientras las condiciones anteriores no se cumplan, la función hallará el máximo de la matriz $\mathbf{F_{i,j}}$, y comprobará su posición dentro de la matriz, (fila,columna), o lo que es lo mismo: **(nodo_origen,nodo_destino)**. Pero antes de dar por bueno este par, se comprueba que la distancia entre ellos no supere la longitud máxima de ruta permitida (**L**), y que además **nodo_origen** y **nodo_destino** no son nodos incluidos (ambos) en la ruta principal (metro).

Pero, tal y como se comentó en la *Introducción al algoritmo*, si el índice **j**, que representa el número de pares hallados hasta el momento, es superior a uno, antes de dar por bueno el nuevo par $\mathbf{(nodo\_origen_{j+1},nodo\_destino_{j+1})}$ hallado, habrá que estudiar si puede ser integrado dentro de alguna ruta (formada hasta el momento por sólo un par de nodos) ya iniciada.

El proceso que se sigue es el siguiente:

Tomando una a una las **j** rutas iniciadas hasta el momento, se analiza si **nodo_origen** puede ser introducido en ellas sin superar la longitud **L**. Si la respuesta es afirmativa, se estudia si **nodo_destino** puede ser introducido en la misma sin sobrepasar la longitud **L**. Si la respuesta vuelve a ser afirmativa, se guardará en la posición **j** de la matriz de vectores **ruta_posible**, la nueva ruta obtenida al añadir el último par $\mathbf{(nodo\_origen_k,nodo\_destino_k)}$ al par $\mathbf{(nodo\_origen_j,nodo\ destino_j)}$; así como la longitud de la nueva ruta en la posición **j** del vector **Lruta_posible**. Esto se hace así, porque pudiera darse el caso en el que el último par hallado pudiera ser integrado dentro de varias rutas anteriores. En tal caso nos quedaremos con aquella ruta de la matriz **ruta_posible** que tenga una **Lruta_posible** menor. Así pues, solo quedará modificada la matriz de vectores **ruta** en la posición en que hallamos anteriormente, manteniéndose el resto de pares igual.

En este caso en concreto, en el cual hemos integrado el último par de flujo máximo hallado dentro de una ruta creada anteriormente, sólo se verá incrementado el índice **i**, pero no así el índice **j** que indica el número de rutas iniciadas (y que no puede superar el número **N**).

Observar, que cada vez que es analizado un par (nodo_origen,nodo_destino), su correspondiente valor en la matriz $\mathbf{F_{i,j}}$ es

igualado a cero, con la intención de que no vuelva a salir como par de flujo máximo en la siguiente iteración.

Cuando termine de implementarse la función, se habrán iniciado las **N** rutas que intentamos crear. Los argumentos que devuelve la función son:

- Un vector con los nodos de origen de cada una de las rutas: **nodo_origen**.
- Un vector con los nodos de destino de cada una de las rutas: **nodo_destino**.
- Una matriz de vectores que contiene los nodos que inicialmente componen cada una de las rutas (los nodos generadores de la ruta, más nodos que hayan podido ser integrados en la misma): **ruta**.
- Un vector con los pasajeros que demandan viajes entre los dos nodos originadores de las rutas: **pasajeros**.

Es importante dejar claro una cuestión que ya ha sido mencionada con anterioridad, y es que, tanto el **nodo_origen** como el **nodo_destino** (que son los nodos que provocan que se inicie una ruta entre ambos), durante la ejecución del algoritmo puede perder su condición de nodos inicial y final de la ruta, respectivamente. Es decir, podrán ser añadidos nodos delante del **nodo_origen**, o detrás del **nodo_destino**.

Por último, reseñar que esta función debe recurrir a otra función fundamental en la programación del algoritmo: ***Longitud_ruta,*** que obtiene la posición en que deben ser insertados los nodos dentro de una ruta, y cómo queda la longitud de la misma. Esta función será explicada más adelante.

Función 'cálculo'. (ver diagrama sinóptico en el anexo).

Aunque todas las funciones tienen su importancia, es ésta la que, quizás, tenga un mayor peso dentro de la programación, no ya solo por su complejidad, sino porque será la encargada de ir creando las rutas, uniendo en cada iteración el nodo a la ruta correspondiente que aporte un mayor valor de la función objetivo.

El desarrollo y programación de esta función ha sido bastante complicada, al permitirse el hecho de que las rutas puedan interseccionar entre si en varios nodos comunes, y por lo tanto puedan producirse transbordos de una ruta a otra.

Veamos con detenimiento las características de esta función, que será llamada desde la función ***general*** de manera reiterativa, hasta el momento en que todas las rutas se encuentres "cerradas" (el vector **estado** es enteramente nulo).

Los argumentos de entrada a la función son:

- Matriz de flujos: $\mathbf{F_{i,j}}$.
- Vector de estados de las rutas: **estado**.
- Vector con los nodos de origen de las rutas: **nodo_origen**.
- Vector con los nodos de destino de las rutas: **nodo_destino**.
- Matriz con las rutas: **ruta**.
- Matriz de distancias entre los nodos: $\mathbf{L_{i,j}}$.
- Número de rutas a crear: **N**.
- Longitud máxima de ruta permitida: **L**.
- Porcentaje de personas dispuestas a transbordar entre dos rutas secundarias (rutas de autobús), para llegar desde un nodo de origen a otro nodo de destino: **alfa_bus**.
- Porcentaje de personas dispuestas a transbordar entre una ruta principal (metro) y una secundaria (autobús), para llegar desde un nodo de origen a otro nodo de destino: **alfa_metro**.

- Vector que contiene la longitud de las rutas: **Longitud_rutas**.
- Máximo número de nodos permitidos entre los dos nodos generadores de la ruta: **n_entre**.

El proceso que sigue la función es el siguiente:

Comienza tomando el segundo vector de la matriz de rutas (el primer vector corresponde a la línea principal –metro – que es fija), y comprueba que esta ruta no está "cerrada", en tal caso estará abierta a incluir nuevos nodos.

No se establece ninguna limitación en cuanto los nodos que pueden ser unidos a la ruta, es decir, todos los nodos en estudio son candidatos, en principio, a ser unidos a la ruta en estudio.

Se van tomando uno a uno todos los nodos candidatos (de aquí en adelante, se denominará **'nodo'** al nodo candidato), y se va obteniendo el valor de la función objetivo al ser añadido dicho nodo a la ruta en estudio (que normalmente denominamos **ruta i**).

Ahora bien, como una de las restricciones es la longitud máxima de ruta, la primera comprobación que haremos es que al introducir el nodo candidato **'nodo'**, en la **ruta i**, no se supere el valor **L** (para ello habrá que llamar a la función *Longitud_ruta* que se detallará más tarde). Si es así seguiremos con la ejecución de la función con la intención de hallar el valor de la función objetivo; en caso contrario, el elemento **FO(nodo,i)** de la matriz que se irá creando, y que representa el valor de la función objetivo de la pareja **(nodo,i)**, tomará un valor nulo(**FO(nodo,i) = 0**), y pasará a analizarse el siguiente nodo candidato.

Como sabemos, la función objetivo que tratamos de maximizar es:

$$F.O = \frac{\Delta F_{`nodo´,i}}{\Delta L_{`nodo´,i}}$$

siendo:

$\Delta F_{`nodo',i}$ el incremento de viajeros que se produce al introducir 'nodo' en la ruta i.

$\Delta L_{nodo',i}$ el incremento en la longitud de la ruta i, al integrar 'nodo' en la misma.

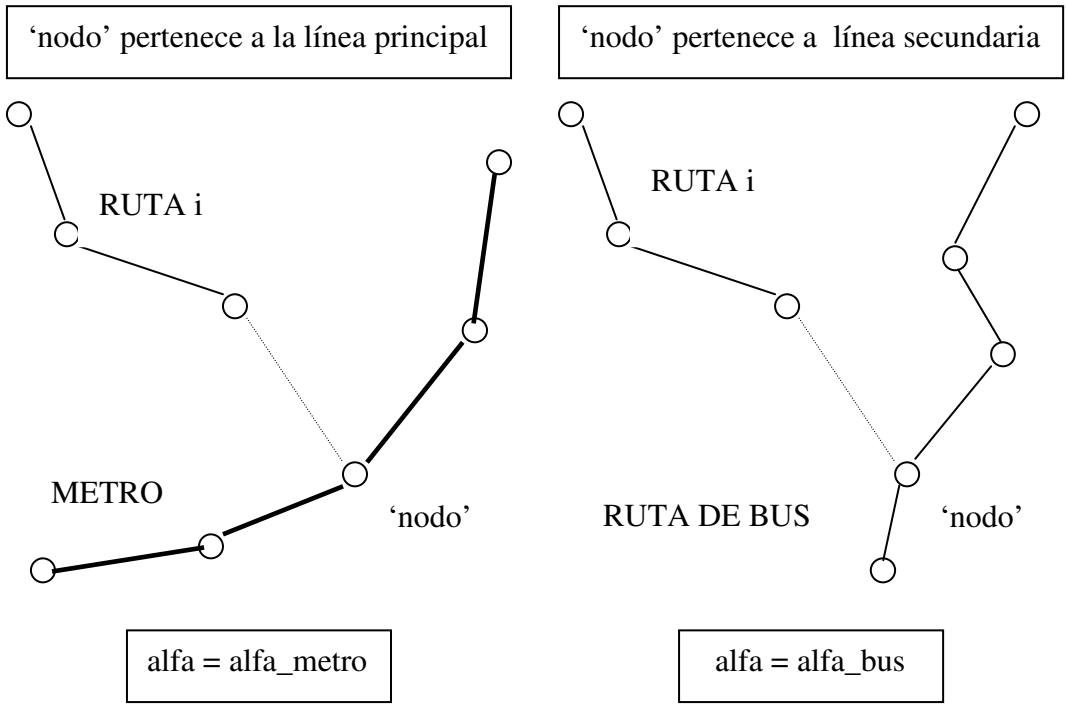
Por lo tanto, lo primero en lo que estamos interesados es en hallar el incremento en el número de viajeros que se produce al añadir **'nodo'** a la **ruta i** en estudio.

Son múltiples los casos posibles, y por lo tanto habrá que tener en cuenta todos ellos para obtener el incremento en el flujo de viajeros.

Una vez seleccionada la **ruta i** y el nodo candidato (**'nodo'**), se irán tomando una a una todas las rutas restantes (empezando por la **ruta 1**, que corresponde a la línea principal – metro-), y se irá diferenciando si **'nodo'** pertenece o no a esa ruta (**ruta h,** distinta de la **ruta i**).

Caso 1: 'nodo' pertenece a alguna de las rutas.

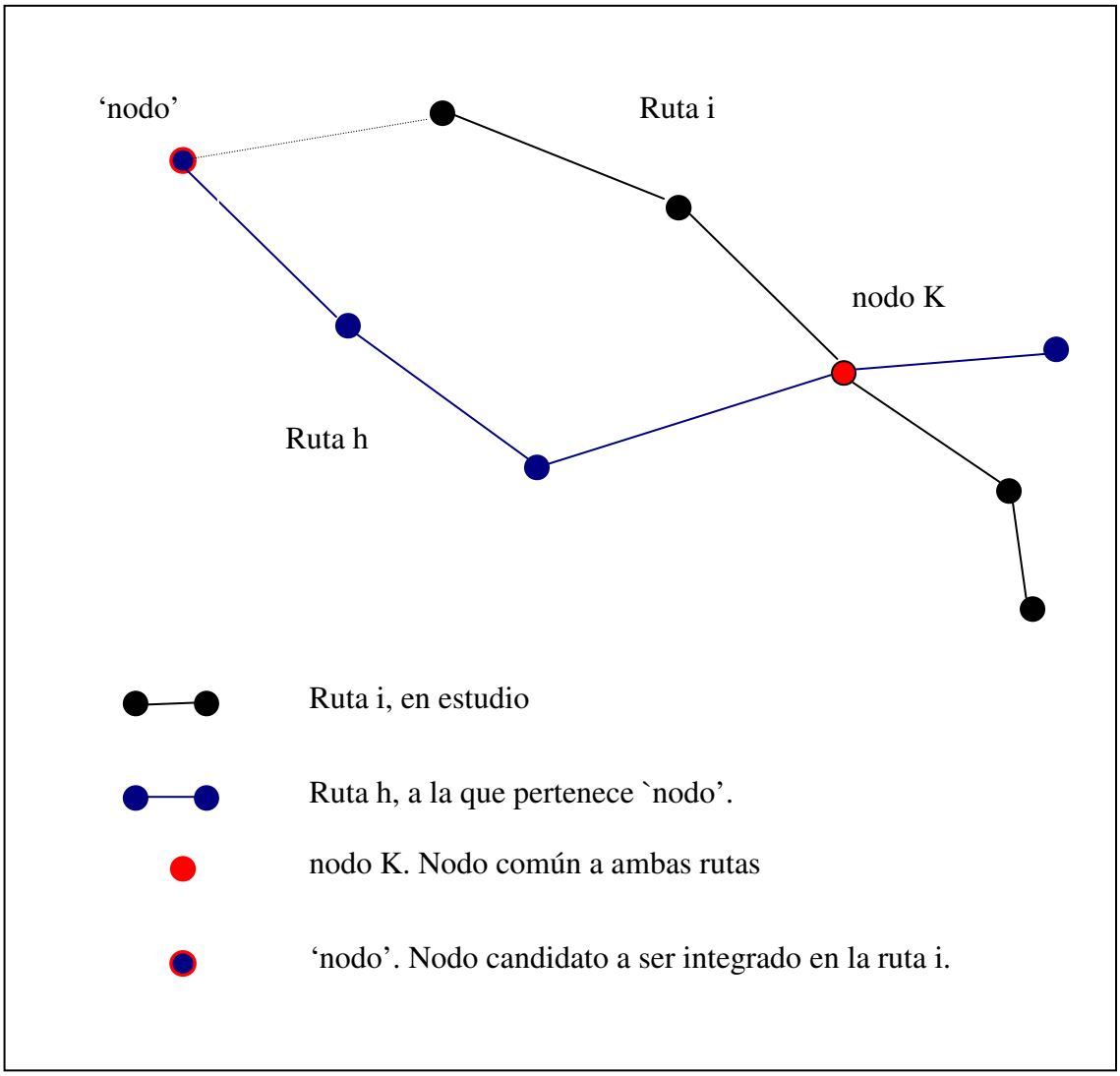
Si **'nodo'** pertenece a la **ruta h**, habrá que contabilizar el transbordo de pasajeros que se producen desde la ruta en estudio (**ruta i**) hacia los nodos de la ruta a la que pertenece **'nodo'** (**ruta h**). Como ya se ha mencionado hay diferentes tipos de transbordo, según se hagan entre dos líneas secundarias (de autobús) o una principal (metro) y otra secundaria. Por lo tanto, lo primero es saber si **'nodo'** pertenece ya a alguna ruta, y en tal caso si ésta (**ruta h**) es secundaria (de autobús), o principal (de metro), para así asignar un porcentaje de transbordo: *alfa_bus* o *alfa_metro*.



Siendo alfa_bus < alfa_metro, es decir, es menor la demanda de viajes satisfecha entre dos nodos, cuando se ha de realizar un viaje por ruta secundaria (autobús), y luego transbordar hacia otra línea secundaria para llegar al destino; que si el transbordo se realiza hacia una línea principal (de metro), o desde una principal a otra secundaria.

Pero para contabilizar el transbordo entre las rutas que se cruzan en **'nodo'**, estas no deben cruzarse con anterioridad en otro nodo común a ambas, pues en tal caso, se considerará que los posibles viajes con transbordo, entre los nodos de las diferentes líneas ya han sido computados.

Quedará perfectamente claro con el siguiente ejemplo:



En este ejemplo, el nodo candidato a ser integrado dentro de la **ruta i**: **'nodo'**, ya pertenece a otra ruta, la **ruta h**. En condiciones normales, esto supondría que el flujo de viajeros se vería incrementado en los viajeros que transbordan en 'nodo' para viajar entre los nodos de la ruta i y los nodos de la ruta h (además del incremento de "viajes directos", que después se comentarán). Sin embargo, estos viajes no se contabilizarán, puesto que los viajeros que pretendían viajar entre nodos de ambas rutas, ya tienen la posibilidad de hacerlo transbordando en el **nodo K**, y por lo tanto ya se consideran contabilizados.

Así pues, situándonos dentro del caso 1, en el cual el nodo candidato 'nodo' pertenece a alguna de las rutas (pongamos ruta_h), la siguiente

pregunta a hacerse es si ésta, **ruta h**, y la ruta actualmente en estudio, **ruta i** ya tenían algún nodo en común. Si es así, no habrá que sumar al incremento de flujo que estamos hallando, los transbordos entre ambas rutas **i** y **h**.

Pero si la **ruta i** y la **ruta h** no se "cortaban" con anterioridad, entonces sí habrá que tener en cuenta los posibles transbordos entre ambas rutas. En tal caso, el proceso general es el de hallar el flujo entre todos los nodos de la **ruta i** y todos los nodos de la **ruta h** (exceptuando el nodo candidato 'nodo', pues ya también pertenece a la **ruta i**), afectados por el parámetro **alfa**. Aunque esto es algo muy general, puesto que ahora pueden empezar a aparecer las peculiaridades que dificultarán las operaciones:

Supongamos que la ruta que estamos estudiando es la **ruta 2**, y que el nodo candidato **'nodo'** pertenece a la **ruta 3.** Pero además, existe una **ruta 4** que interseccciona con la **ruta 2** en el nodo **P**, y con la **ruta 3** en el nodo **K** (ver figura en página 42).

En este caso, y como se ha comentado, deberá sumarse el flujo de viajeros que transbordan entre la **ruta 2** y la **ruta 3** (al añadir **'nodo'** a la **ruta 2**). Pero no habrá que contar los viajeros que viajen de **P** a la **ruta 3** realizando transbordo en **'nodo**', puesto que estos viajes ya pueden realizarse a través de la **ruta 4**, transbordando en **K**. Tampoco se deberá contabilizar como un incremento en la demanda satisfecha los viajes realizados por transbordo en **'nodo'** desde **K** a la **ruta 2**, ya que éstos pueden ser realizados a través de la **ruta 4**, por transbordo en **P**.

El algoritmo procede de la siguiente manera:

1. Calcula el flujo de viajeros que transbordan entre la **ruta 2** (ruta i) y la **ruta 3** (ruta h), al unir **'nodo'** a la **ruta 2**, y lo guarda en la variable **suma1**.

   En el ejemplo: **suma1 =** $\mathbf{F_{ruta2,ruta3-\{'nodo'\}}}$

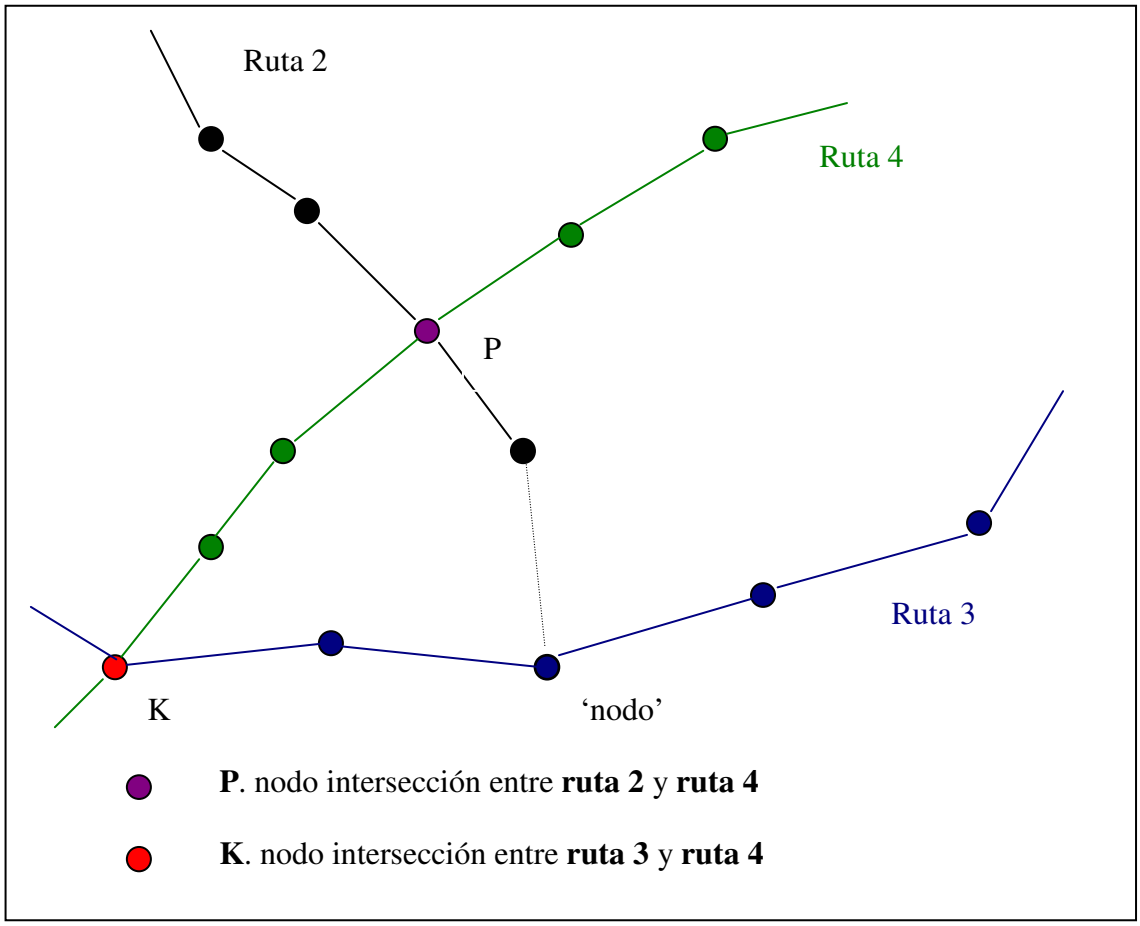
2. Toma una a una las rutas existentes (**ruta_t**, con $t \neq i \neq h$) y analiza si se cortan con la **ruta i** (ruta 2)**.** En caso afirmativo, comprueba si, además, esa ruta interseccciona con la **ruta h** (ruta 3). Si la respuesta

a las dos preguntas es afirmativa, guarda en la variable **sum_com1**, el flujo de viajeros entre los nodos intersección de la **ruta_i** y **ruta_t** (nodo **P**)**,** y la **ruta h**, sin incluir 'nodo'**,** en el ejemplo anterior $\mathbf{F_{P,ruta\_3}}$; y los viajes entre el nodo K (intersección de ruta_h y ruta_t) y la ruta_i (excluyendo el nodo P, para no restar dos veces); afectados por el mismo valor de **alfa** utilizado en el paso 1.

En el ejemplo expuesto:

$$\mathbf{sum\_com1 = F_{P,ruta\_3\text{-}\{`nodo\}} \cdot alfa + F_{K,ruta\_2\text{-}\{P\}} \cdot alfa.}$$

Este paso se realiza hasta revisar todas las rutas (para todo t ≠ i ≠ h), de modo que se irá acumulando en **sum_com1** un valor que después deberá ser restado a **suma1**.



3. Comprueba que no se vuelva a sumar el transbordo entre la **ruta i** en estudio (en el ejemplo, **ruta 2**), y nodos de otras rutas, que ya fueron sumados con anterioridad.

   Siguiendo con el ejemplo anterior. Supongamos que continúo estudiando el aumento de viajeros que me produce el integrar **'nodo'** en la **ruta 2**. Y que al hallar las rutas que contienen al nodo candidato, después de la **ruta 3** aparece la **ruta 5**. Esta ruta no tiene nodos que se unan a la **ruta 2** de forma directa mediante otras rutas, pero sí intersecciona con la **ruta 3**, además de en **'nodo'**, en otro nodo **Q.** (ver figura en página 44).

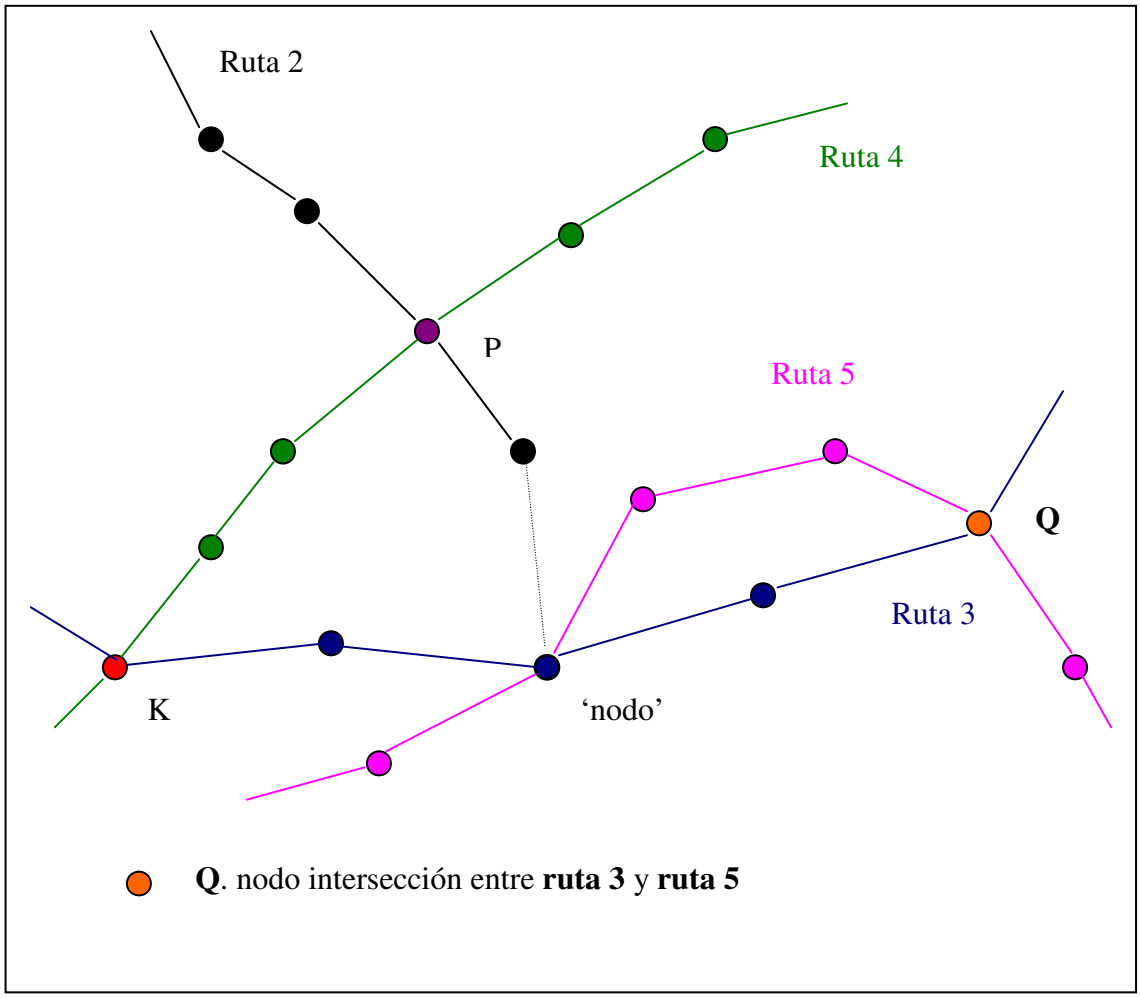
   El programa procederá a contabilizar los transbordos entre la **ruta 2** y la **ruta 5**, pero no deberán ser tenidos en cuenta los que se realizan entre la **ruta 2** y **Q**, pues estos ya fueron contabilizados anteriormente cuando se estudió el transbordo entre **ruta 2** y **ruta 3**.

   Esto queda solucionado en la programación con el vector **nx** el cual va acumulando los nodos hacia los que ya se ha contado el transbordo desde la **ruta i** (**ruta 2**, en el ejemplo). Así pues, se obtienen los nodos que son comunes a **nx** y a la **ruta 5** (en este caso, el nodo **Q**), y se guardará en la variable **sux2** el flujo entre la **ruta 2** y el nodo **Q**, afectado por el **alfa** correspondiente.

   En el ejemplo: **sux2 = $F_{ruta2,Q}$ x alfa**

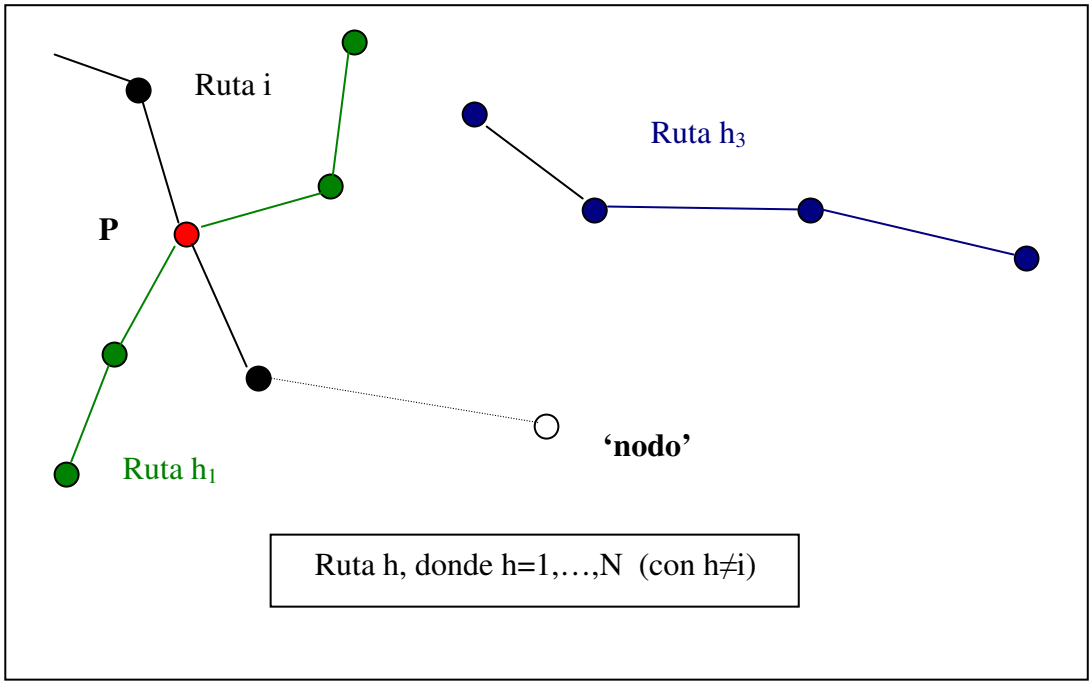
4. Finalmente, se le restará a la variable **suma1**, **suma_com1** y **sux2**.

   suma1 = suma1 – sum_com1 – sux2



Sin ánimo de complicar más la explicación, es necesario comentar que se ha tenido un especial cuidado en el cálculo del incremento de flujo, y que pudieran darse otros casos que también son tenidos en cuenta, como pudiera ser que la **ruta 5** tuviera otro corte con la **ruta 3**, en el mismo punto **K**, que se encontraba unido con la **ruta 2.** Habrá que tener especial cuidado en no restar dos veces el flujo $\mathbf{F_{P,K}}$, una en **sum_com1** y otra en **sux2.** Es por esto, que el vector **nx** está continuamente revisándose, como se puede observar en la programación del mismo (ver anexo).

Caso 2: 'nodo' NO pertenece a la ruta h.



El caso opuesto, es que **'nodo'** no perteneciera a la **ruta h**.

Entonces pueden ocurrir dos subcasos:

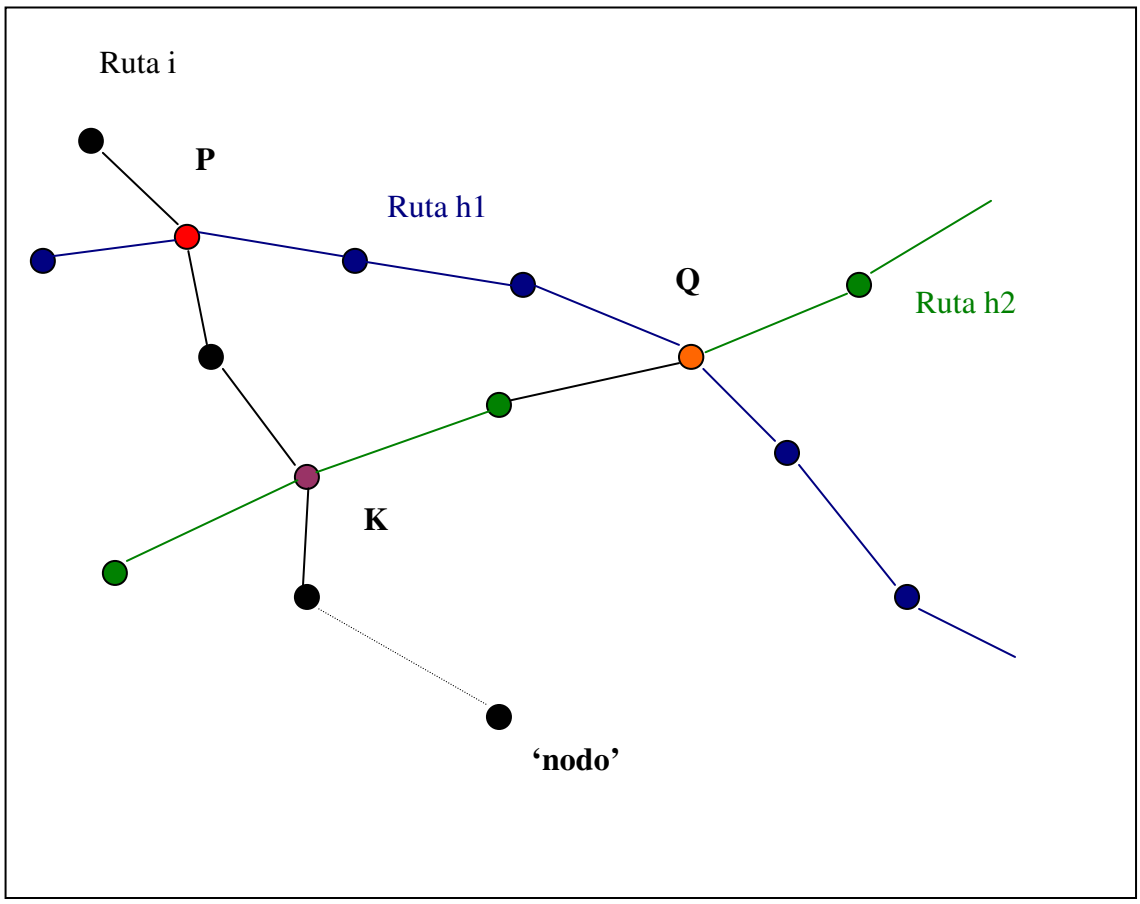
2.1) Que la **ruta i** y la **ruta h**, no se interseccionen en ningún nodo (como ocurre en el gráfico anterior entre **ruta i** y **ruta h_3)**.

En tal caso, no habrá que sumar ningún transbordo entre ambas rutas.

2.2) Que la **ruta i** y la **ruta h**, se interseccionen en algún nodo (como ocurre con la **ruta i** y la **ruta h_1**, que se cruzan en el nodo **P** -ver página 46-. En tal caso, se contabilizarían los transbordos entre **'nodo'** y todos los nodos de la **ruta h_1**, exceptuando el nodo **P**. Como ocurría en el caso 1, ya explicado, un vector (**nx**) guardará todos aquellos nodos hacia los que ya se han contabilizado los viajes por transbordo desde **'nodo'**, con la intención de no sumar doblemente estos flujos cuando sean analizadas otras rutas h.

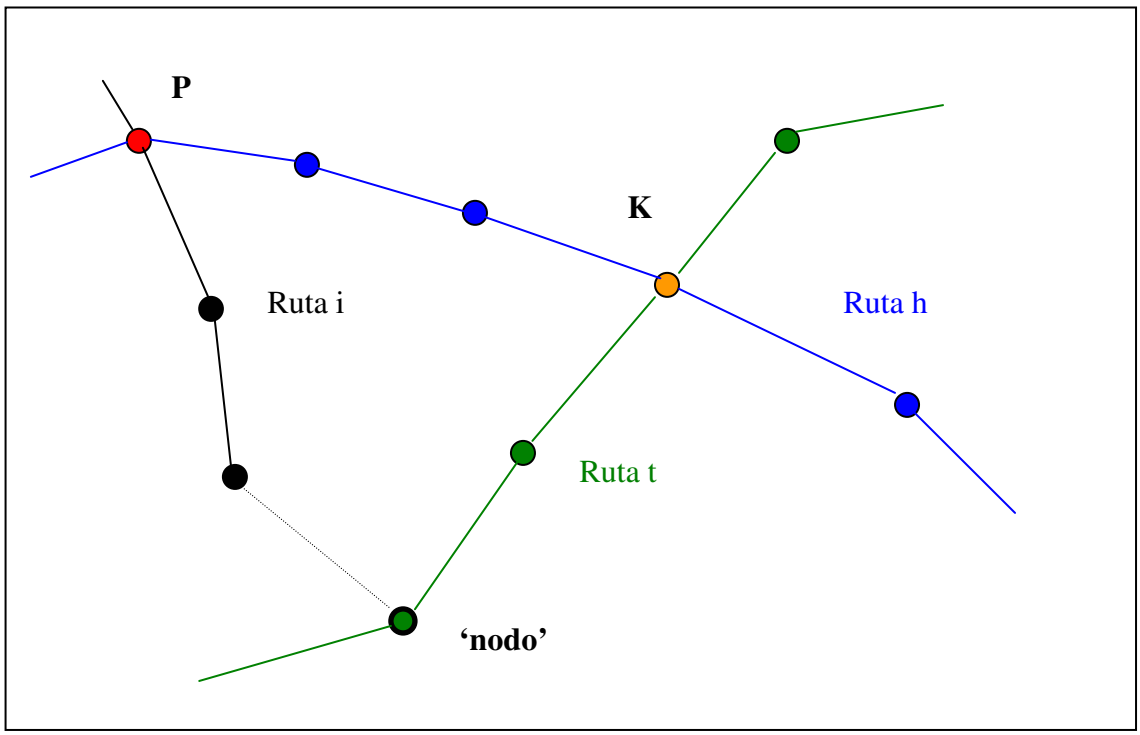
Observar en la siguiente figura cómo cuando estoy contabilizando los transbordos en **K** de los viajeros que demandan el viaje desde

'nodo' hacia los nodos de la **ruta h_2** (al unir **'nodo'** a la **ruta i**), no se debe computar los transbordos en **K** desde **'nodo'** al nodo **Q,** pues este incremento de flujo que se produciría en la **ruta i** al añadir **'nodo'** a la misma, ya se contabilizó al sumar los transbordos hacia la **ruta h_1**, a la que también pertenece el nodo **Q**.



Al igual que en el *caso 1*, hay que estudiar el resto de rutas, pues pudiera haber alguna **ruta t**, que cortase a la **ruta h** y que contuviera al nodo candidato **'nodo'** (ver figura en página 47). En este caso, no hay que incrementar el flujo de viajeros de **'nodo'** que van por la **ruta i,** para transbordar en P hacia los nodos de la **ruta h**, puesto que ya lo podían hacer viajando por la **ruta t**, y transbordando en **K** hacia la **ruta h**.

Lógicamente, esto es así porque lo único que pretende el algoritmo es hallar el incremento de viajeros en general, no el de una ruta en particular. Poco nos interesa ahora, si la carga de viajes entre **'nodo'** y la **ruta h,** se repartirá a medias entre la **ruta i** y la **ruta t**. Esto, sí será tenido en cuenta en la última función que veremos, que halla la carga total de las rutas.



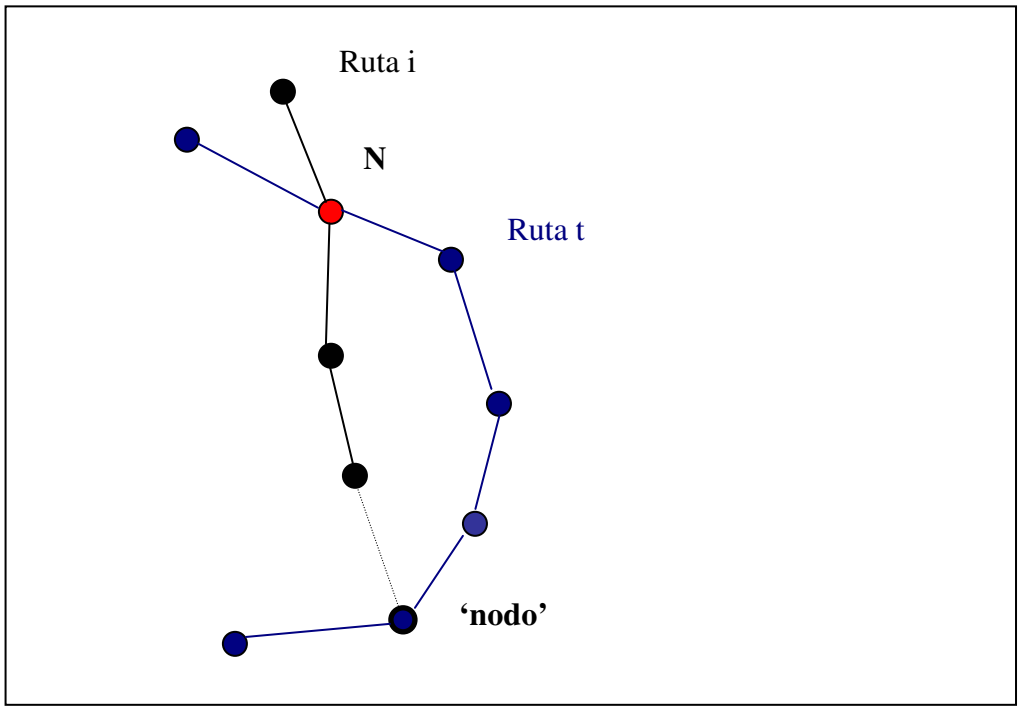
No entraremos en más detalle, por no complicar más la explicación de la función, pero observando la programación de la misma (incluida en el anexo), se observa como se acumulará en la variable **suma3** todos los viajes con transbordo expuestos en este ***caso 2.***

Finalmente, una vez recorridas todas las rutas (**ruta h,** distintas de la ruta en estudio, **ruta i)**, para poder analizar los transbordos que provoca la unión de el nodo candidato **'nodo'** a la **ruta i,** se habrán obtenidos las variables **suma1** y **suma3.** La suma de ambas me indica el número total de viajeros captados que viajan realizando algún transbordo, a consecuencia de unir el nodo candidato **'nodo'** a la **ruta i**.

trans(nodo,i) = suma1 + suma3

Hasta el momento, sólo hemos hablado del incremento en el flujo de viajeros producidos por la posibilidad de transbordar, pero evidentemente, al unir un nuevo nodo a una ruta, se produce un incremento de viajeros que viajarán entre los nodos de la ruta y el nuevo nodo, es lo que hemos llamado incremento del **flujo directo.**

Aunque, nuevamente, habrá que analizar si hay alguna otra ruta (**ruta t**), que una algún nodo de la **ruta i** (por ejemplo: el nodo **N**), con **'nodo'**, en cuyo caso, no se sumará al incremento de flujo los viajes entre **N** y **'nodo'**, pues ésta era ya una demanda cubierta.



Al añadir **'nodo'** a la **ruta i,** se captará el flujo de viajeros que demandan viajar entre los nodos de la **ruta i** y **'nodo'**, sin computar los viajes entre **N** y **'nodo'**, que se encuentran unidos por la **ruta t**.

Llegado a este punto, tenemos el incremento de viajes que se obtiene al unir el nodo candidato (**'nodo'**) a la **ruta i**:

total(nodo,i) = directo(nodo,i) + trans(nodo,i)

Una vez que se haya analizado toda la lista de nodos candidatos a ser unidos a la **ruta i,** se procederá a tomar la siguiente ruta que no se encuentre cerrada, y se operará de la misma manera, estudiando qué incremento de viajeros supone el unir cada uno de los nodos candidatos (todos menos los pertenecientes a la misma ruta) a ésta, recientemente elegida y renombrada **ruta i**.

El resultado final de este proceso, será la obtención de una matriz de tantas filas como nodos y tantas columnas como rutas. Cada elemento **(nodo,i)**, indicará el incremento que se produce en el flujo de viajeros al unir ese nodo con la ruta especificada por el índice **i**.

Ya se ha resuelto la parte más ardua del algoritmo, pero aún queda por obtener qué pareja **(nodo,i)** proporciona el mayor valor de la función objetivo.

Es por ello necesario, obtener el incremento que se produce en la longitud de la **ruta i** al unir **'nodo'** a la misma. Esto se hará con una llamada a la función **Longitud_ruta**, que será explicada posteriormente.

Finalmente se obtendrá la matriz **FO** de dimensión **f x N** (siendo **f** el número de nodos, y **N** el número de rutas).

Si alguna columna de esa matriz es enteramente nula, podremos considerar que la ruta que corresponde a esa columna estará cerrada, y por lo tanto se hará el elemento correspondiente del vector **estado** igual a cero (una ruta menos que deberá ser estudiada en la próxima iteración).

Si no todos los elementos de la matriz **FO** son nulos, se procederá a obtener el elemento máximo. La fila correspondiente de la matriz indica el nodo a añadir, y la columna, la ruta a la que se añade el nodo.

La llamada a la función **Longitud_ruta** no sólo proporciona la longitud, sino también la estructura óptima de la ruta. Ésta quedará almacenada en la matriz de vectores **ruta**, cuya actualización será devuelta, junto al **estado**, y la **Longitud_rutas** a la función ***general.***

La función ***general***, como se comentó al comienzo, volverá a llamar a esta función mientras queden rutas *abiertas*.

Función Longitud_ruta. (ver diagrama sinóptico en el anexo).

Esta función es llamada tanto desde la función ***maximos***, como desde ***cálculo***, y analizará en que forma será integrado el nodo candidato a la ruta a la que se unirá, tal que la longitud de la misma sea lo menor posible. Atenderá, además, que no sean insertados más de **n_entre** nodos entre los dos nodos originadores de la ruta.

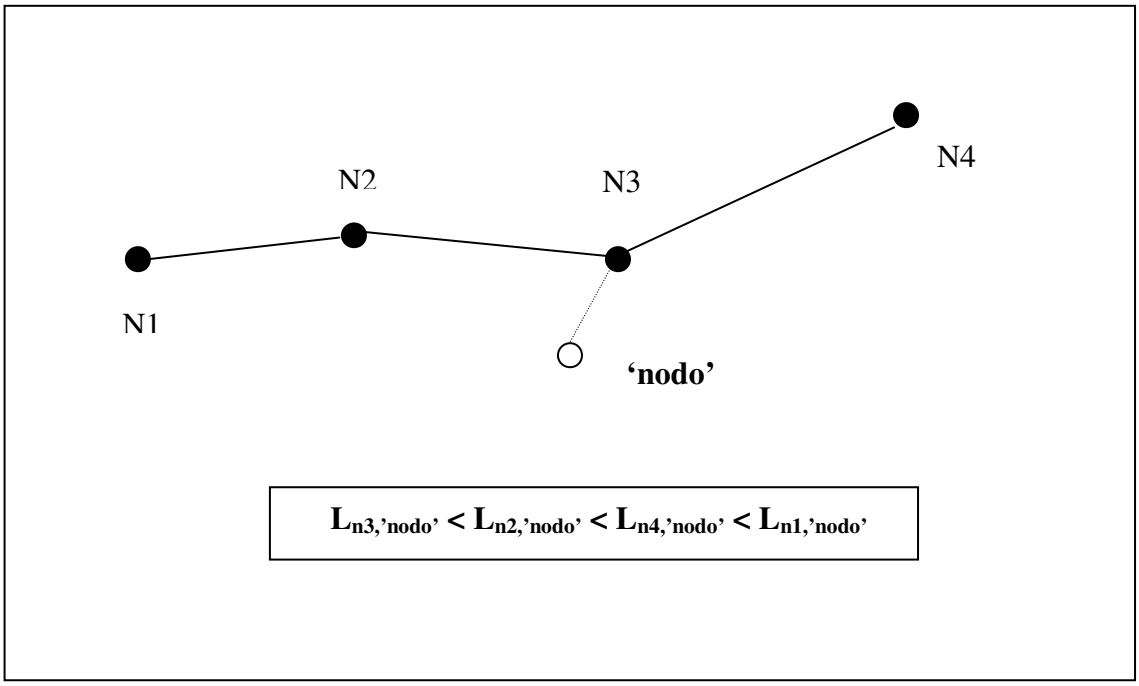
Los argumentos de entrada son:

- El número identificativo de la ruta que se desea modificar: **i**.
- La ruta que se desea modificar: **rut**.
- El nodo que se quiere introducir en la ruta: **nodo**.
- La matriz de distancias entre los nodos: $\mathbf{L_{ij}}$.
- El vector con todos los nodos de origen: **nodo_origen**.
- El vector con todos los nodos de destino: **nodo_destino**.
- Número máximo de nodos permitidos entre los *nodos originadores* de la ruta: **n_entre.**

La forma de actuar general es la siguiente:

Se halla la distancia entre todos los nodos de la ruta y **'nodo'** ($\mathbf{l_{i,'nodo'}}$).

'nodo' será unido al nodo de la ruta que esté más cercano a él (y que desde ahora llamaremos **nodo_de_ruta**, como se hace en la programación de la función). Veámoslo en el ejemplo siguiente:



Se busca cuál es el nodo de la ruta que guarda una menor distancia con el nodo candidato (**'nodo'**). En este caso: **n3**, será el nodo al que se una **'nodo'** (**n3** es el **nodo_de_ruta)**. A continuación habrá que reestructurar los enlaces entre **n2,n3,n4** y **'nodo'**, siempre con el objetivo de minimizar la longitud de la ruta.

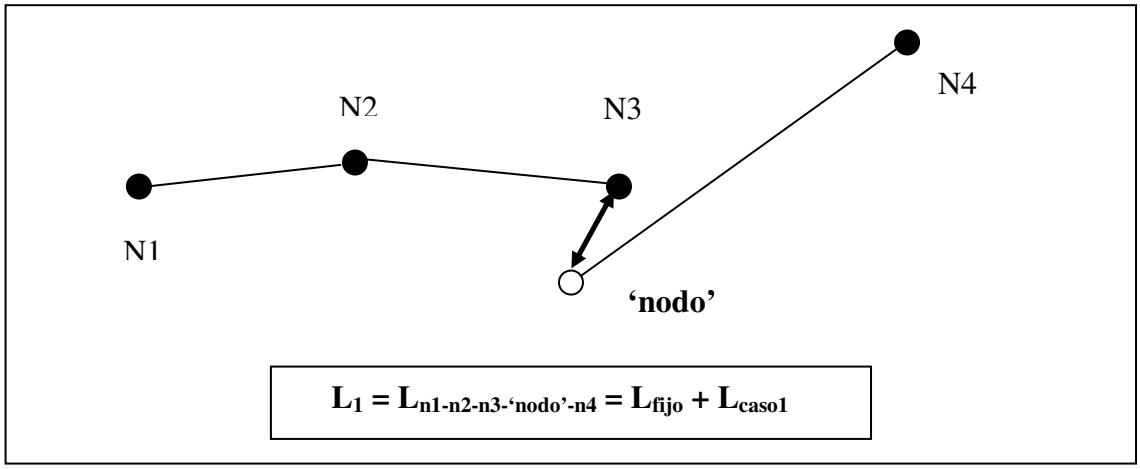
El algoritmo funciona hallando la longitud de la ruta que es fija: L_{fijo}, y la que es variable, según el caso: L_{caso1} y L_{caso2}. En el ejemplo siguiente:

- $L_{fijo} = l_{N1-N2} + l_{N3-'nodo'}$
- $L_{caso1} = l_{N2-N3} + l_{'nodo'-N4}$
- $L_{caso2} = l_{N2-'nodo'} + l_{N3-N4}$

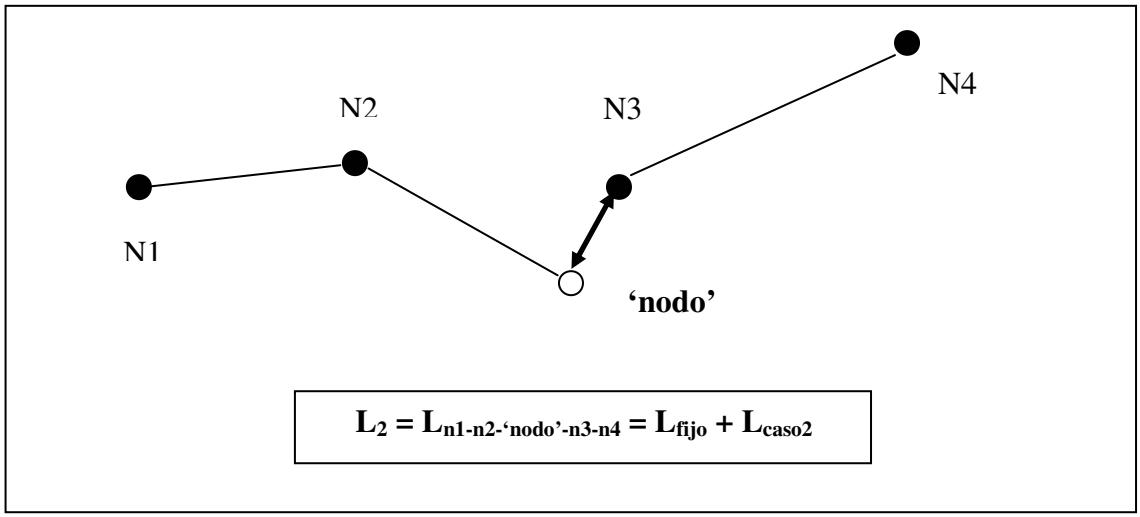
La ruta óptima será aquella que tenga menor *longitud variable*.

Veamos, pues, las posibilidades del caso más general:

1ª posibilidad:



2ª posibilidad:

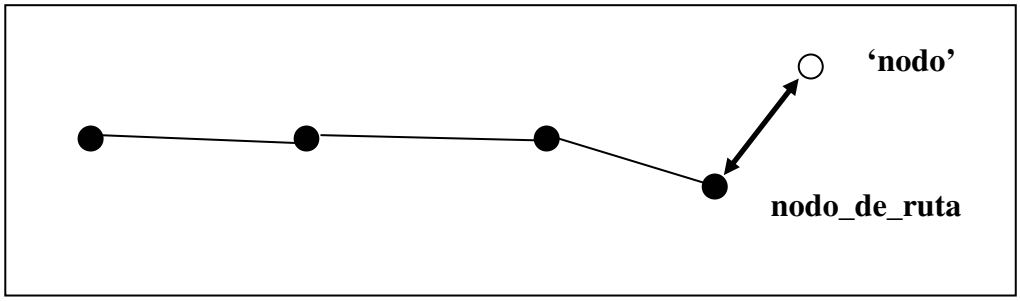


En este ejemplo, se observa claramente, que $L_2 < L_1$. Por lo tanto, la estructura correcta será la segunda.

Si bien, éste es un caso muy sencillo, sirve para entender el fundamento de la función.

Si el **nodo_de_ruta** fuera el **último nodo** de la ruta, habría también dos posibilidades:

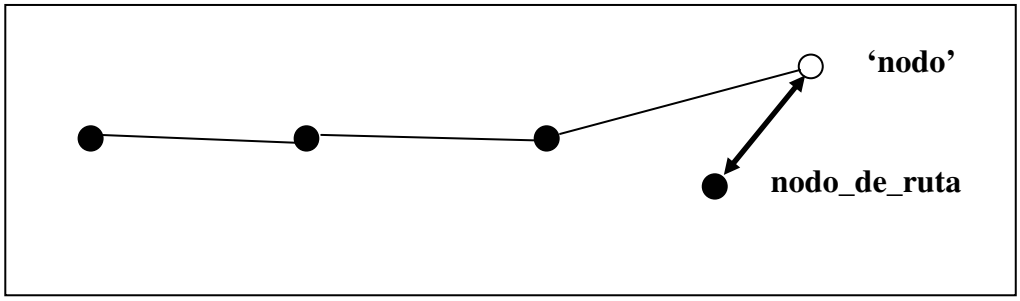
1ª posibilidad:



No hay que hacer ninguna reestructuración en la ruta. Tan solo añadir '**nodo**' al final.

2ª posibilidad:

En esta ocasión hay que rehacer un enlace.



Tras comparar ambos casos, nos quedaremos con aquel que proporcione una menor longitud de ruta (en el ejemplo expuesto, el caso primero).

Si el **nodo_de_ruta** fuera el **primer nodo** de la ruta, habría también dos posibilidades muy similares al caso anteriormente expuesto. El estudio a realizar sería, por lo tanto, análogo al anterior.

La situación en la que podemos encontrarnos es variada, y las complicaciones vendrán derivadas de que se halla alcanzado, o no, el número máximo de nodos permitidos entre los *nodos originadores* de la ruta.

Hagamos un análisis un poco más detallado:

En primer lugar, estudiamos si se ha alcanzado el número máximo de nodos permitido entre los *nodos originadores* (**n_entre**).

1) Si **no** se ha alcanzado el valor **n_entre**.

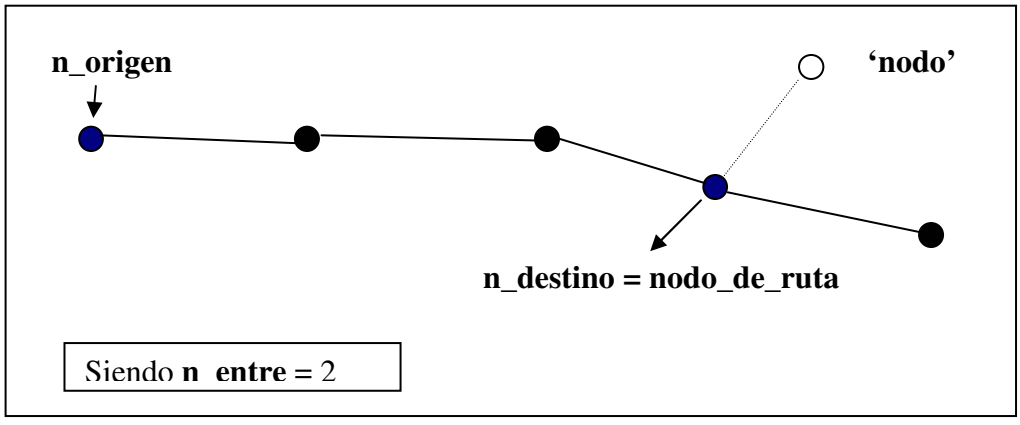
El estudio que se realiza es idéntico a lo expuesto hasta el momento.

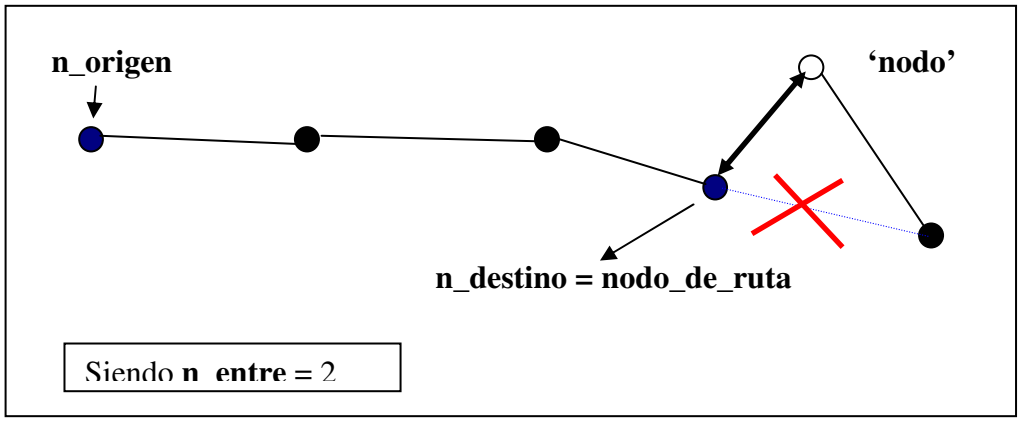
2) Sí se ha alcanzado el valor **n_entre**.

Si el número de nodos entre los *nodos originadores* es igual a **n_entre**, ya no podrán ser añadidos más nodos entre los *nodos originadores*. En tal caso, se buscará cuál es el nodo que guarda menor distancia con **'nodo'** (**nodo_de_ruta**), pero sin contar a los nodos que se encuentran entre los *nodos originadores*.

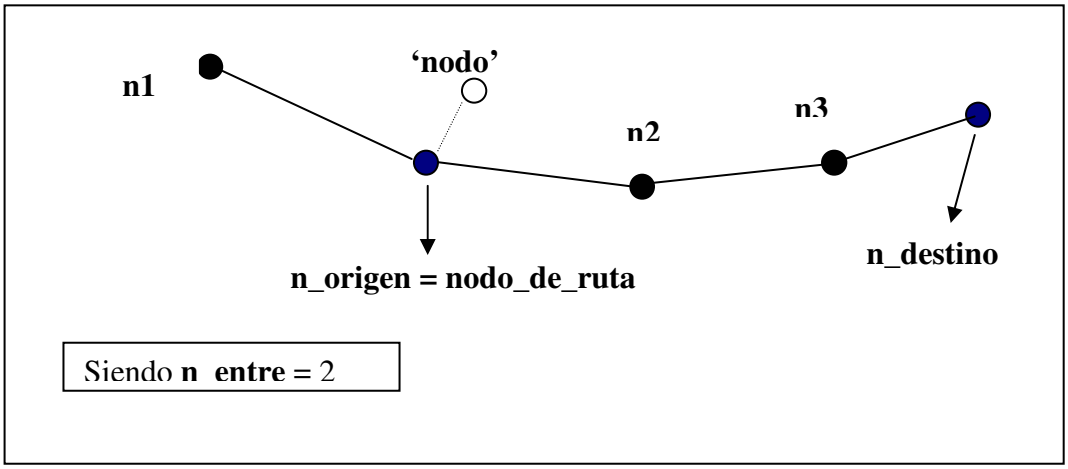
Existen varias posibilidades a tener en cuenta.

a) Que el **nodo_de_ruta** coincida con el **nodo destino** (es decir, uno de los *nodos originadores*). Entonces, la <u>única posibilidad</u> será unir **'nodo'** <u>DETRÁS</u> de **nodo_de_ruta** (siguiendo la dirección **nodo origen** → **nodo destino**), para así no sobrepasar el límite **n_entre**; y rehacer, si fuera necesario, los enlaces del **nodo destino**.



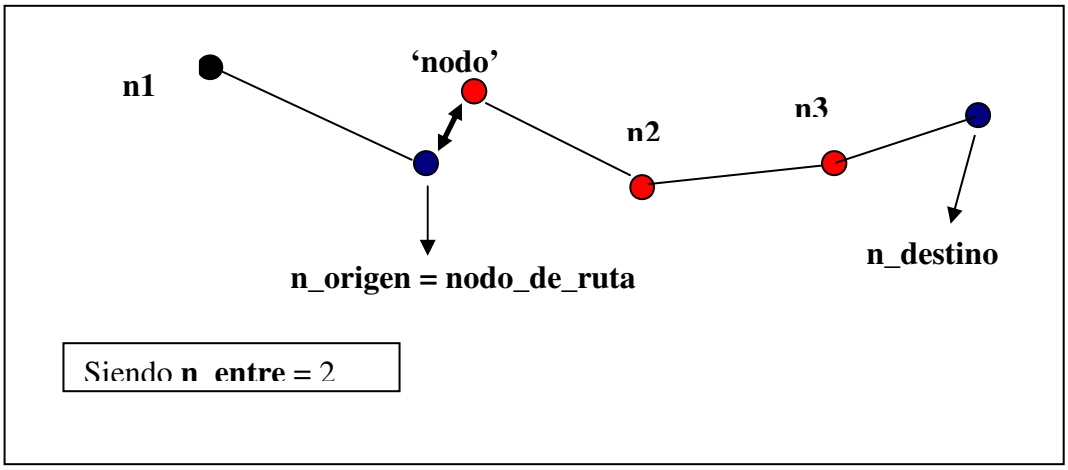


b) Una situación análoga a la expuesta en el apartado anterior es que el **nodo_de_ruta** coincida con el **nodo origen.** Entonces, la única posibilidad será insertar **'nodo'** DELANTE de **nodo_de_ruta** (siguiendo la dirección **nodo origen → nodo destino**), para así no sobrepasar el límite **n_entre**; y rehacer,si fuera necesario, los enlaces del **nodo origen**.

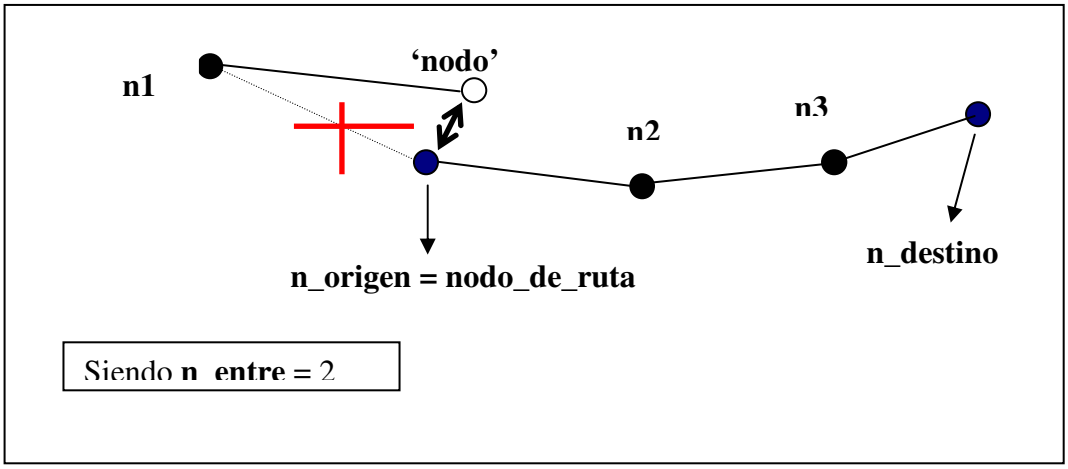


En este caso, parece que lo más lógico sería una ruta:

n1 - n_origen – 'nodo' – n2 – n3 – n_destino:



Pero de esta forma, no se estaría respetando la restricción de no permitir más de **n_entre** (en el ejemplo, **n_entre=2**) nodos entre los nodos originadores. Así pues, como se comentó al comienzo de este apartado b, **'nodo'** deberá ser insertado en la ruta precediendo a **nodo_de_ruta**:



c) **Nodo_de_ruta**, no coincide ni con el **nodo origen**, ni con el **nodo destino** de la ruta. Este caso se resuelve igual que el caso general, solo que, como ya se ha comentado, los posibles **nodo_de_ruta**, estarán limitados a ser aquellos nodos que no se encuentren entre **nodo origen** y **nodo destino**.

Una vez terminada la función, devolverá dos argumentos:

- Un vector con la nueva estructura de la ruta, tras ser añadido el nodo candidato a la misma. **ruta_nueva**.
- La longitud de la misma, tras haber sido modificada con la inclusión del nuevo nodo. **L_ruta**.

Función carga_rutas. (ver diagrama sinóptico en el anexo).

Ésta es la última función que es llamada desde la ***función general,*** una vez que ya han sido cerradas todas las rutas. Es decir, se ha dado por concluido el algoritmo de generación de rutas.

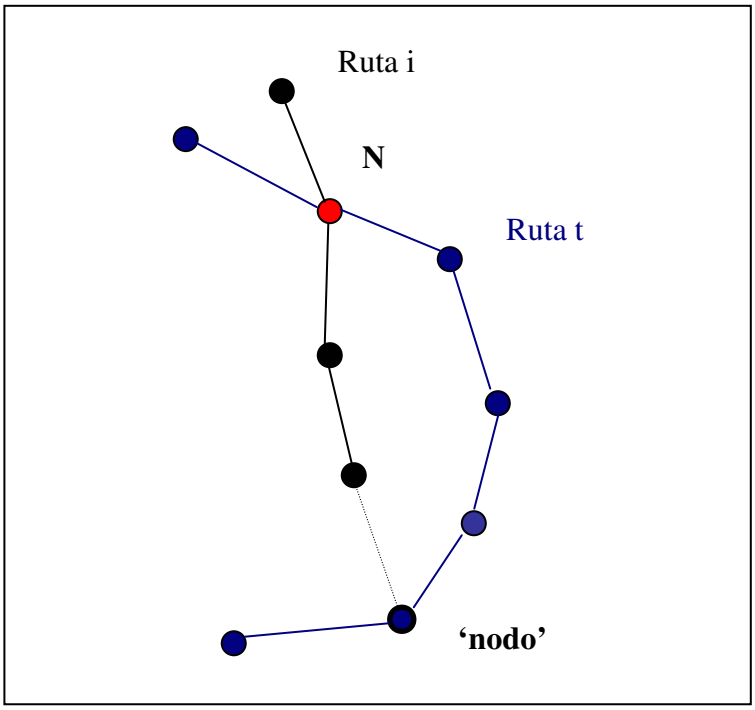
La idea de crear esta función, fue la de tener una orientación sobre la cantidad de viajes que debe soportar una determinada ruta, y por lo tanto aproximar la cantidad de vehículos necesarios. Si bien, esto requeriría otro estudio relacionado, también, con la frecuencia de paso, que queda fuera del objetivo de este proyecto.

Como fue comentado en el capítulo 1, el hecho de no trabajar con frecuencias no nos va a permitir emplear modelos de equilibrio para obtener el reparto de cargas, es decir, si una demanda origen-destino está servida por dos rutas, consideraremos que la carga se reparte a partes iguales entre ambas. Creemos que esta suposición no es nada descabellada en la aplicación que del algoritmo se hará sobre la ciudad de Sevilla, pues al tratarse de rutas de longitudes no muy largas, no existirá gran diferencia en los tiempos de viajes entre una u otra ruta. Además al no trabajar con frecuencias, no podemos contabilizar los tiempos de espera de los viajeros de una u otra línea, que sería otro dato fundamental para aplicar modelos de equilibrio. (Suponemos misma frecuencia de paso para todas las rutas).

Es necesario dejar claro que, al igual que cuando calculábamos el incremento de viajeros en la ***función cálculo***, se contabilizará el número de viajeros que han viajado en algún momento por la ruta en estudio, aunque solo sea durante el trayecto que une dos nodos consecutivos de la ruta.

Se podría pensar que es aprovechable la ***función cálculo***, para conseguir el objetivo planteado en el segundo párrafo, sin embargo no es posible. La ***función cálculo***, obtiene el incremento de viajeros en general, al unir un determinado nodo a una ruta señalada. Da igual el modo en que estos se distribuyan, es decir, no particulariza en el incremento de viajeros sufrido por una ruta concreta.

En el ejemplo siguiente queda claro lo aquí expuesto:



En este caso, la ***función cálculo***, al estudiar el incremento en el flujo que se produce al unir **'nodo'** a la **ruta i**, no sumará los viajes entre **N** y **'nodo'**, puesto que estos ya están cubiertos por la **ruta t**, y por lo tanto no supone un aumento de viajeros. Sin embargo, a la hora de calcular la carga de ambas rutas, sí habrá que contabilizar los viajes entre **N** y **'nodo'**, que deberán ser repartidos de forma equitativa entre la **ruta i** y **ruta t**. Esto es realizado por la ***función carga_rutas***.

Esta función recibe los siguientes argumentos:

- La matriz de vectores que contiene las rutas creadas: **ruta.**
- La matriz de demanda de viajes entre todos los nodos: $\mathbf{F_{ij}}$.
- El número de rutas creadas: **N**.
- Los porcentajes de personas dispuestas a realizar un transbordo para llegar a su destino: **alfa**.
  - **alfa_bus:** Transbordo entre dos rutas secundarias (autobús).
  - **alfa_metro**: Transbordo entre una principal y una secundaria.

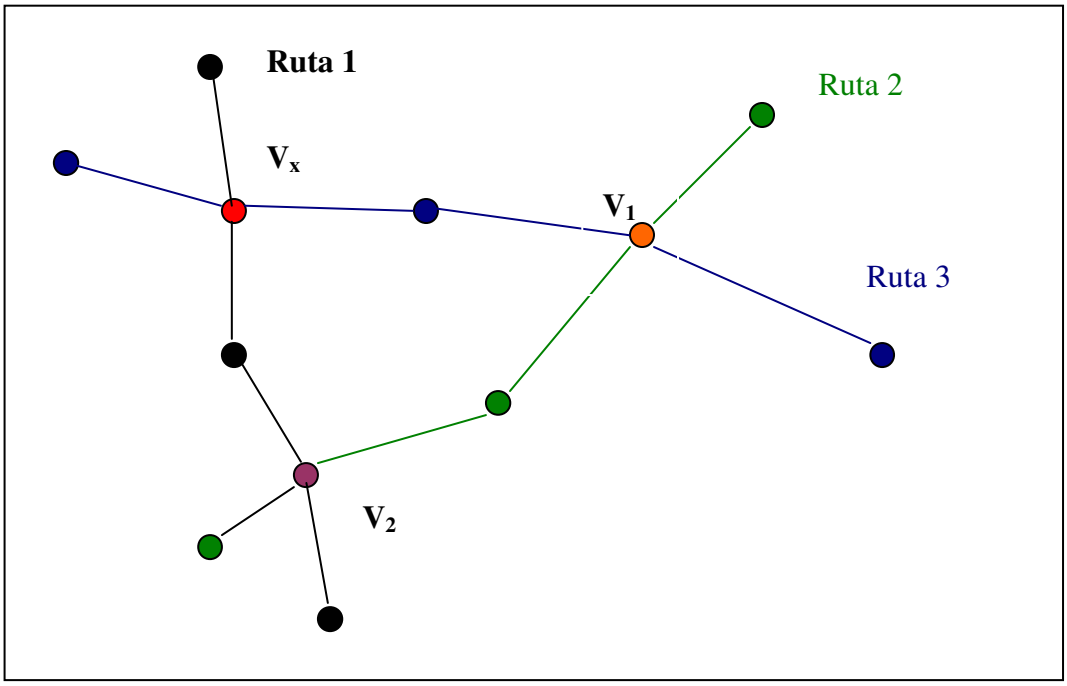
La función comienza analizando las rutas una a una, y creando una matriz cuadrada f x f (siendo f, el número de nodos) en la que cada elemento representa el número de rutas que unen de manera directa los dos nodos (fila y columna), o si están unidos por una línea principal (metro). Esto nos será de gran ayuda en la obtención del "flujo directo" entre los nodos que componen las rutas, que se halla al final de la función.

Así pues, se comienza haciendo el análisis por la primera ruta (la ruta que se está analizando en un momento determinado se denomina **ruta i**).

Lo primero, es comprobar si ésta intersecciona con alguna otra ruta (**ruta h**), por el hecho de contabilizar los transbordos. Los índices de las rutas que se cortan con la **ruta i** son guardados en el vector **int**.

Si tenemos que una **ruta h** intersecciona con la **ruta i**, en principio, se sumarán los transbordos entre ambas rutas (afectados por el factor **alfa** que corresponda). Pero seguidamente, se analizará si existe alguna ruta (**ruta t**) que ya haya sido analizada y que también corte a la **ruta i** (es decir, el número de elementos del vector **int** es superior a 1).

Para que todo quede más claro, veamos un ejemplo típico con tres rutas:

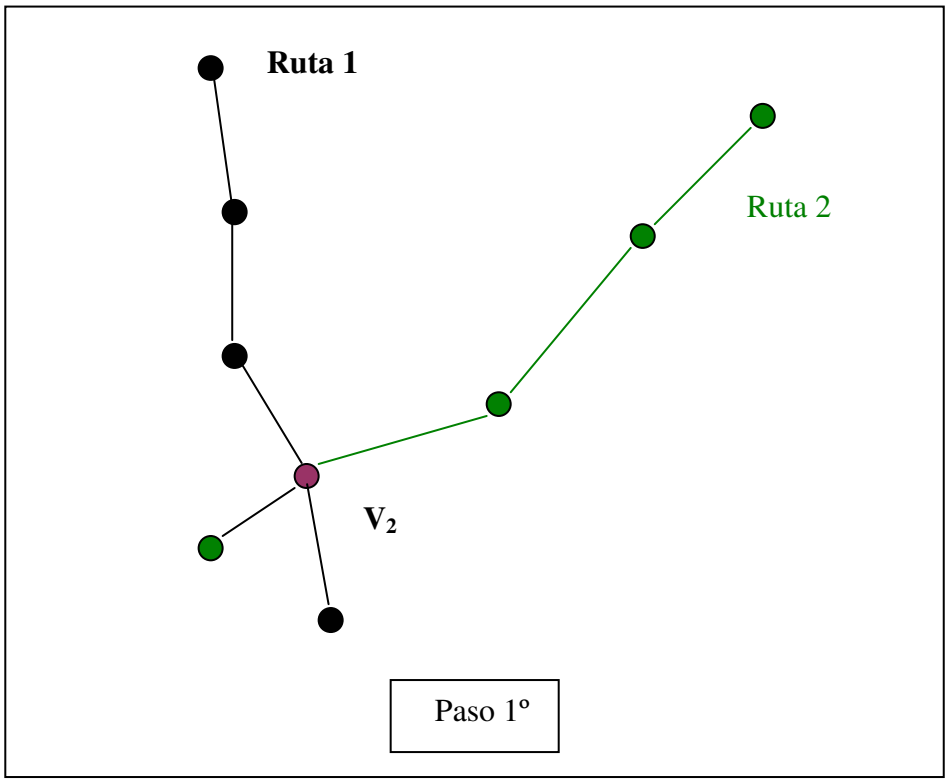


En este ejemplo, la **ruta i** de la que queremos hallar la carga es la **ruta1.**

Lo primero es calcular la carga debido a los transbordos.

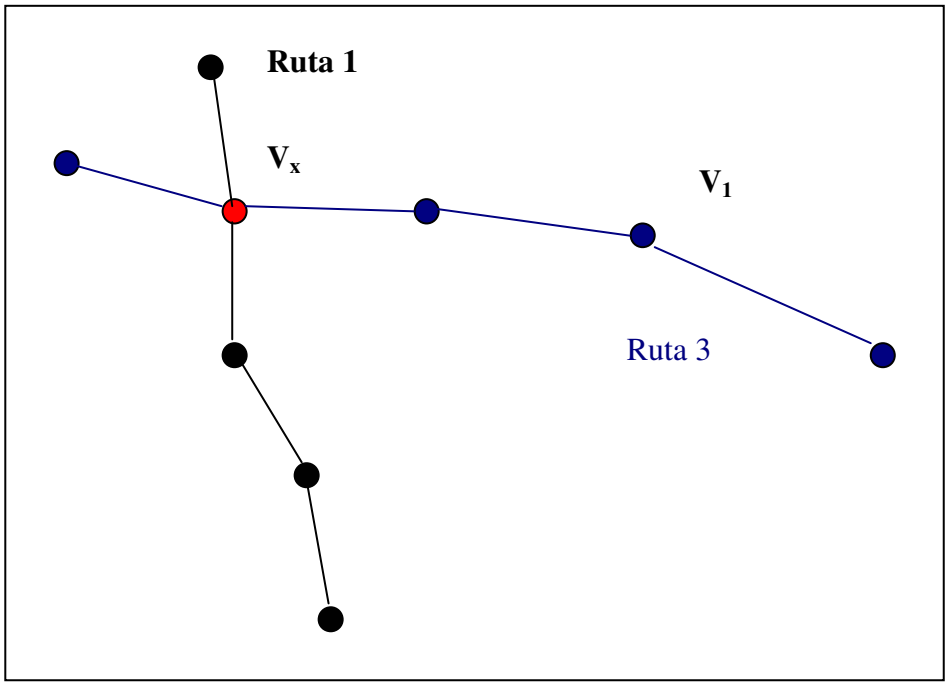
La función va tomando el resto de rutas existentes, y computando los transbordos con aquellas que interseccionan con la ruta en estudio (**ruta 1**).

La primera ruta en tomarse es la **ruta 2.** Efectivamente, ésta se corta con la **ruta 1** en el nodo $\mathbf{V_2}$, luego se contabilizarán los transbordos entre ambas rutas, guardándose su valor en **suma1**.



La **ruta 2**, será "recordada" (introduciendo su índice en el vector **int**).

A continuación, se toma la siguiente ruta existente, la **ruta 3**, se vuelve a comprobar si intersecciona con la ruta en estudio (**ruta 1**). Al ser la respuesta afirmativa (se cortan en $\mathbf{V_x}$), se añaden a la variable **suma1**, los viajes producidos con transbordo entre la **ruta 1** y la **ruta 3**.



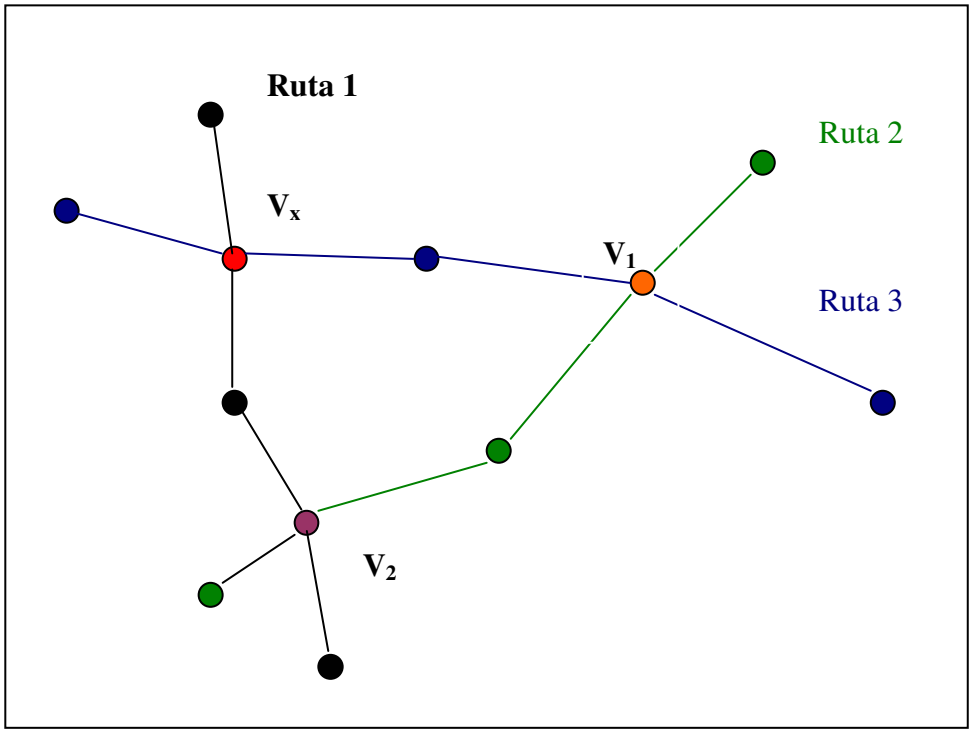
El que el vector **int** tenga guardado elementos, nos indica que ya se han estudiado los transbordos entre la ruta en estudio (**ruta 1**), y otra que se interseccionaba con ella (**ruta 2**), es por ello que hay que cuidar que no se vuelvan a contabilizar en la carga de la **ruta 1**, viajes entre nodos que ya fueron computados. Así pues, la siguiente pregunta a responder es si la **ruta 3**, se corta con las rutas guardadas en **int** (**ruta 2**). En este caso, la respuesta es afirmativa, puesto que lo hacen en $\mathbf{V_1}$ (como se puede observar en la primera figura de introducción al ejemplo en la página 60).

En adelante, habrá que comenzar a restar flujos que no debían de haber sido añadidos. Como se explica ahora, deberán ser restados:

- **suma2**, que engloba los flujos por transbordo entre la **ruta 1** (exceptuando $\mathbf{V_x}$) y el nodo $\mathbf{V_1}$. Puesto que esta carga de la **ruta 1**, ya fue contabilizada anteriormente.
- **suma3**, para no contar como carga de la **ruta 1**, los viajes con transbordo en $\mathbf{V_2}$ desde el nodo $\mathbf{V_x}$ hasta el $\mathbf{V_1}$, que fueron sumados al estudiar los transbordos entre la **ruta 1** y la **ruta 2**.
- **suma4**. Habrá que restar parte de la carga atribuida a la **ruta 1**, correspondiente a los viajes desde $\mathbf{V_2}$ hasta la **ruta 3** (excepto el

nodo V_1), transbordando en V_x, puesto que esta carga puede ser compartida por la **ruta 2** (suponemos al 50%); excepto si la **ruta 2** es una ruta principal (**metro**), en cuyo caso se supone que el 100% de los viajeros que desean ir de V_2 a la **ruta 3**, lo harán viajando por la línea de metro.

- **suma5**. Análogamente al caso anterior, también habrá que restar un 50% de los viajes realizados por transbordo en V_2 entre V_x y la **ruta 2** (excepto los nodos V_1 y V_2), puesto que esta carga será compartida entre la **ruta 1** (transbordando en V_2) y la **ruta 3** (transbordando en V_1).



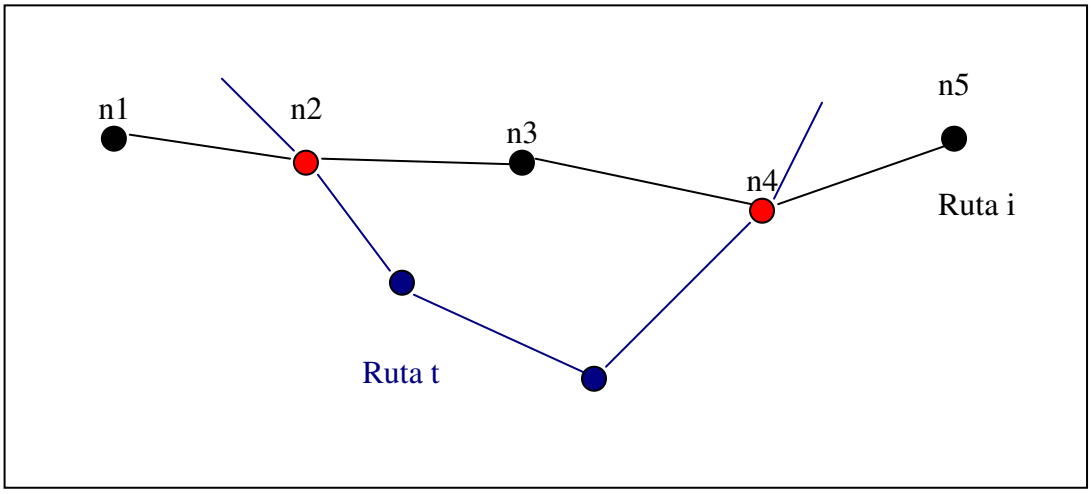
Finalmente, la carga de la **ruta 1** debido a los viajes a otras rutas mediante el transbordo, se obtiene:

trans = suma1 – suma2 – suma3 – suma4 – suma5

Es al final de la función cuando se calcula lo que se ha denominado carga directa de la ruta, es decir, la cantidad de viajes que se realizan entre los nodos de una misma ruta. Para ello se utiliza la matriz que fue obtenida al principio de la función y que indicaba el número de rutas por las que estaban unidas dos nodos cualesquiera, o si estos se encuentran unidos por una ruta principal (línea de metro).

Así pues, si la ruta en estudio es la **ruta i**, se irán tomando cada uno de los nodos que la componen, y analizando que tipo de unión y en qué cantidad (si no hay ninguna línea de metro que los una) tiene con el resto de nodos de la ruta. La suma se irá acumulando en la variable **directo**.

Veamos un par de ejemplos:



En este caso, al hallar la carga directa de la **ruta i** (el flujo de viajeros entre todos los nodos de la **ruta i**), habrá que tener en cuenta, que aproximadamente la mitad de lo viajes que se realizan entre **n2** y **n4** se hacen por la **ruta t.**

Existe una excepción, y es que la **ruta t**, fuera una línea de metro, en tal caso se considerará que toda la demanda de viajes entre **n2** y **n4** viajará utilizando la línea de metro (**ruta t**), y por lo tanto, el flujo entre **n2** y **n4** no aportará nada a la carga de la **ruta i**.

La carga final de la **ruta i** será: **carga(i) = trans + directo.**

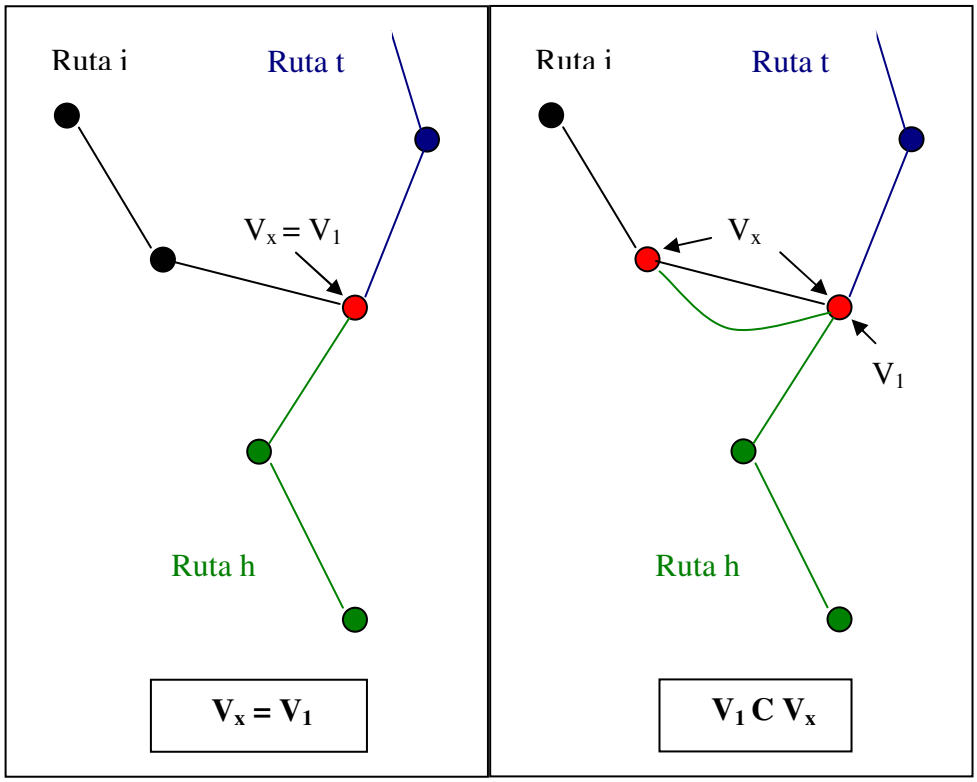
Sin embargo, aunque el ejemplo anteriormente expuesto es un ejemplo general, las configuraciones de las rutas entre si puede ser muy diversas. El

algoritmo trata de salvar todas las dificultades que puedan ir apareciendo, por ejemplo:

- Si $\mathbf{V_x = V_1}$ ó $\mathbf{V_1\ C\ V_x}$.

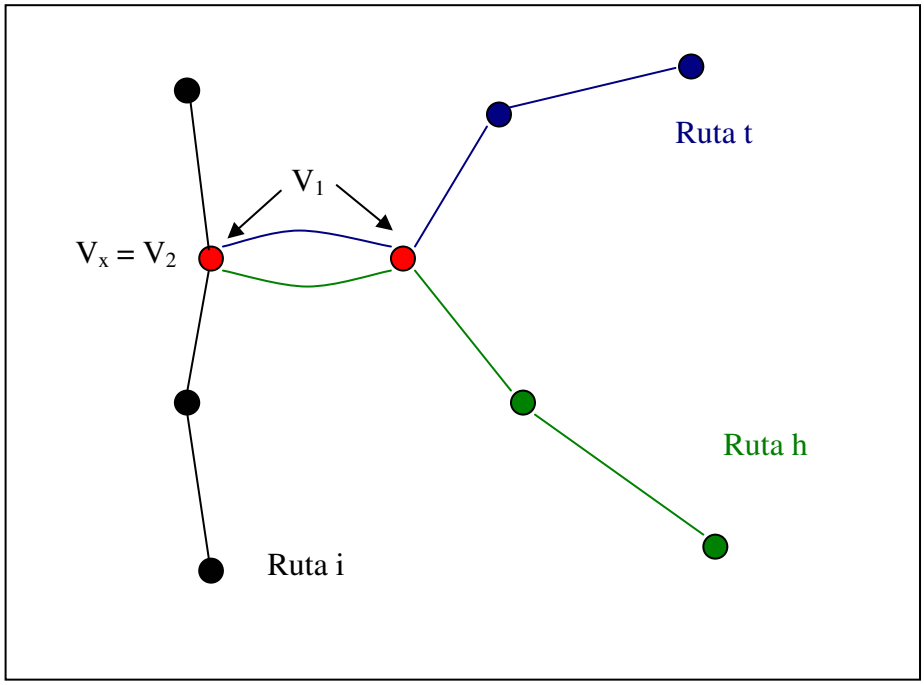
  Siendo: $\mathbf{V_x}$, la intersección de la **ruta i**, de la queremos hallar la carga, con la **ruta h**, que es la ruta con la que en estos momentos queremos hallar los transbordos.

  $\mathbf{V_1}$, la intersección entre la **ruta h** y la **ruta t** (ruta que ya fue analizada su intersección con la **ruta i**, y que también se corta con la **ruta h**).



En estos casos, no hay que restar nada a los transbordos sumados en la variable **suma1**, al hallar la carga de la **ruta i**.

- Si **V_x C V_1** :

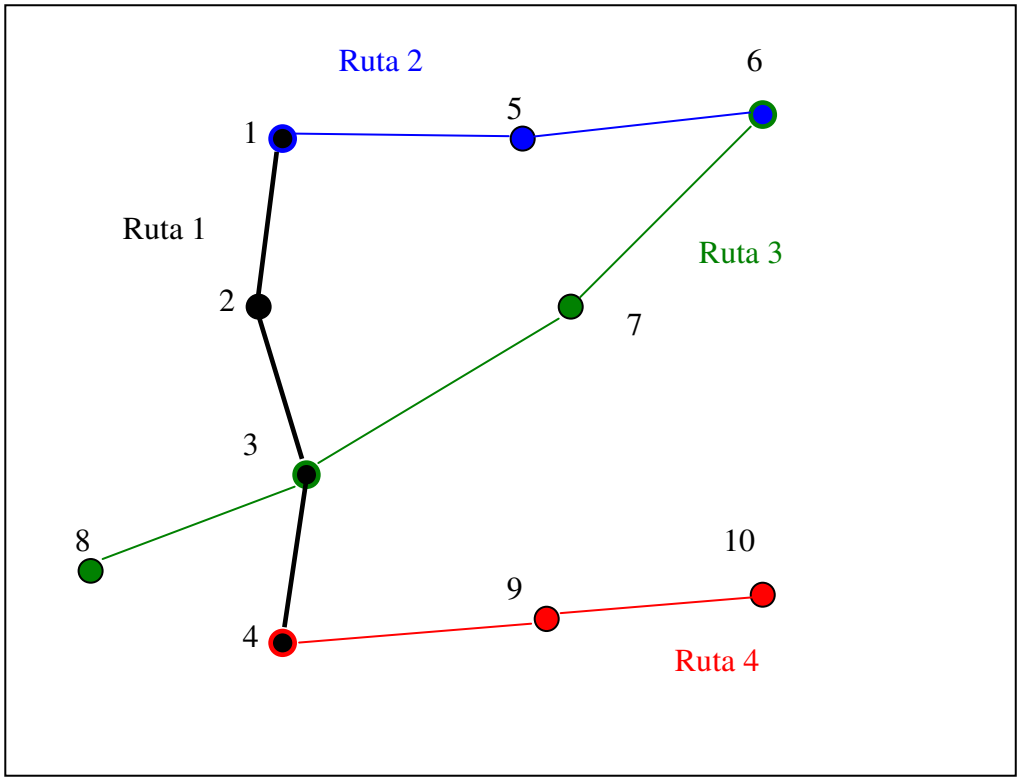


En este caso particular, solo habrá que restar a la carga de la **ruta i**, **suma3** (los viajes entre **V_x** y **V_1**, puesto que ya fueron contabilizados al estudiar los transbordos con la **ruta t**, y no deben ser sumados dos veces).

La mejor manera de comprobar que la función ha sido programada correctamente, es probando algunos ejemplos generales, pero lo suficientemente manejables como para poder comprobar manualmente que los resultados obtenidos son efectivamente los reales. Así pues, y para concluir con la explicación de esta función se mostrarán un par de casos que pueden ser encontrados en una situación real con bastante frecuencia, y se hará una comprobación manual de los resultados:

Ejemplo 1. comprobación de la función carga_rutas.

(Este es un ejemplo que corresponde al grupo en los que $\mathbf{V_x \neq V_1}$).



Ruta 1, es una ruta principal fijada de antemano (línea de metro).

Ruta 2, **Ruta 3** y **Ruta 4**, son las rutas secundarias (de autobús), halladas por el algoritmo, y en las que estamos interesados por hallar la carga que soportan.

Supondremos:

alfa_bus = 0.5. Solo es captado el 50% de la demanda de viajes entre dos nodos, si para ir de uno a otro hay que transbordar entre dos rutas secundarias (líneas de autobús).

alfa_metro = 0.8. Se capta el 80%, si la demanda se cubre transbordando entre una ruta principal, y otra secundaria.

Se dispone de la matriz simétrica de flujos entre los diferentes nodos.

$F_{i,j} = f_{i,j} + f_{j,i}$. (se han tomado valores sencillos para facilitar las operaciones.

	1	2	3	4	5	6	7	8	9	10
1	0	1	2	2	3	2	1	2	2	1
2	1	0	1	1	2	3	1	2	1	2
3	2	1	0	2	2	1	2	3	2	2
4	2	1	2	0	1	1	1	2	2	1
5	3	2	2	1	0	2	1	2	2	2
6	2	3	1	1	2	0	2	2	1	1
7	1	1	2	1	1	2	0	2	2	2
8	2	2	3	2	2	2	2	0	2	2
9	2	1	2	2	2	1	2	2	0	2
10	1	2	2	1	2	1	2	2	2	0

Carga de la Ruta 2:

$Directo_{R2} = F_{15} + F_{16} + F_{56} = 3 + 2 + 2 = \underline{7}$

$Trans_{R2} = [F_{57} + F_{58}] \cdot 0.5 + [F_{52} + F_{53} + F_{54}] \cdot 0.8 +$

$[(F_{62} + F_{64}) \cdot 0.5] \cdot 0.8 =$

$= (1+2) \cdot 0.5 + (2+2+1) \cdot 0.8 + (3+1) \cdot 0.5 \cdot 0.8 = \underline{7.1}$

CARGA(2) = 7 + 7.1 = 14.1

Si introducimos los datos en la función ***carga_rutas***, obtenemos los siguientes valores:

Suma1 = **[F_{52} + F_{53} + F_{54} + F_{62} +F_{63} +F_{64}] · 0.8 +**

+ [F_{18} + F_{13} + F_{17} + F_{58} + F_{53} + F_{57}] · 0.5 =

= [2+2+1+3+1+1] · 0.8 + [2+2+1+2+2+1] · 0.5 = 8 + 5 = 13

Suma2 = [F_{13} + F_{53}] · 0.5 = [2+2] · 0.5 = 2

Suma3 = F_{63} · 0.8 = 1 · 0.8 = 0.8

Suma4 = [F_{17} + F_{18}] · 0.5 = [1+2] · 0.5 = 1.5

Suma5 = [(F_{62} + F_{64}) · 0.5] · 0.8 = (3+1) · 0.5 · 0.8 = 1.6

Trans = suma1 – suma2 – suma3 – suma4 – suma5 =

= 13 – 2 – 0.8 – 1.5 – 1.6 = 7.1

Directo = F_{15} + F_{16} + F_{56} = 3 + 2 + 2 = 7

CARGA(2) = 7 + 7.1 = 14.1

Efectivamente, la función nos da el mismo valor que obtuvimos manualmente.

De igual modo, se puede comprobar como la función también nos proporciona correctamente el valor de la carga de las **ruta 3** y **ruta 4**, cuyos cálculos manuales mostramos a continuación, para que puedan ser comparados estos valores con el que se obtenga tras implementar la función.

Cálculo manual de la carga de la Ruta 3:

$Directo_{R3}$ = F_{83} + F_{87} + F_{86} + F_{37} + F_{36} + F_{76} = 12

$Trans_{R3}$ = [F_{85} + F_{75}] · 0.5 + [F_{81} + F_{82} + F_{84} + F_{71} + F_{72} + F_{74}] · 0.8 +

+ [(F_{62} + F_{64}) · 0.5] · 0.8 =

= (2+1) · 0.5 + (2+2+2+1+1+1) · 0.8 + (3+1) · 0.5 · 0.8 = <u>10.3</u>

carga(3) = $Directo_{R3}$ + $Trans_{R3}$ = 12 + 10.3 = 22.3

<u>Cálculo manual de la carga de la Ruta 4:</u>

Éste es más sencillo, puesto que no hay complicaciones de intersección con varias rutas.

$Directo_{R4} = F_{49} + F_{4,10} + F_{9,10}$ = 2+1+2 = 5

$Trans_{R4} = [F_{91} + F_{92} + F_{93} + F_{10,1} + F_{10,2} + F_{10,3}] \cdot 0.8$ =

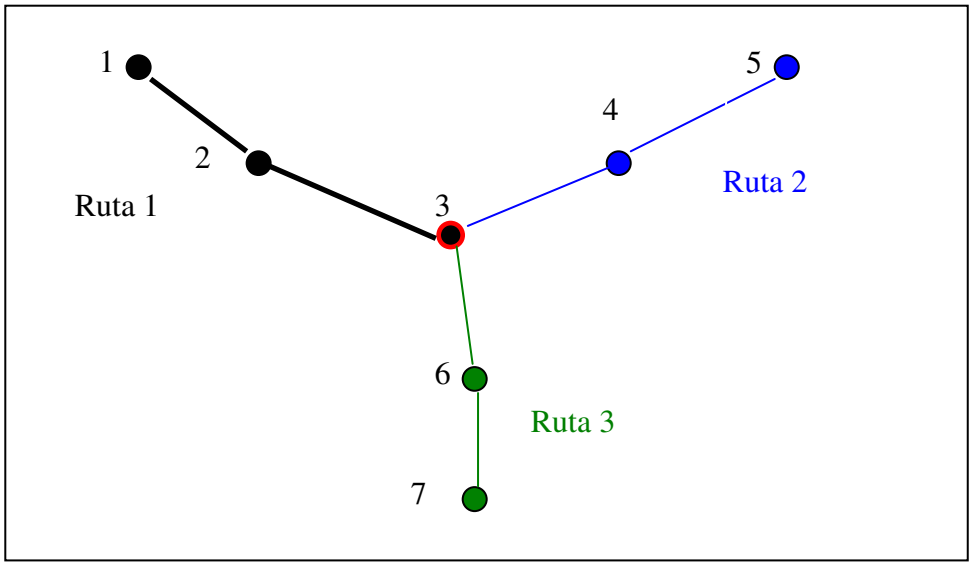
(2+1+2+1+2+2) · 0.8 = 8

Carga(4) = $Directo_{R4}$ + $Trans_{R4}$ = 5 + 8 = 13

Ejemplo 2. comprobación de la función carga_rutas.

(Corresponde al caso en el que $\mathbf{V_x=V_1}$).

Éste es más sencillo que el anterior, puesto que aquí solo hay que contabilizar transbordos, sin necesidad de restar nada al primer cálculo.



La **ruta 1**, es principal (metro),y por lo tanto fijada de antemano.

La intención es calcular la carga de las rutas creadas **ruta 2** y **ruta 3**.

Utilizando la misma matriz $\mathbf{F_{i,j}}$ del ejemplo anterior:

Carga de la Ruta 2:

$\mathbf{Directo_{R2} = F_{54} + F_{53} + F_{43} = 1 + 2 + 2 = \underline{5}}$

$\mathbf{Trans_{R2} = [\, F_{51} + F_{52} + F_{41} + F_{42}\,] \cdot 0.8 +}$

$\mathbf{[\, F_{56} + F_{57} + F_{46} + F_{47}\,] \cdot 0.5 =}$

$\mathbf{(\, 3+2+2+1) \cdot 0.8 + (\, 2+1+1+1) \cdot 0.5 = \underline{8.9}}$

Tal y como se comentó, este es el valor que se obtiene con **suma1**, y al que no hay necesidad de restar ningún valor.

$\mathbf{Carga(2) = Directo_{R2} + Trans_{R2} = Directo_{R2} + suma1 = 5 + 8.9 = 13.9}$

De igual manera se puede obtener el valor de la carga de **ruta 3**.

2.6 TEST DEL ALGORITMO SOBRE DIFERENTES CONFIGURACIONES.

Una vez presentado con detalle el algoritmo, y su programación, el siguiente paso debía ser el de testar el funcionamiento del mismo sobre varias configuraciones geométricas, con la intención de comprobar si los resultados obtenidos se encontraban dentro de unos patrones de comportamiento lógicos, solucionar posibles problemas de diseño o de programación, y obtener conclusiones.

Como la intención final de este proyecto es la de comprobar resultados sobre las líneas de autobús en la ciudad de Sevilla, se optó por aplicar el programa sobre algunas configuraciones típicas de ciudades, añadiendo cada vez más complejidad en las mismas.

El proceso seguido fue el siguiente: una vez diseñada la configuración sobre la que íbamos a trabajar (mallado, mallado-triangular, circular, circular-triangular), fueron seleccionadas tres zonas dentro de las mismas, constituidas por una serie de nodos:

- **Zona Centro**. Representados por nodos de color celeste.
- **Zona Media.** Representada por nodos de color verde.
- **Zona Periférica.** El resto de nodos de color negro.

A continuación, fue elaborado un pequeño programa en **MATLAB 6.1.** con la intención de generar de una manera aleatoria, pero siguiendo unos patrones establecidos con anterioridad, una matriz de flujos de viajeros. Así, para cada configuración, fue probado el comportamiento del algoritmo en dos situaciones:

- **Flujo radial.** Es decir, un flujo de viajeros con tendencia hacia la zona centro, especialmente, y la zona media, en menor medida, desde la periferia. Los viajes entre nodos periféricos serán muy inferiores a los anteriores. (Esta matriz no tiene por qué ser simétrica).
- **Flujo transversal.** Que representa una demanda de viajes mayor entre los nodos de la zona periférica.

Con estos dos datos: la configuración geométrica de la ciudad, y la matriz de demanda de viajes, ya solo era necesario ejecutar la aplicación programada y comenzar a dar valores a los diferentes parámetros:

- **Numero de rutas a crear: N.**
- **Longitud máxima de ruta: *L*.**
- **Número máximo de paradas permitidas entre los nodos originadores de la ruta: *n_entre*.**
- **Tanto por ciento de demanda cubierta, si los viajeros deben realizar un transbordo entre dos líneas de autobús: *α_bus*.**
- **Tanto por ciento de demanda cubierta, si los viajeros deben realizar un transbordo entre una línea de autobús y otra de metro: *α_metro*.**

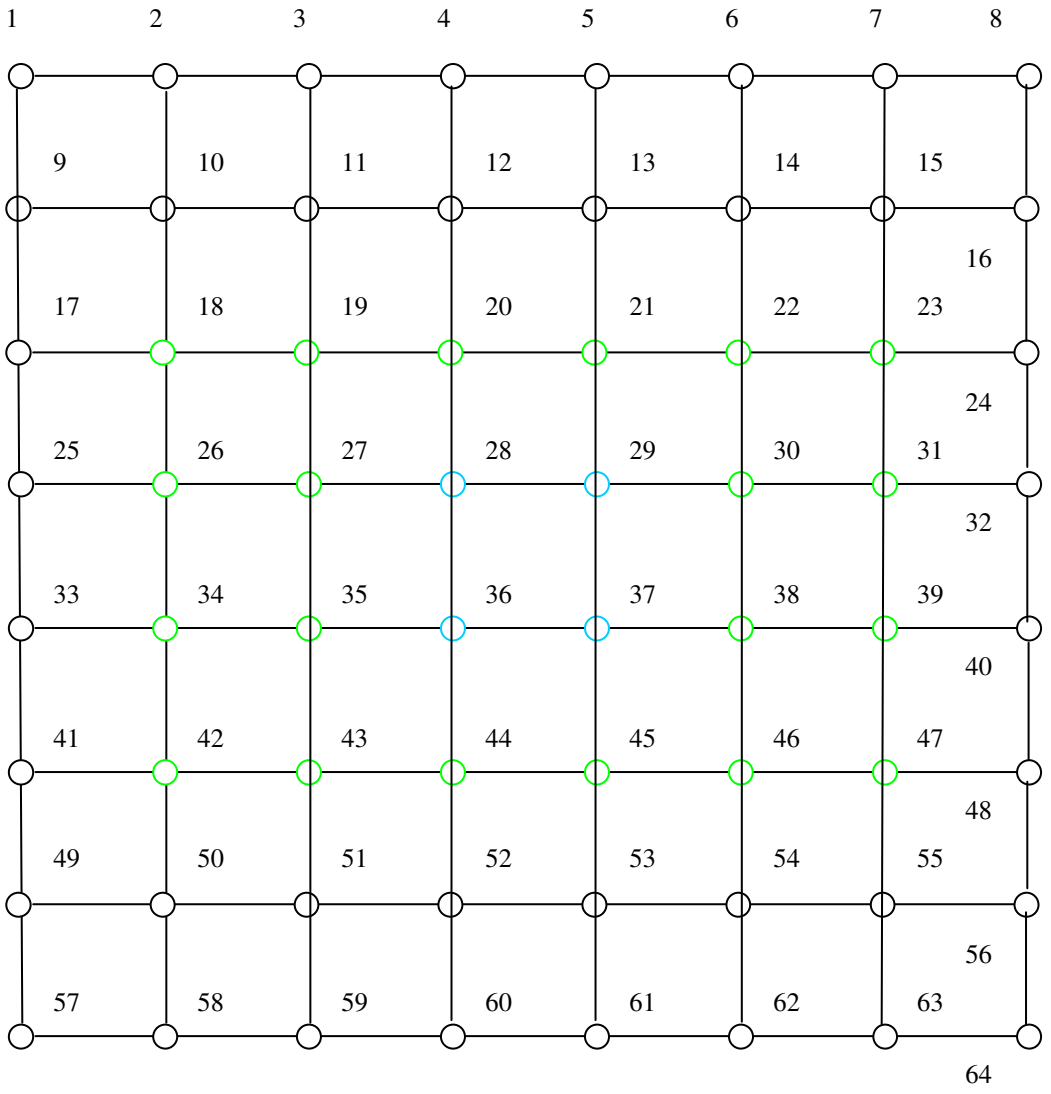
Además, el estudio se realiza en dos partes:

- **Sin la existencia de ninguna línea de metro previa a la creación de las rutas de autobuses,** en cuyo caso se estudiará el modo progresivo en el que son creadas las líneas de autobús.
- **Modificaciones que provocaría en la red de autobuses la creación de una línea de metro.**

Según estemos en un caso u otro, habrá que introducir el valor del vector que define la línea de metro: **ruta_metro**.

2.6.1. CONFIGURACIÓN MALLADA de 64 NODOS.

Consideramos todos los nodos equidistantes con sus vecinos, y a una distancia de 1 unidad, sin importarnos, para el ejemplo, a qué longitud representa dicha unidad.



2.6.1.1. MATRIZ DE DEMANDA DE VIAJES GRAVITATORIA (RADIAL).

α_bus = 0.5

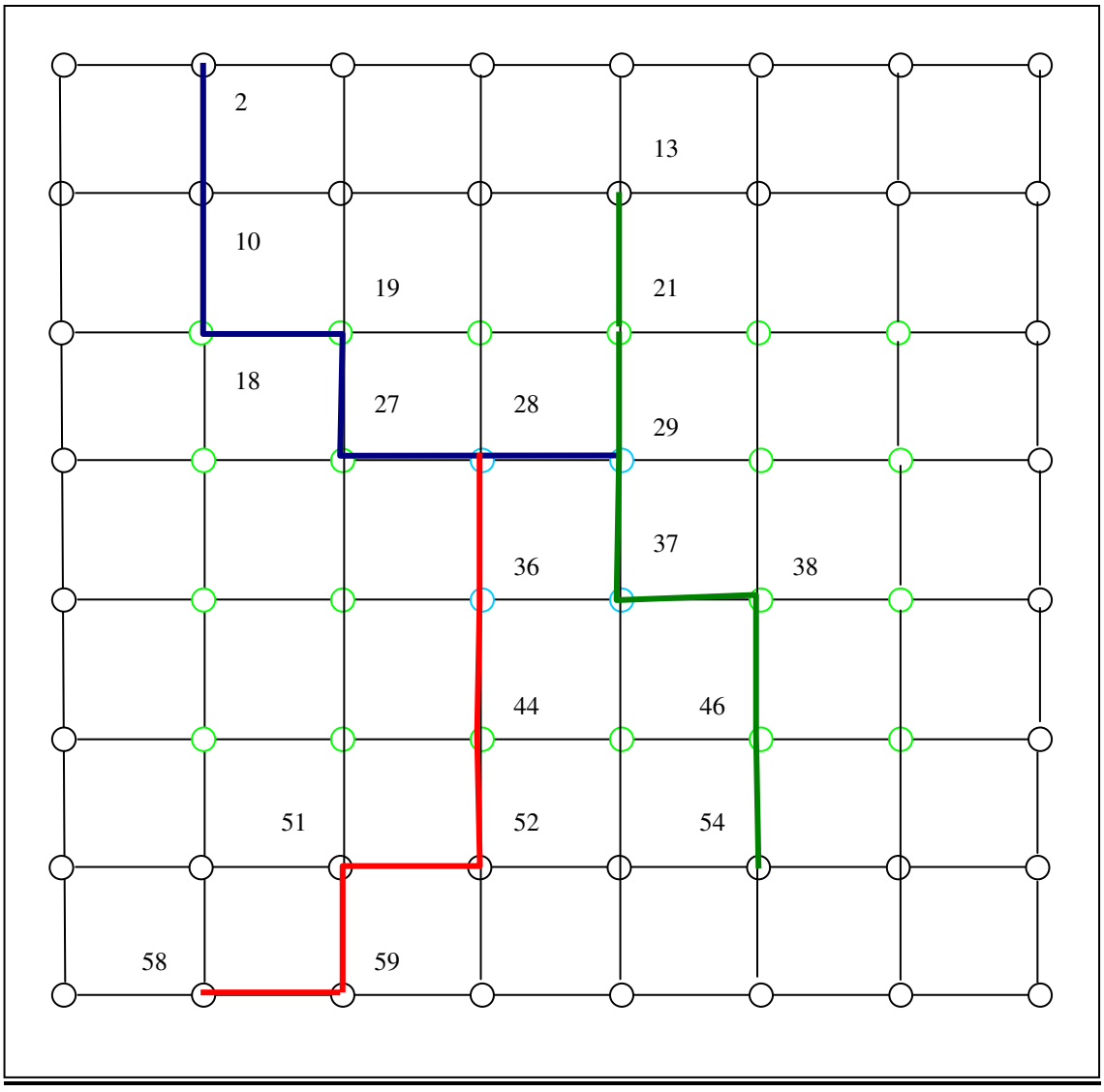
α_metro = 0.8

n_entre = 5

L = 6

2.6.1.1.1 SIN LÍNEA DE METRO.

2.6.1.1.1.1. CASO 1: N = 3 (3 rutas de autobús).

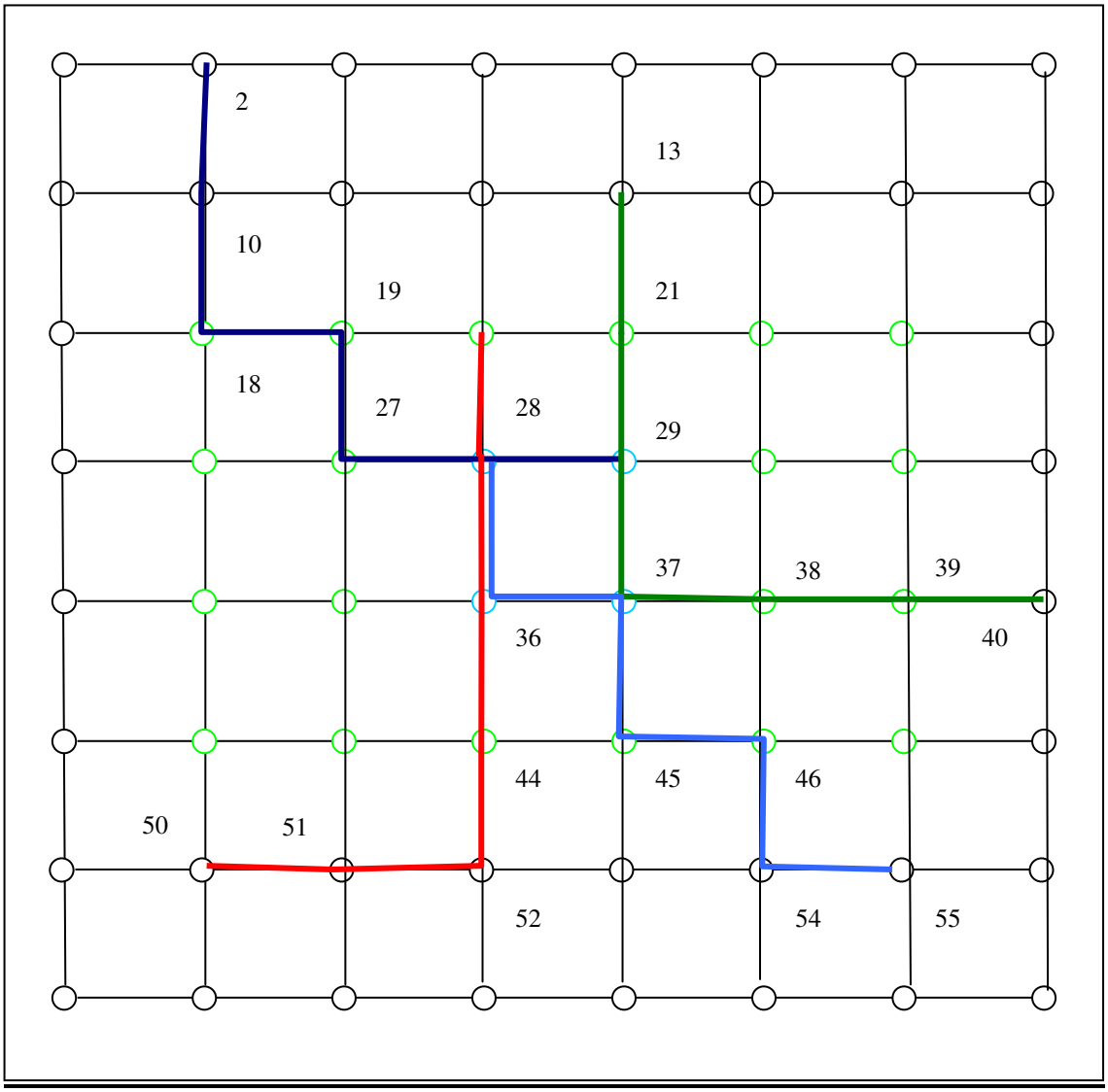


Las rutas son iniciadas entre los tres pares de nodos que proporcionan una mayor demanda de viajes, y que cumplen con todos los condicionantes que han sido explicados con detalle en este capítulo. Estos son:

(2, 29)	**(51, 28)**	**(13, 37)**

2.6.1.1.1.2. CASO 2: N = 4.

La cuarta ruta es iniciada entre los nodos **(28, 55)**.



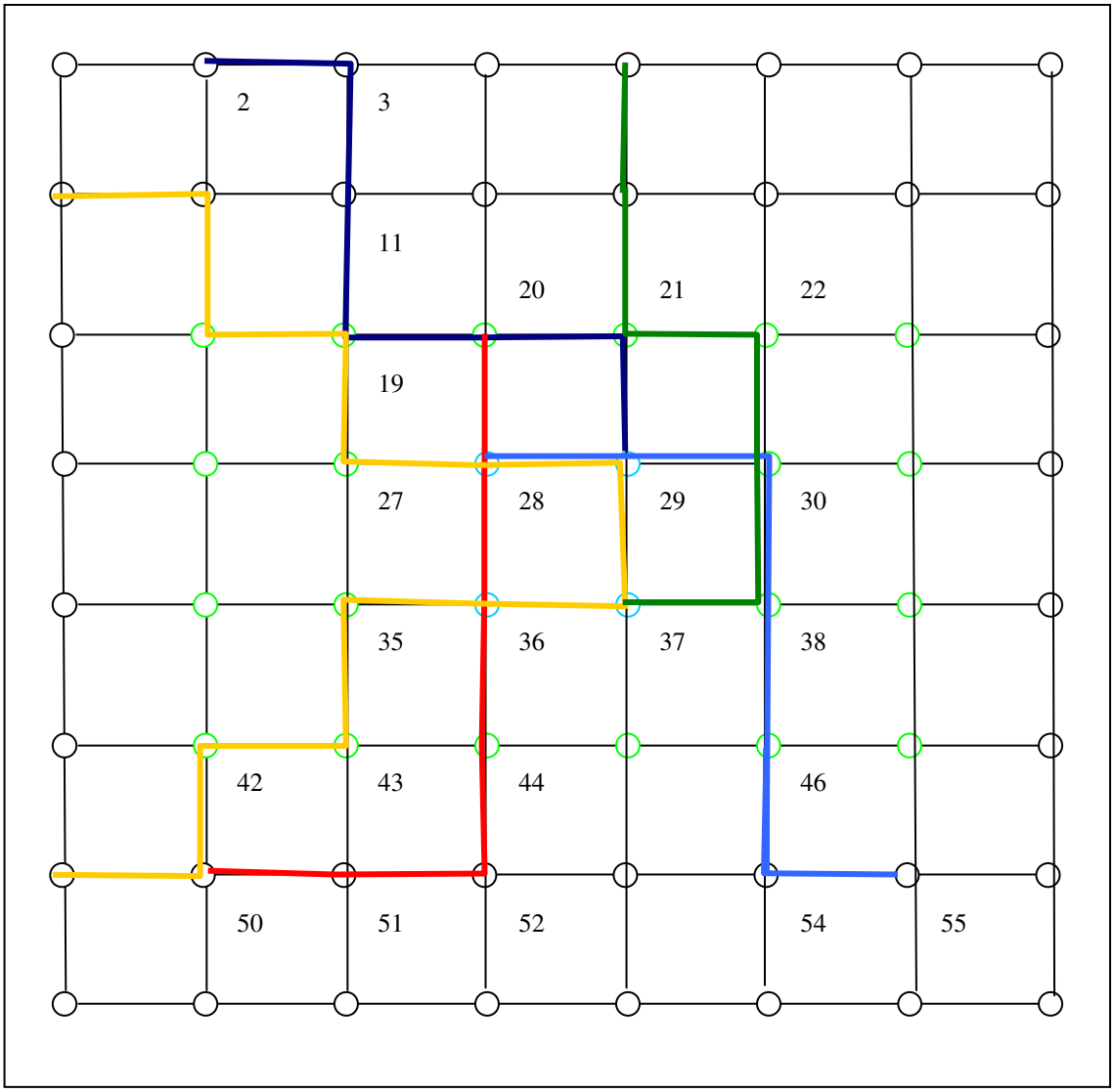
NOTA: Observar como el hecho de crear una ruta más, no es lo mismo que añadir una nueva línea a las ya existentes, de hecho éstas pueden variar, puesto que todas son generadas a la vez (como le ocurre a las rutas roja y verde).

2.6.1.1.2 CON LÍNEA DE METRO.

ruta_metro = [9, 10, 18, 19, 27, 28, 29, 37, 36, 35, 43, 42, 50, 49]

2.6.1.1.2.1 CASO 1: N = 4.

20



NOTA: Observar como el añadir una ruta de metro, ha supuesto la modificación de algunas de las rutas de autobús.

2.6.1.2. MATRIZ DE DEMANDA DE VIAJES TRANSVERSAL.

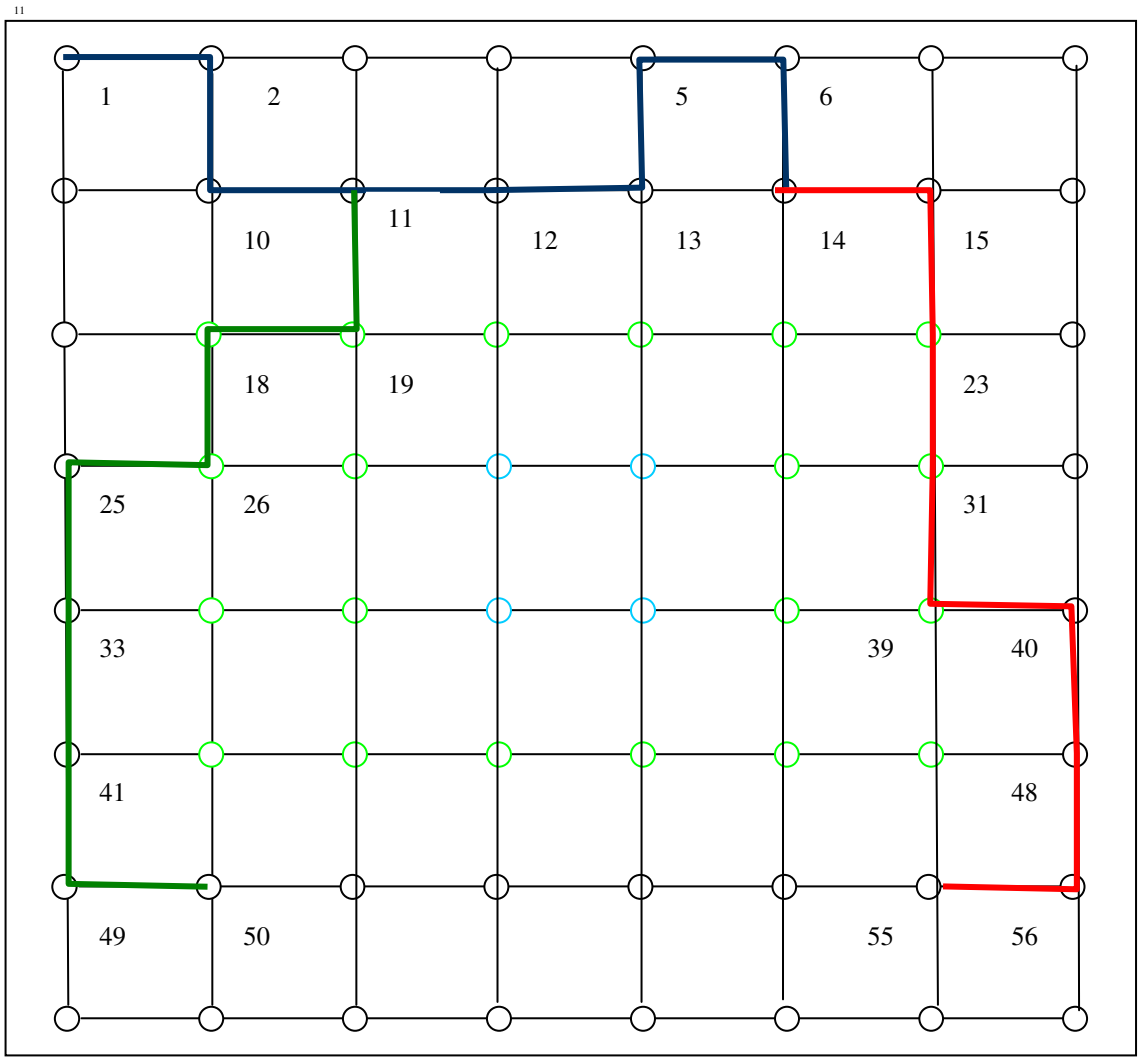
Variando tan solo la longitud máxima de ruta (**L = 8**) con respecto a los ejemplos anteriores.

2.6.1.2.1. SIN LÍNEA DE METRO.

2.6.1.2.1.1. CASO 1: N = 3

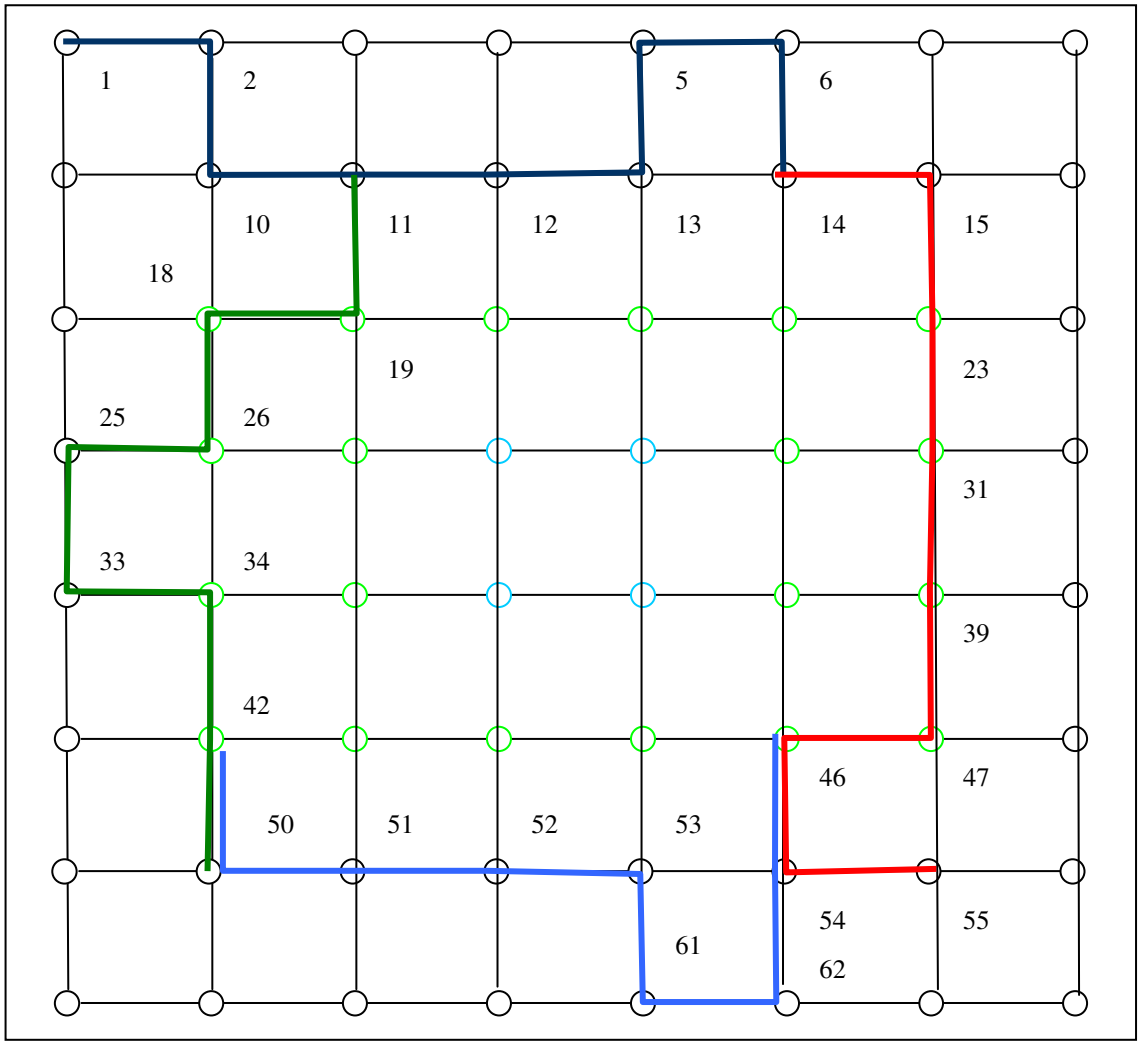
Los tres pares entre los que se comienza a generar las rutas son:

(2, 5)	**(19,50)**	**(14, 39)**



2.6.1.2.1.2. CASO 2: N = 4

La cuarta ruta es iniciada entre los nodos **(42, 61)**.

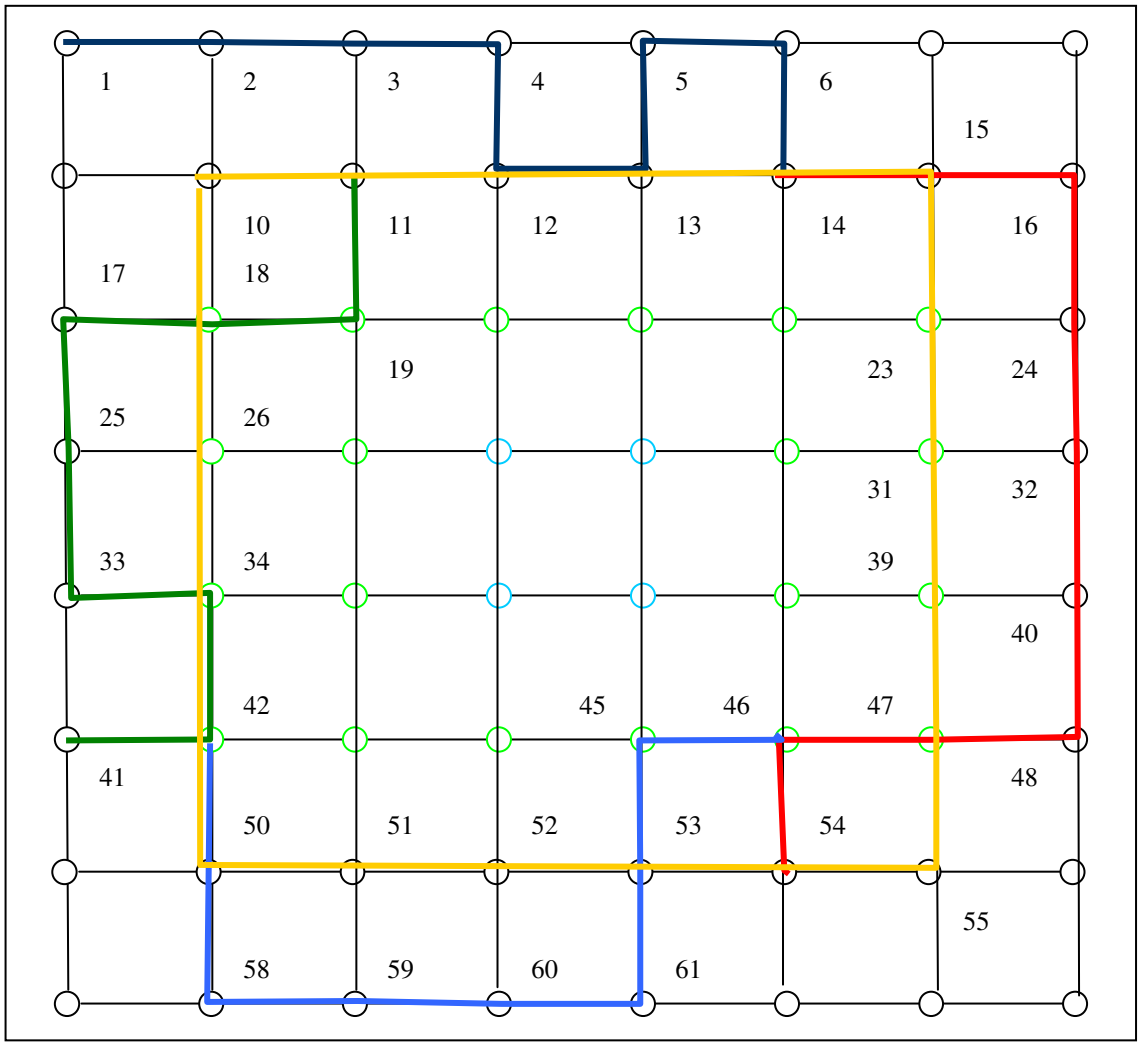


2.6.1.2.2. CON LÍNEA DE METRO.

ruta_metro = [10 11 12 13 14 15 23 31 39 47 55 54 53 52 51 50 42 34 26 18 10]

2.6.1.2.2.1 CASO 1: N = 4

Como se comentará posteriormente en las conclusiones finales, es fácil apreciar como la creación de una línea de metro, va a suponer una amplia reestructuración en las líneas, sobretodo en aquellas coincidentes en su trayectoria con la de metro.

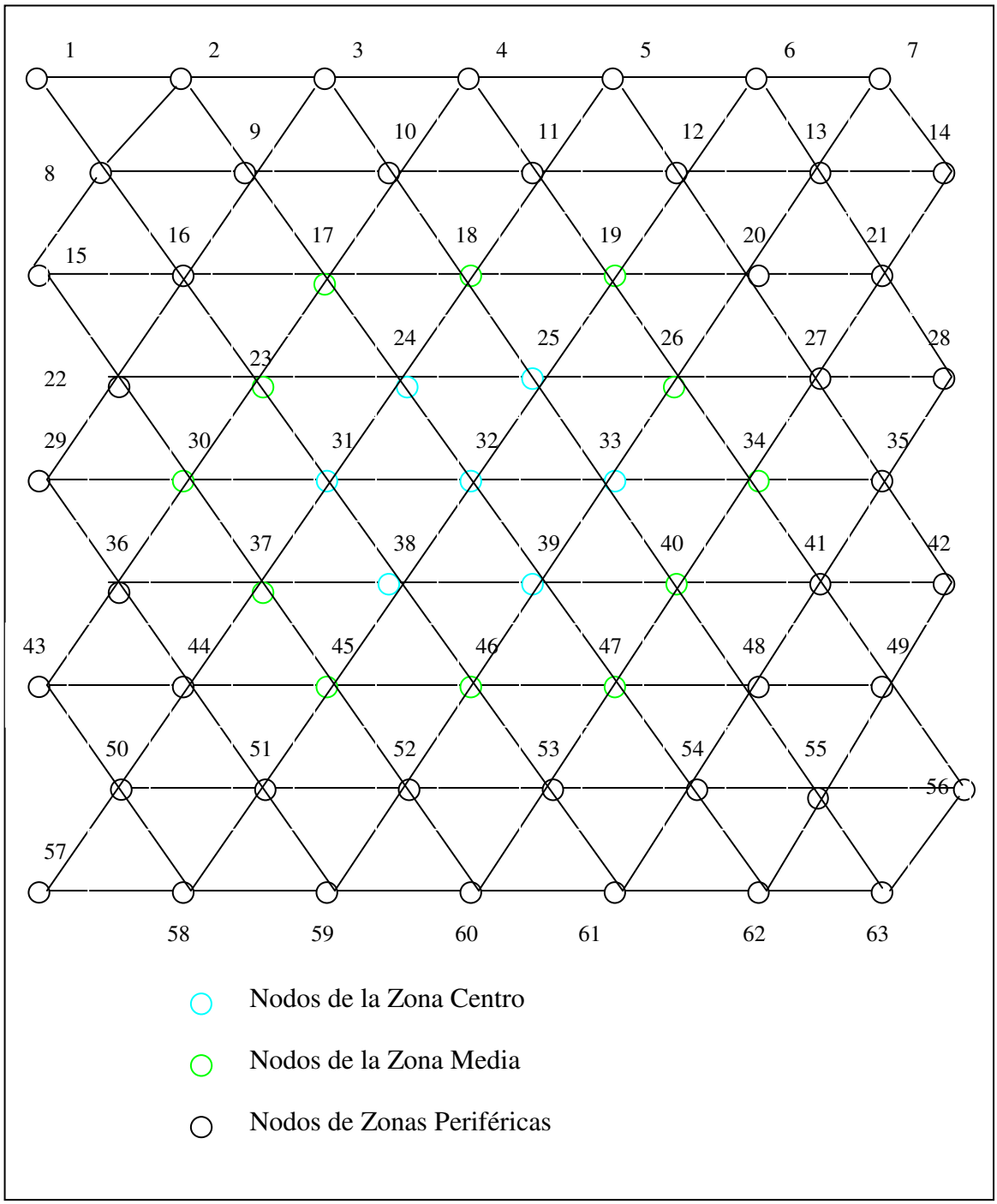


NOTA:

Como era de esperar, tanto en el ejemplo de matriz de demanda de viajes radial, como en la transversal, las rutas que se crean responden a la lógica, teniendo éstas tendencia gravitacional o entre nodos periféricos según el caso.

2.6.2. CONFIGURACIÓN MALLADA-TRIANGULAR de 63 NODOS.

Consideramos todos los nodos equidistantes con sus vecinos, y a una distancia de 1 unidad, sin importarnos, para el ejemplo, a qué longitud representa dicha unidad.



2.6.2.1. MATRIZ DE DEMANDA DE VIAJES GRAVITATORIA (RADIAL).

α_bus = 0.4

α_metro = 0.8

n_entre = 3

L = 6

2.6.2.1.1. SIN LÍNEA DE METRO.

2.6.2.1.1.1. CASO 1: N = 3 (3 rutas de autobús).

Los pares de nodos originadores de las tres primeras rutas son:

(57, 39) (31,4) (32, 15)

Las rutas resultantes son:

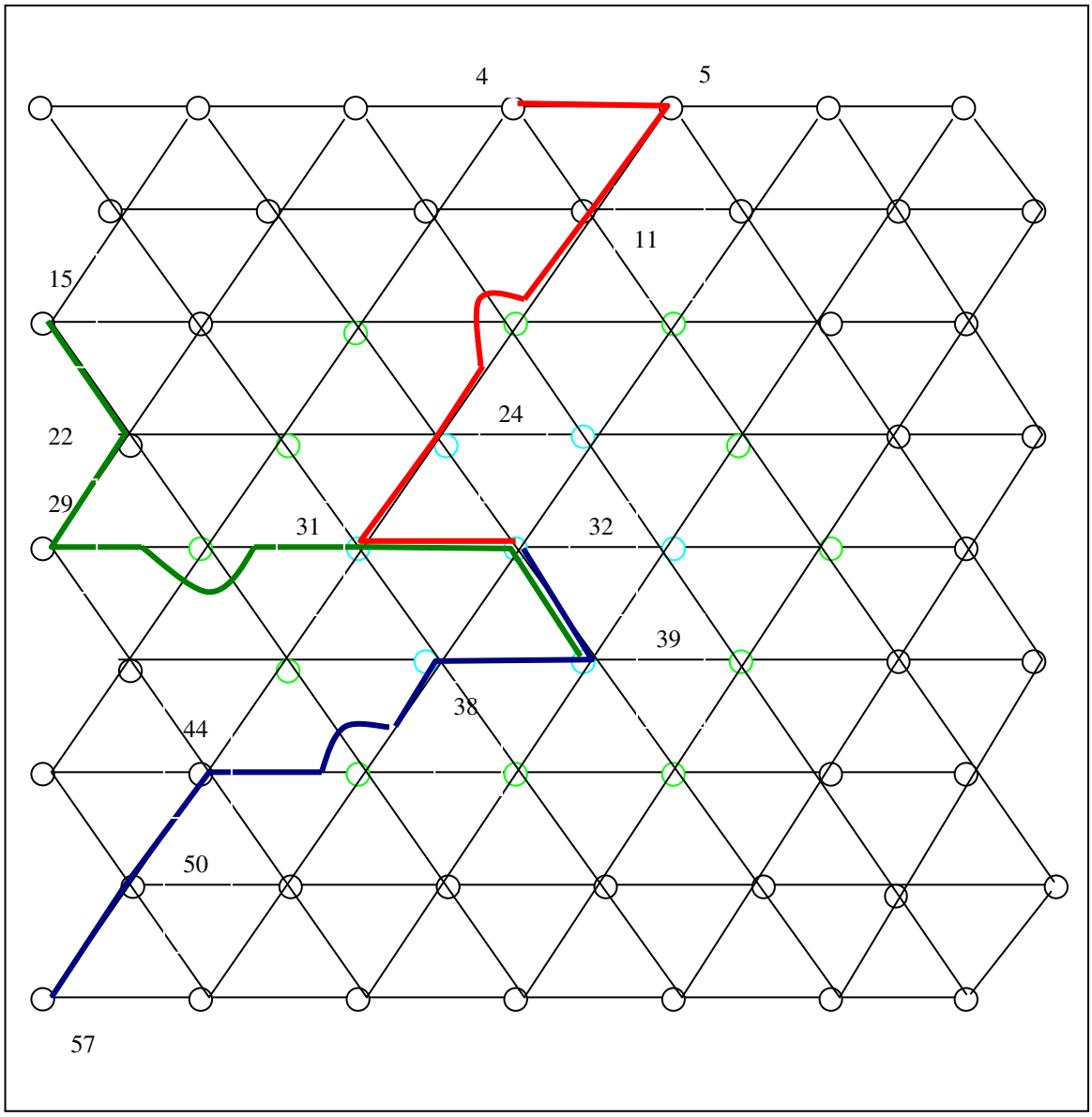
Ruta 1 = (57 50 44 38 39 32) carga = 2168

Ruta 2 = (32 31 24 11 5 4) carga = 2285

Ruta 3 = (39 32 31 29 22 15) carga = 1839

Alcanzado todas ellas la longitud máxima (**L = 6**).

Si observamos la carga diaria de pasajeros, veremos como la de la ruta 2 supera a la de la ruta 1, a pesar de que los nodos que originaron ésta, tenían una demanda menor de viajes.



<u>NOTA:</u> Los nodos intermedios de las rutas que son circunvalados, indican que la línea debe pasar por allí, pero que, en principio, la restricción *n_entre = 3* , les impide pertenecer a la ruta, lo que significa que no son paradas de la ruta.

2.6.2.1.1.2. CASO 2: N = 4.

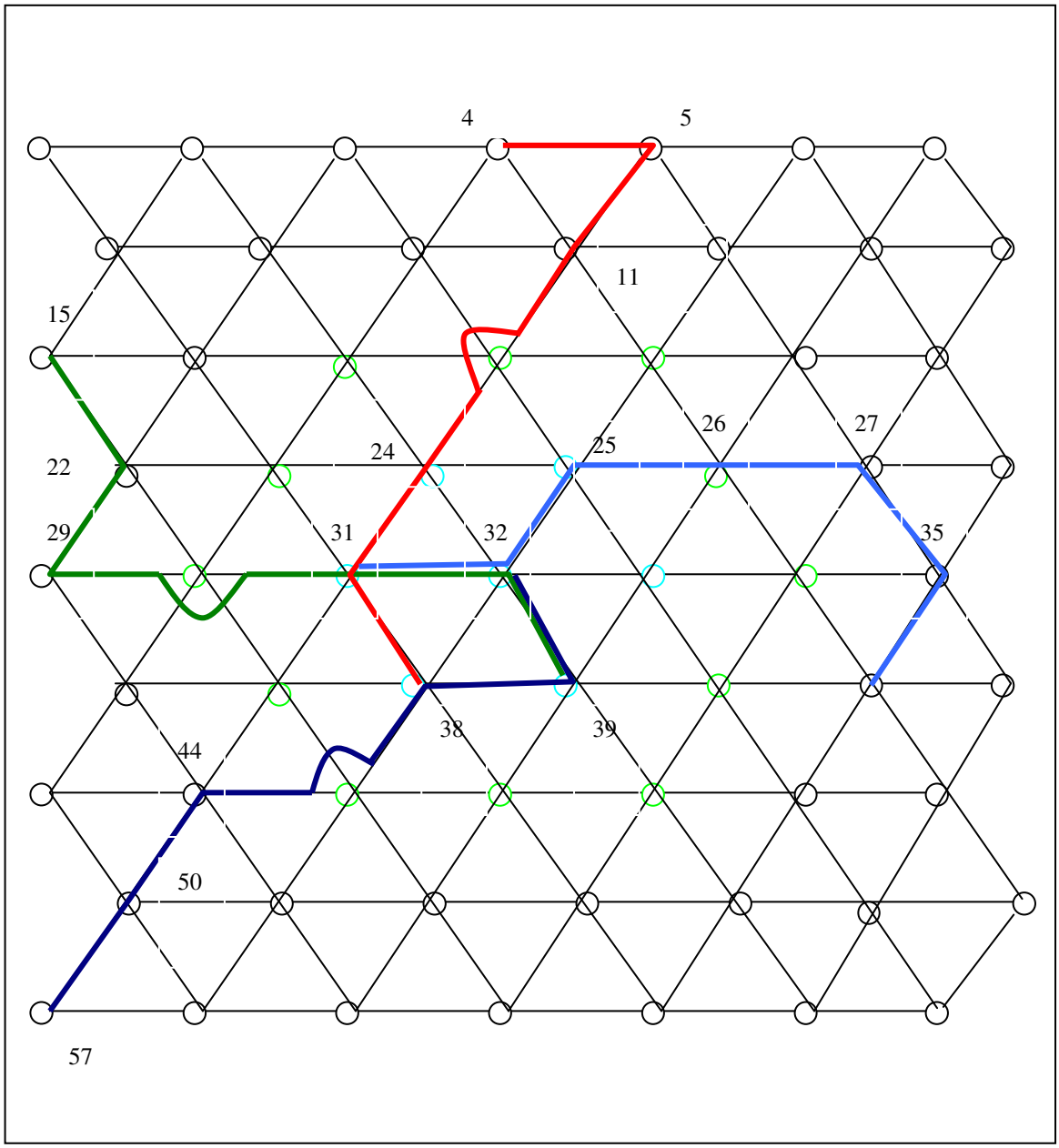
La cuarta ruta es iniciada entre los nodos **(35, 25)**.

Como se observa en la figura, las tres primeras rutas apenas se ven modificadas (tan solo la segunda, ruta roja, varía algo). Esto también puede venir motivado por las restricciones impuestas (***n_entre = 3*** y ***L = 6***).

La creación de una nueva ruta, en este caso no ha supuesto una amplia reestructuración de las anteriores, aunque si se verá claramente incrementada la carga de todas ellas, al tener que añadir los viajeros que están dispuestos a transbordar en la cuarta ruta (línea celeste), hacia las paradas de la misma.

Ruta 1 = (57 50 44 38 39 32)	**carga = 2656**
Ruta 2 = (38 31 24 11 5 4)	**carga = 2847**
Ruta 3 = (39 32 31 29 22 15)	**carga = 2514**
Ruta 4 = (41 35 27 26 25 32 31)	**carga = 2534**

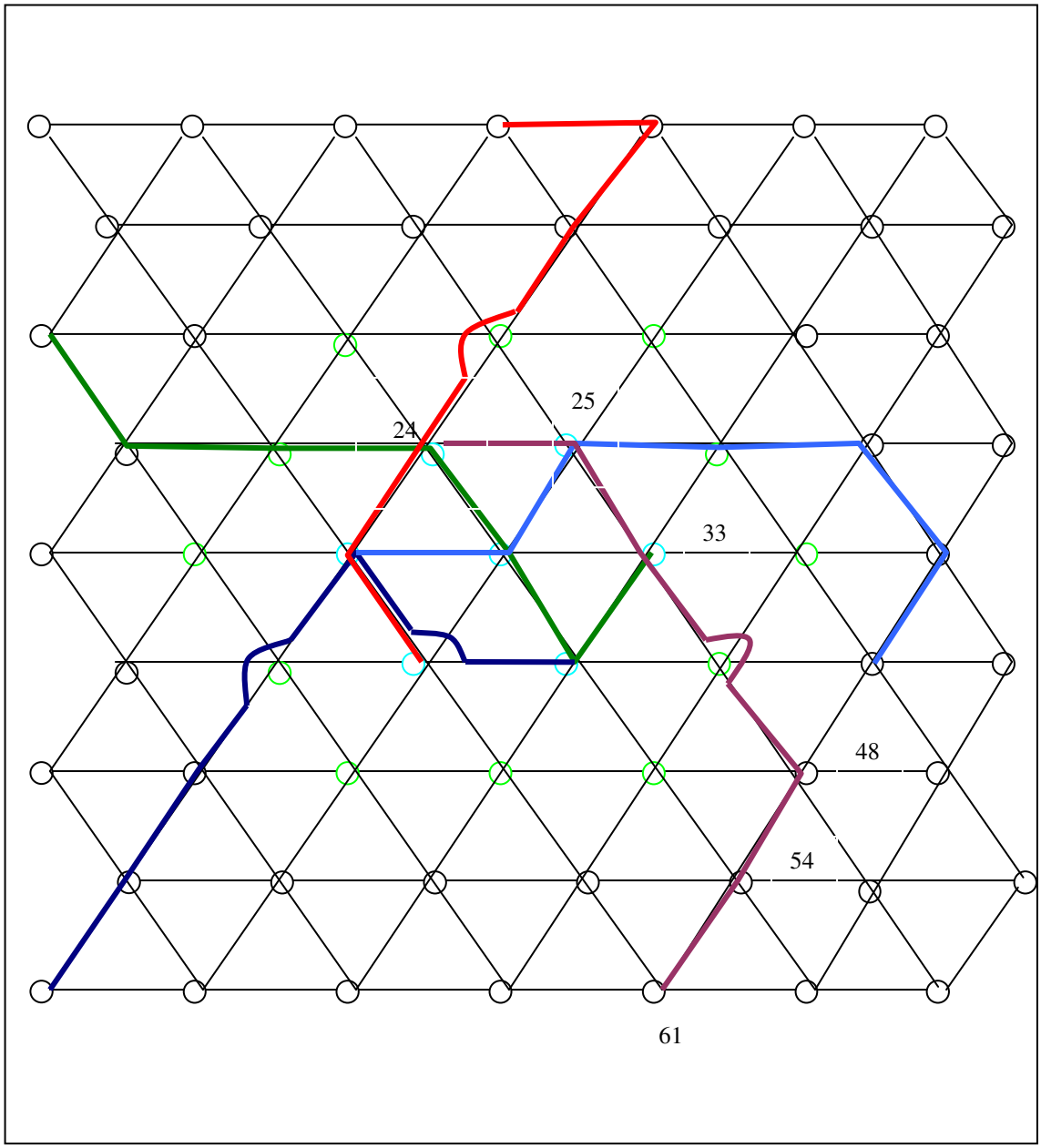
Todas las rutas mantienen la estructura radial, como respuesta a la demanda gravitacional de viajes.


4
5
11
15
22
24
25
26
27
29
31
32
35
38
39
44
50
57

2.6.2.1.1.3. CASO 3: N = 5.

La quinta ruta (morada) es iniciada entre los nodos **(61, 25)**.

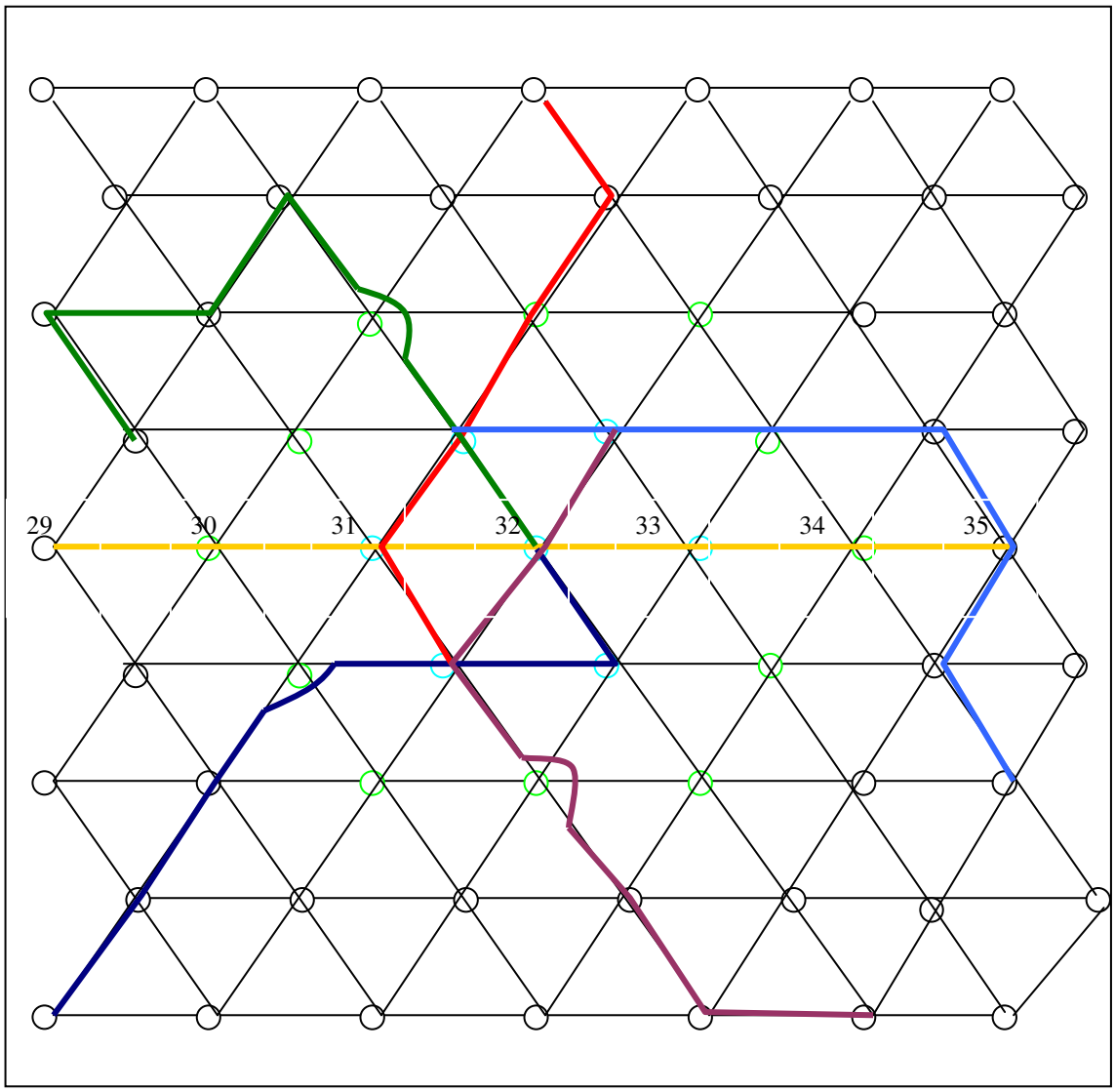
La introducción de esta nueva ruta sí provoca cambios en la geometría de las rutas primera y tercera (azul y verde)



2.6.2.1.2. CON LÍNEA DE METRO.

Veamos como afecta al caso anterior la introducción de una línea de metro bastante sencilla que atravesase la ciudad de Este a Oeste pasando por tres nodos del centro.

ruta_metro = [29 30 31 32 33 34 35]



Todas las rutas se ven afectadas por la línea de metro, variando sensiblemente su trayectoria, muy especialmente las rutas 3 y 5 (verde y morada, respectivamente).

2.6.2.2. MATRIZ DE DEMANDA DE VIAJES TRANSVERSAL.

Variando tan solo la longitud máxima de ruta (**L = 8**) con respecto a los ejemplos anteriores.

2.6.2.2.1. SIN LÍNEA DE METRO.

2.6.2.2.1.1. CASO 1: N = 5

Los cinco pares entre los que se comienza a generar las rutas son:

(22, 13)	**(60, 14)**	**(43, 16)**	**(55, 50)**	**(20, 2)**

Las rutas pasan siguen el siguiente itinerario:

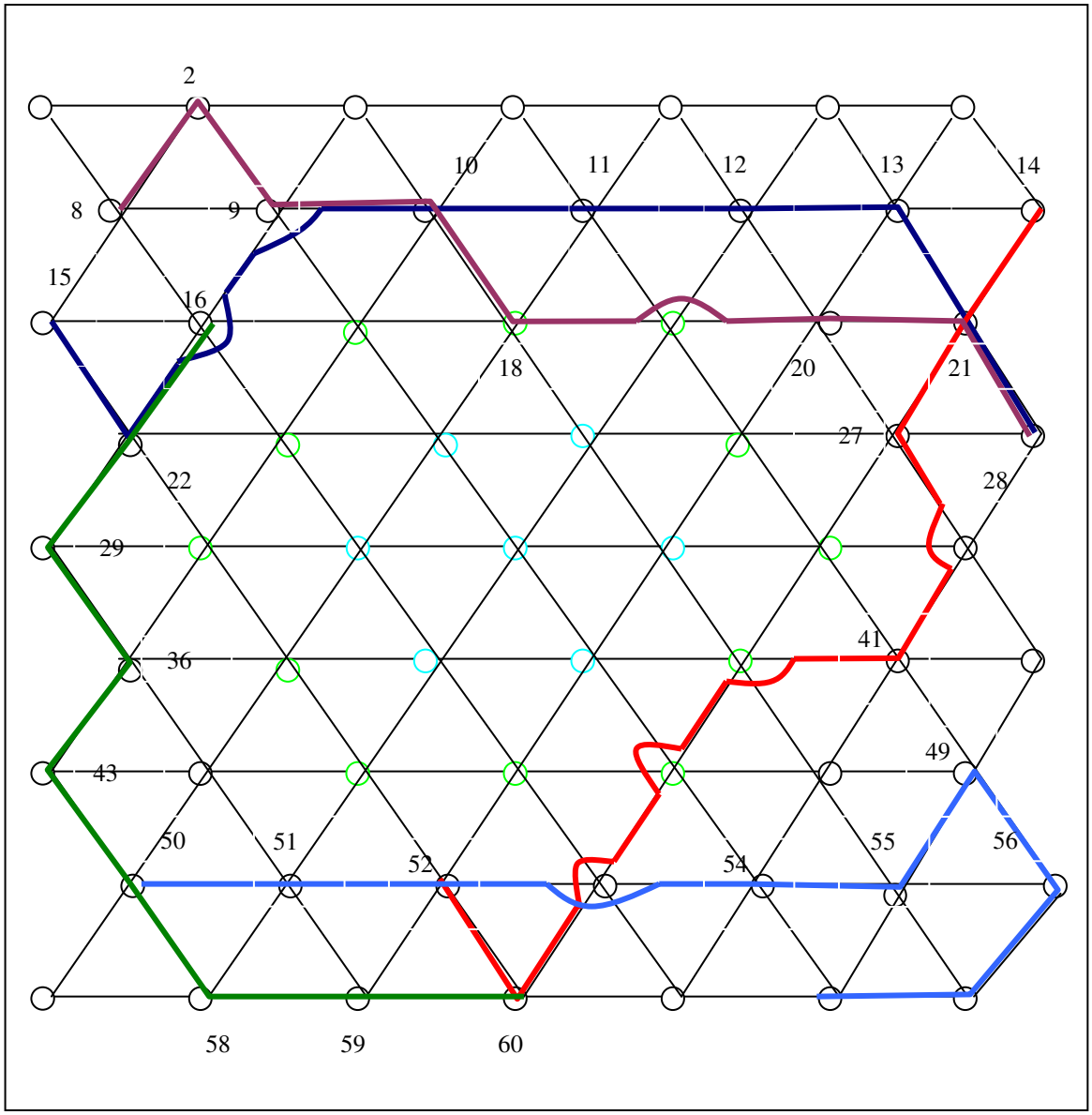
Ruta 1 = (15 22 10 11 12 13 21 28) **carga = 641**

Ruta 2 = (52 60 41 27 21 14) **carga = 784**

Ruta 3 = (60 59 58 50 43 36 29 22 16 9) **carga = 1363**

Ruta 4 = (62 63 56 49 55 54 52 51 50) **carga = 959**

Ruta 5 = (28 21 20 18 10 9 2 8) **carga = 804**


2
8
9
10
11
12
13
14
15
16
18
20
21
22
27
28
29
36
41
43
49
50
51
52
54
55
56
58
59
60

2.6.2.2.2. CON LÍNEA DE METRO.

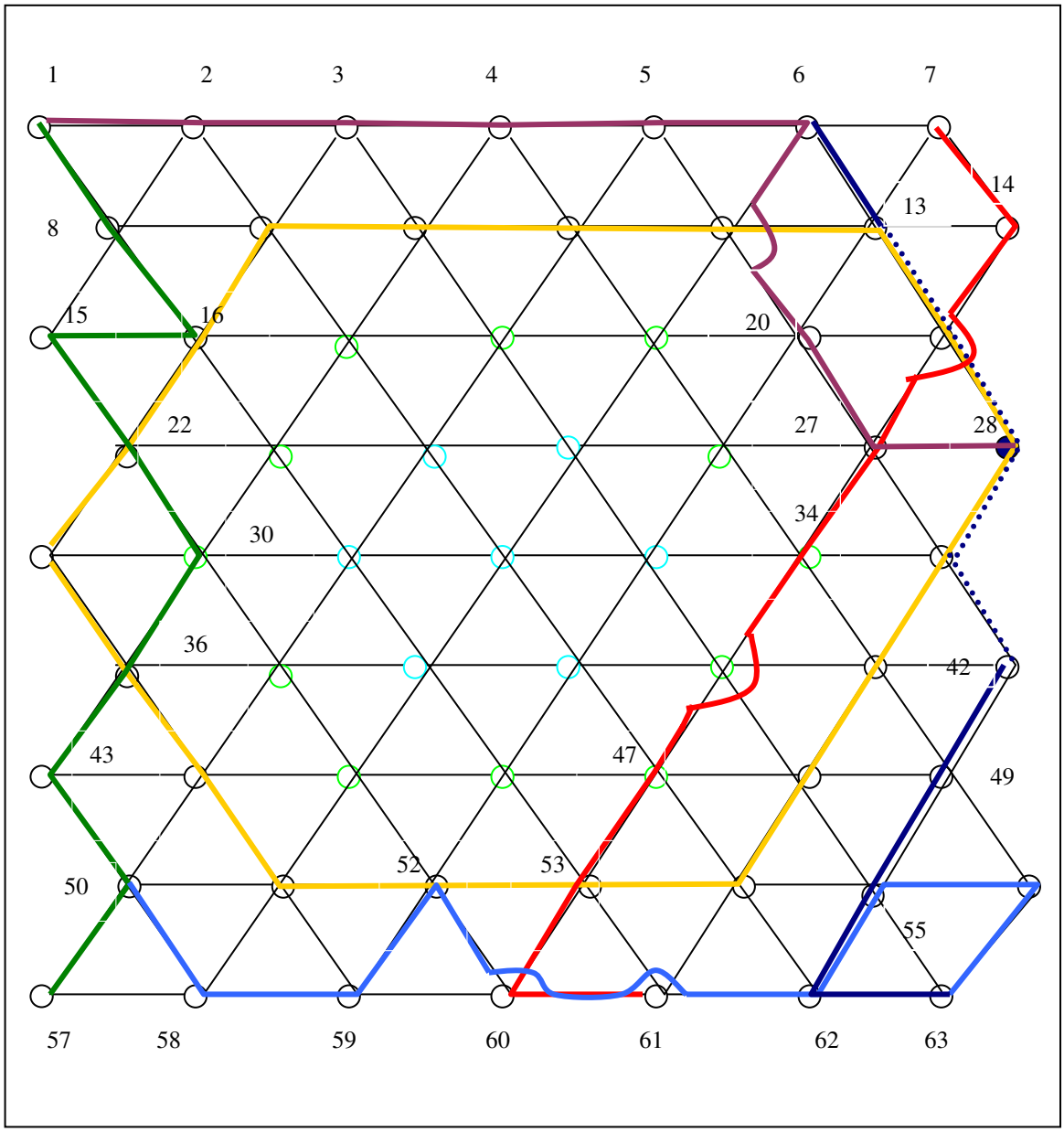
2.6.2.2.2.1. CASO 1: N = 5

Aquí se produce una circunstancia nueva, puesto que el par de nodos que proporcionaba una demanda mayor de viajes **(22, 13)**, y entre los que se iniciaba la primera ruta, ahora son dos nodos pertenecientes a la línea de metro, por lo que se procede a la elección de un nuevo par de nodos originadores de la que ahora será la quinta ruta. Este es el **(55, 6)**, por lo que se creará una nueva ruta (azul), completamente diferente a la anterior.

El resto de rutas también se verán bastante modificadas, en especial en los nodos que coinciden con las paradas de metro.

Ruta 1 = (63 62 55 49 42 28 13 6)	**carga = 1407**
Ruta 2 = (61 60 53 47 34 27 14 7)	**carga = 1483**
Ruta 3 = (57 50 43 36 30 22 15 16 8 1)	**carga = 1764**
Ruta 4 = (63 56 55 62 52 59 58 50)	**carga = 1496**
Ruta 5 = (28 27 20 6 5 4 3 2 1)	**carga = 2378**

El aumento de la carga soportada por las rutas se debe a la amplia cobertura de la línea de metro, y por lo tanto al aumento en el número de viajeros que viajarán en autobús para transbordar en el metro.



<u>NOTA:</u> En este ejemplo, se ha aumentado el valor de *n_entre* a 4, para poder apreciar mejor los cambios producidos.

La línea de puntos de la ruta azul pretende indicar que el nodo 13 debe ser unido al 42, pero sin que se produzcan paradas intermedias. El camino entre estos dos nodos debe ser uno de longitud mínima, por ejemplo el indicado, aunque puede haber otros puesto que las longitudes de los arcos son todas iguales.

2.6.3. CONFIGURACIÓN CIRCULAR de 121 NODOS.

En este caso, la distancia entre los nodos adyacentes no es la misma siempre. Éstas han sido halladas dando los siguientes valores a los diferentes radios de cada una de las cinco circunferencias:

$R_1 = 3$	$R_2 = 6$	$R_3 = 10$	$R_4 = 14$	$R_5 = 18$

9 nodos en la zona Centro
48 nodos en la zona Media
64 nodos en las zonas Periféricas

2.6.3.1. MATRIZ DE DEMANDA DE VIAJES GRAVITATORIA (RADIAL).

α_bus = 0.5	**α_metro = 0.8**	**n_entre = 3**	**L = 22**

2.6.3.1.1. SIN LÍNEA DE METRO.

2.6.3.1.1.1. CASO 1: N = 1 (una ruta de autobús).

Ruta 1 = [120 113 97 95 63 31]	
Longitud = 19.72	**carga = 869**

2.6.3.1.1.2. **CASO 2: N = 2**

Ruta 1 = [120 113 97 95 63 31]	**long = 19.72**	**carga = 1136**
Ruta 2 = [120 113 114 100 39 40]	**long = 20.83**	**carga = 1286**

2.6.3.1.1.3. CASO 3: N = 3

Ruta 1 = [121 113 112 95 63 31]	**long = 20.36**	**carga = 1772**
Ruta 2 = [120 113 114 100 39 40]	**long = 20.83**	**carga = 2050**
Ruta 3 = [120 121 117 81 82 50 51]	**long = 21.71**	**carga = 2270**

2.6.3.1.1.4. CASO 4: N = 4

Ruta 1 = [120 113 112 95 63 31]	**long = 19.72**	**carga = 2258**
Ruta 2 = [120 113 114 100 39 40]	**long = 20.83**	**carga = 2541**
Ruta 3 = [120 121 117 81 82 50 51]	**long = 21.71**	**carga = 2907**
Ruta 4 = [117 121 113 120 110 91 59]	**long = 21.72**	**carga = 2217**

Ver figura 2.6.3.1.1.4.

Los cambios en la estructura de las tres primeras rutas son apenas significativos, pues todavía son pocas para verse afectadas unas por otras. A partir de cinco rutas (ejemplo siguiente) se apreciarán algo más los cambios.

Lo que sí se puede notar claramente es que las rutas siempre intentan buscar un nodo de intersección con las demás, incrementándose así la carga de todas ellas, debido a los transbordos.

2.6.3.1.1.5. CASO 5: N = 5

Ruta 1 = [120 113 112 95 63 31]	**long = 19.72**	**carga = 2668**
Ruta 2 = [120 121 115 40 39]	**long = 21.71**	**carga = 2917**
Ruta 3 = [115 116 117 105 49 50 51]	**long = 21.22**	**carga = 3507**
Ruta 4 = [115 114 113 120 110 91 59]	**long = 20.44**	**carga = 3528**
Ruta 5 = [117 118 119 120 111 93 61 62]	**long = 20.83**	**carga = 4458**

Ver figura 2.6.3.1.1.5

2.6.3.1.2. CON LÍNEA DE METRO.

Nuevamente veremos el efecto que produciría sobre las cinco rutas creadas, el hecho de añadir una línea de metro que cruce la ciudad de Este a Oeste pasando por cinco nodos de la zona centro.

Ruta_metro = [25 57 89 109 119 120 113 114 115 101 73 41 9]		
Ruta 1 = [113 33 64 63 31]	**long = 20.5**	**carga = 3785**
Ruta 2 = [120 113 100 71 39 40]	**long = 20.83**	**carga = 3885**
Ruta 3 = [115 116 117 105 106 83 51 52]	**long = 20.83**	**carga = 5237**
Ruta 4 = [115 121 113 120 110 91 59]	**long = 21.72**	**carga = 5135**
Ruta 5 = [117 118 119 120 111 93 61 60]	**long = 20.83**	**carga = 5539**

Ver figura 2.6.3.1.2.

Se observa un incremento bastante importante de viajeros sobre todo en las líneas 3, 4 y 5 (celeste, verde y morada, respectivamente).

2.6.3.2. MATRIZ DE DEMANDA DE VIAJES TRANSVERSAL.

Estableciendo la longitud máxima de ruta en 28 unidades (**L = 28**).

2.6.3.2.1. SIN LÍNEA DE METRO.

Ruta 1 = [6 5 4 36 35 34 33 1 2]	**long = 26.84**	**carga = 1193**
Ruta 2 = [42 10 9 8 7 6 5 37]	**long = 25.65**	**carga = 991**
Ruta 3 = [41 42 43 44 45 46 47 49 17]	**long = 26**	**carga = 804**
Ruta 4 = [89 57 56 55 22 21 20 19]	**long = 26.84**	**carga = 691**
Ruta 5 = [57 58 59 60 61 62 63 64 32 1]	**long = 26.78**	**carga = 1395**

Ver figura 2.6.3.2.1.

2.6.3.2.2. CON LÍNEA DE METRO.

Introducimos ahora una línea circular de metro recorriendo gran parte de los nodos periféricos, que son entre los que se produce mayor demanda de viajes.

Ruta_metro = [33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64]]

Ruta 1 = [37 36 4 3 2 1 32 31 30]	**long = 27.93**	**carga = 2407**
Ruta 2 = [11 10 9 8 7 6 5 37 36]	**long = 27.93**	**carga = 2885**
Ruta 3 = [45 14 15 16 17 18 19 20]	**long = 27.93**	**carga = 2561**
Ruta 4 = [90 89 88 86 54 22 23 24 25]	**long = 26.43**	**carga = 2097**
Ruta 5 = [27 28 61 60 30 31 32]	**long = 24.87**	**carga = 1311**

Ver figura 2.6.3.2.2.

Nuevamente podemos comprobar las tendencias que ya se venían observando. Un incremento en la carga de las rutas, un desplazamiento de las rutas que coinciden en muchas de sus paradas con la línea de metro, y la búsqueda continua de intersecciones entre las rutas entre si y en especial con la línea de metro.

Figura 2.6.3.1.1.4.

Matriz de demanda gravitacional. N=4. Sin línea de metro.

Figura 2.6.3.1.1.5.

Matriz de demanda gravitacional. N=5. Sin línea de metro.

Figura 2.6.3.1.2.

Matriz de demanda gravitacional. N=5. Con línea de metro.

Figura 2.6.3.2.1.

Matriz de demanda transversal. N=5. Sin línea de metro.

Figura 2.6.3.2.2.

Matriz de demanda transversal. N=5. Con línea de metro.

2.6.4. CONFIGURACIÓN CIRCULAR - TRIANGULAR de 73 NODOS.

Con la intención de complicar algo más la geometría de estudio, vamos a suponer una configuración de la ciudad radial, pero con varias ramas oblicuas que permitan más interconexiones entre los diferentes nodos (algo también más cercano a la realidad).

En total hay 9 nodos en la zona centro, 16 nodos en la zona media y 48 nodos pertenecientes a las zonas periféricas.

Las distancias entre los nodos adyacentes son calculadas según el siguiente valor de cada uno de los radios de las cuatro circunferencias:

$R_1 = 4$	$R_2 = 9$	$R_3 = 14$	$R_4 = 18$

2.6.4.1. MATRIZ DE DEMANDA DE VIAJES GRAVITATORIA (RADIAL).

α_bus = 0.5	α_metro = 0.8	n_entre = 4	L = 24

2.6.4.1.1. SIN LÍNEA DE METRO.

2.6.4.1.1.1. CASO 1: N = 3

Ruta 1 = [71 73 67 53 25 5]	**long = 22**	**carga = 1718**
Ruta 2 = [71 72 65 64 46 15]	**long = 22.93**	**carga = 2057**
Ruta 3 = [72 65 68 55 29 30]	**long = 23.89**	**carga = 1392**

Ver figura 2.6.4.1.1.1.

2.6.4.1.1.2. CASO 2: N = 4

Ruta 1 = [71 73 67 53 25 5]	**long = 22**	**carga = 2638**
Ruta 2 = [73 65 72 63 46 15]	**long = 22.69**	**carga = 2387**
Ruta 3 = [65 73 68 55 29 28 27]	**long = 23.5**	**carga = 3494**
Ruta 4 = [65 72 71 70 69 57 32]	**long = 23.26**	**carga = 2817.3**

Ver figura 2.6.4.1.1.2.

2.6.4.1.1.3. CASO 3: N = 5

Ruta 1 = [71 73 67 53 25 5]	**long = 22**	**carga = 2470**
Ruta 2 = [66 65 72 63 46 45 15]	**long = 23.73**	**carga = 3739**
Ruta 3 = [65 73 68 55 29 28 27]	**long = 23.5**	**carga = 4300**
Ruta 4 = [65 72 71 70 69 57 32]	**long = 23.26**	**carga = 3836**
Ruta 5 = [67 66 65 71 61 42]	**long = 23.26**	**carga = 2730**

Ver figura 2.6.4.1.1.3

En este caso, podemos ver como solo la ruta 1 ha disminuido su carga, debido a que al añadir la quinta ruta, el algoritmo ha transformado la estructura de algunas de ellas, perdiéndose el transbordo entre las rutas 1 y 2 (azul y roja, respectivamente). Parece que esto no es la mejor solución. Éste es el caso más claro de incongruencia encontrada en todos los ensayos hasta ahora, es por ello que es importante recordar el carácter de "preplanificación" que tiene la aplicación de este algoritmo; y que, por supuesto, no siempre nos va a ofrecer las mejores soluciones.

2.6.4.1.2. CON LÍNEA DE METRO.

Introduciendo una línea de metro de Este a Oeste, pasando por cinco nodos del centro.

Ruta_metro = [13 41 61 71 72 65 66 67 53 25 5]

Ruta 1 = [69 70 71 72 63 45 15]	**long = 23.42**	**carga = 4507**
Ruta 2 = [65 73 68 55 29 30 31]	**long = 23.5**	**carga = 4302**
Ruta 3 = [71 70 58 35 34 33 32]	**long = 22.49**	**carga = 4976**
Ruta 4 = [67 73 71 61 41 42 43]	**long = 23.5**	**carga = 3189**
Ruta 5 = [65 73 70 59 36 10]	**long = 23.55**	**carga = 2964**

Ver figura 2.6.4.1.2.

En este caso, aparecen varias cosas interesantes:

La ruta que en el ejemplo anterior (sin línea de metro) era la 1 (azul), desaparece como tal, pues la pareja de nodos de flujo máximo que daban origen a la misma, **(71, 5)**, pertenece a la ruta de metro introducida. Por lo tanto, habrá que buscar otro par de nodos que inicien una nueva ruta.

El algoritmo obtiene como el siguiente par de "nodos originadores de ruta" al par **(71, 35)**, sin embargo el programa detecta que no es interesante iniciar una nueva ruta entre ambos, pues estos son fácilmente integrados en la ruta 3 (verde), que pasará a estar inicialmente constituida por los nodos **(71 70 35 32)**.

Así pues, es necesario buscar el siguiente par de nodos con una demanda de viajes mayor. Este es el **(71, 43)**, que como ocurría en el caso anterior puede ser integrado en la ruta 4 (morada), quedando ésta constituida (a la espera de incluir más nodos), por los nodos **(67 71 42 43)**.

El siguiente par de nodos con una demanda de viajes mayor es el **(65, 10)**. Éste sí cumple todos las condiciones para que entre ellos comience a crearse una ruta.

2.6.4.2. MATRIZ DE DEMANDA DE VIAJES TRANSVERSAL.

Aumentando la longitud de ruta máxima a 30 unidades.

2.6.4.2.1. SIN LÍNEA DE METRO.

2.6.4.2.1.1. CASO 1: N = 5

Ruta 1 = [6 26 25 24 4 3 2]	**long = 29.4**	**carga = 2776**
Ruta 2 = [59 58 34 35 36 10 11]	**long = 26.65**	**carga = 5120**
Ruta 3 = [11 37 38 39 40 41 42 43 44 14]	**long = 28.1**	**carga = 9109**
Ruta 4 = [31 56 73 65 49 17]	**long = 29.1**	**carga = 1147**
Ruta 5 = [37 59 70 73 66 52 23]	**long = 29.1**	**carga = 2831**

Ver figura 2.6.4.2.1.1.

Es interesante en este ejemplo, pararse a analizar el itinerario de las rutas 4 y 5 (verde y morada, respectivamente).

Centrémonos, por ejemplo, en la ruta 4 (verde). **(31, 17)** es el par de nodos que obliga a generar esta ruta. Probablemente si hubiéramos permitido una longitud de ruta mayor que 30, el itinerario seguido por la línea, sería a través de nodos de la periferia (como había venido ocurriendo en los ejemplos con matriz de demanda transversal mostrados hasta el momento); sin embargo, la limitación en la longitud máxima de ruta solo permite la unión entre ambos nodos a través de una ruta que acorte por el centro, aunque esto suponga pasar por nodos de las zonas media y centro que aportarán menos carga a la ruta. De hecho, se puede observar como ésta es la ruta con menor carga de viajeros. Muy pobre, comparada con las de las rutas 2 y 3 (roja y celeste, respectivamente). En este punto, el decisor o encargado debería decidir si es preferible no crear esta ruta, aunque se quede sin satisfacer la elevada demanda de viajes entre la pareja **(31, 27)**; o quizás, sea más interesante permitir una longitud de ruta mayor para que la ruta en cuestión pueda tener una estructura que le permita evitar los nodos del centro, y así, casi con total seguridad, ganar carga de viajeros.

También resulta interesante indicar que los pares de nodos con una demanda de viajes elevada: **(6, 24)** y **(59, 11)** son introducidos en las rutas 1 (azul) y 2 (roja), respectivamente, pues son fácilmente asimilables por éstas, sin necesidad de tener que iniciar nuevas rutas.

2.6.4.2.2. CON LÍNEA DE METRO.

Introducimos ahora una línea circular de metro recorriendo gran parte de los nodos periféricos, que son entre los que se produce mayor demanda de viajes.

Ruta_metro = [17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 17]

Ruta 1 = [6 26 25 24 4 3 2]	**long = 29.4**	**carga = 1497**
Ruta 2 = [59 58 35 10 11 38 12]	**long = 29.3**	**carga = 1344**
Ruta 3 = [42 41 13 14 15 16]	**long = 28.05**	**carga = 1407**
Ruta 4 = [59 57 56 53 52 51]	**long = 29.5**	**carga = 805**
Ruta 5 = [1 2 3 22 23 24 5]	**long = 29.4**	**carga = 1589**

Ver figura 2.6.4.2.2.

En este caso, las rutas 4 y 5 (verde y morada, respectivamente) del ejemplo anterior, desaparecerán, pues los nodos que las generaban **(31, 17)** y **(37, 23)**, están englobados dentro de la línea de metro. Iniciándose, entonces, la generación de nuevas rutas entre **(59, 23)** y **(23, 5)**.

Obsérvese cómo la introducción de la línea de metro, alivia la excesiva carga que soportaban algunas rutas. El caso más significativo es el de la ruta 3 (celeste), que se ha visto desplazada paralelamente a la línea de metro, asumiendo ésta, la gran carga que mantenía la ruta de autobús.

Figura 2.6.4.1.1.1.

Matriz de demanda gravitacional. N = 3. Sin línea de metro.

Figura 2.6.4.1.1.2.

Matriz de demanda gravitacional. N = 4. Sin línea de metro.

Figura 2.6.4.1.1.3.

Matriz de demanda gravitacional. N = 5. Sin línea de metro.

Figura 2.6.4.1.2.

Matriz de demanda gravitacional. N = 5. Con línea de metro.

Figura 2.6.4.2.1.1.

Matriz de demanda transversal. N = 5. Sin línea de metro.

Figura 2.6.4.2.2.

Matriz de demanda transversal. N = 5. Con línea de metro.

2.6.5. CONCLUSIONES DE LAS PRUEBAS REALIZADAS SOBRE LAS DIFERENTES CONFIGURACIONES.

- Para todas las configuraciones probadas, se observa un comportamiento análogo.
- Las rutas que el algoritmo va creando responden con lógica a los flujos radiales o transversales, según el caso.
- Conforme van añadiéndose rutas, éstas van buscando la intersección entre ellas, con la intención de cubrir la mayor demanda posible mediante el transbordo.
- La adición de nuevas rutas puede provocar la modificación de rutas anteriormente creadas, como era de esperar. Es decir, no se trata de añadir una nueva línea a las ya existentes, sino de crear todas las rutas simultáneamente. Esto puede implicar que la primera ruta en iniciarse, por tener una demanda de viajes mayor entre sus dos nodos iniciales, no tiene porque ser la que se "cierre" primero, ni por supuesto, la que finalmente tenga una mayor carga de pasajeros.
- La introducción de la línea de metro, modifica la estructura de las rutas de autobús.
- Las rutas de autobús, tienden a buscar algún nodo en común con la línea de metro.
- Las rutas de autobús que estaban trazadas sobre el itinerario que sigue la línea de metro, abandonan este camino buscando otro alternativo, normalmente desviando su ruta paralelamente a la de metro.
- El hecho de que el algoritmo esté ideado de manera que no se permita más de un número de nodos intermedios entre los nodos generadores de la ruta, provoca que haya nodos por los que pasen las rutas pero que no sean paradas de las mismas. Un

estudio más detallado sobre la situación real debiera plantearse si es conveniente o no la adición de estos nodos como paradas, atendiendo a los tiempos de viajes y la cobertura.

- La limitación en la longitud de las rutas evita que se creen rutas excesivamente largas. En el algoritmo diseñado, normalmente éste es el factor que hace que se "cierren" las rutas.
- Normalmente, el hecho de añadir nuevas rutas supone un incremento en la carga de todas las rutas por el hecho del incremento de viajeros que pueden transbordar.
- En ocasiones puede que la adición de una nueva ruta provoque la disminución de carga de alguna otra con la que comporta varios nodos, debido al reparto de pasajeros entre ambas.