Trabajo Fin de Máster
Ingeniería Industrial

El problema de flujo máximo a coste mínimo dependiente del flujo

Autor: Alberto Domínguez Palma
Tutor: Francisco García Benítez

Dpto. de Ingeniería y Ciencia de los Materiales y del Transporte
Área: Ingeniería e Infraestructura de los Transportes
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Máster
Ingeniería Industrial

El problema de flujo máximo a coste mínimo dependiente del flujo

Autor:
Alberto Domínguez Palma

Tutor:
Francisco García Benítez
Catedrático

Dpto. de Ingeniería y Ciencia de los Materiales y del Transporte
Área: Ingeniería e Infraestructura de los Transportes
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Trabajo Fin de Máster: El problema de flujo máximo a coste mínimo dependiente del flujo

Autor: Alberto Domínguez Palma

Tutor: Francisco García Benítez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

Agradecimientos

Este Trabajo Fin de Máster supone la culminación de años de esfuerzo y sacrificio en mi formación académica, pero también será el comienzo de nuevos retos personales y profesionales. Quisiera aprovechar estas líneas para transmitir mi más sincero agradecimiento a quiénes han estado presentes en esta etapa.

En primer lugar, a mi tutor, el catedrático Francisco García Benítez, por su implicación en el desarrollo de este Trabajo y por su generosidad a la hora de compartir sus conocimientos.

También, indudablemente, a mi familia, por contribuir día a día a que hoy esté donde estoy, apoyándome en todo momento y brindándome toda su confianza en mis capacidades.

Y a mis amigos, que siempre apostaron por mí, y que me dieron algunas ideas y, sobre todo, ánimos.

Por supuesto, no quiero olvidarme de todos los profesores y compañeros que han formado parte de mi vida académica a lo largo de estos años, cuya aportación también ha sido importante en mi crecimiento personal y profesional.

A todos, gracias.

Alberto Domínguez Palma

Sevilla, 2021

Resumen

El problema de flujo máximo a coste mínimo juega un papel esencial en la optimización de redes. El objetivo principal es determinar el coste mínimo para enviar el máximo flujo posible a través de una red. Estas redes de transporte pueden representar multitud de problemas reales, como el transporte de mercancías o personas, flujo en tuberías, redes de distribución eléctrica, etc.

En este Trabajo se explica la resolución del problema empleando el algoritmo Primal-Dual, así como su implementación en MATLAB. Se diferenciarán dos versiones del algoritmo: costes constantes o dependientes del flujo.

Se exige que los costes no constantes sean continuos y estrictamente convexos. En ese caso, la estrategia es resolver de manera sucesiva el problema tomando los costes marginales de los arcos como constantes en cada iteración.

El algoritmo Primal-Dual no es el único que permite resolver el problema de flujo máximo a coste mínimo. Existen otros algoritmos competitivos como el algoritmo de cancelación de ciclo, los algoritmos basados en ténicas de escalado o algoritmos no lineales como los algoritmos de colonización de hormigas o los algoritmos genéticos.

Abstract

Minimum cost flow problem plays an essential role in network optimization. The main objective is to determine the minimum cost to send the maximum possible flow through a network. These transport networks can represent a multitude of real problems, such as the transport of commodities or people, flow in pipelines, electrical distribution networks, etc.

This work explains the resolution of the problem using the Primal-Dual algorithm, as well as its implementation in MATLAB. Two versions of the algorithm will be differentiated: constant or flow-dependent costs.

Non-constant costs are required to be continuous and strictly convex. In this case, the strategy is to successively solve the problem by taking the marginal costs of the arcs as constant in each step.

Primal-Dual algorithm is not the only one that allows solving the minimum cost flow problem. There are other competitive algorithms such as the cycle cancelying algorithm, algorithms based on scaling techniques or non-linear algorithms such as ant colony optimization algorithm or genetic algorithms.

Índice

Agradecimientos **vii**

Resumen **ix**

Abstract **xi**

Índice **xiii**

Índice de Tablas **xv**

Índice de Figuras **xvii**

Notación **xix**

1 Introducción **1**

2 Algoritmo Primal-Dual con costos independientes del flujo **3**

2.1. Introducción de los datos de la red. *4*

2.2. Estandarización del problema. *5*

2.2.1 Cálculo de ofertas y demandas. 11

2.2.2 Método 1. Equilibrado de la red. 12

2.2.3 Método 2. Equilibrado de la red. 16

2.2.4 Método 3. Conversión en una red de transbordo. 19

2.3. Algoritmo Primal-Dual con costes constantes. *27*

2.3.1 Cálculo de la matriz reducida. 30

2.3.2 Asignación directa de flujos. 31

2.3.3 Algoritmo de Ford-Fulkerson. Proceso de marcaje. 34

2.3.4 Modificación de la matriz reducida. 42

2.3.5 Algoritmo de Ford Fulkerson. Aumento de flujos. 46

2.4. Post-procesado del resultado. *49*

2.5. Resumen de la programación del algoritmo PDTCI. *54*

2.6. Aplicación práctica del código implementado. *57*

2.6.1 Resolución del problema estandarizando la red con el método 1. 58

2.6.2 Resolución del problema estandarizando la red con el método 2. 70

2.6.3 Resolución del problema estandarizando la red con el método 3. 80

3 Algoritmo Primal-Dual con Costos Dependientes del flujo **93**

3.1. Introducción de datos de la red. *95*

3.2. Cálculo de costes marginales. *99*

3.3. Estandarización del problema. *104*

3.4. Algoritmo Primal-Dual para costes no constantes. *109*

3.4.1 Modificación de las matrices de costos marginales y capacidades superiores. 112

3.5. Post-procesado del resultado. *116*

3.6. Resumen de la programación del algoritmo PDTCD. *120*

3.7. Aplicación práctica del código implementado. *122*

4 Algoritmos competitivos y Revisión del Estado del Arte **137**

4.1. Algoritmos básicos. *138*

4.1.1 Algoritmo de cancelación de ciclo. 140

4.1.2 Algoritmo de ruta mínima sucesiva. 141

4.1.3 Algoritmo de relajación. 143
4.1.4 Comparación de algoritmos básicos. 145
4.2. Algoritmos polinomiales. *146*
4.2.1 Algoritmo de escalado en capacidades. 146
4.2.2 Algoritmo de escalado en costes. 147
4.2.3 Algoritmo de doble escalado. 149
4.2.4 Otros algoritmos de escalado. 150
4.3. Algoritmos para costes no lineales. *151*
4.3.1 Aproximación lineal de los costes. 152
4.3.2 Otros tipos de algoritmos. 153

5 Conclusiones **155**

Apéndice A.- Scripts de MATLAB **157**

Apéndice B.- Resolución completa de un ejemplo (algoritmo PDTCI) **159**
B.1. Método 1. *160*
B.2. Método 2. *170*
B.3. Método 3. *179*

Apéndice C.- Resolución completa de un ejemplo (algoritmo PDTCD) **191**
C.1. Método 1. *193*
C.2. Método 2. *224*
C.3. Método 3. *255*

Referencias **281**

ÍNDICE DE TABLAS

Tabla 1.- Equilibrado de redes. 7
Tabla 2.- Tabla de transporte asociada a la Figura 4. 14
Tabla 3.- Tabla de transporte asociada a la red de la Figura 5. 17
Tabla 4.- Tabla de transporte asociada a la red de la Figura 8. 26
Tabla 5.- Subrutinas de la función "pdtci". 28
Tabla 6.- Primera iteración "pdtci" para la red de la Figura 4. Celdas básicas. 32
Tabla 7.- Primera iteración "pdtci" para la red de la Figura 4. Asignación directa. 33
Tabla 8.- Ejemplo de marcaje (Ford-Fulkerson). Paso 1. 34
Tabla 9.- Ejemplo de marcaje (Ford-Fulkerson). Paso 2. 35
Tabla 10.- Ejemplo de marcaje (Ford-Fulkerson). Paso 3. 35
Tabla 11.- Ejemplo de marcaje (Ford-Fulkerson). Paso final. 36
Tabla 12.- Conjuntos I, J, I y J. 42
Tabla 13.- Nueva tabla de transporte tras modificar la matriz reducida (Figura 4). 47
Tabla 14.- Algoritmo de Ford-Fulkerson. Marcaje sobre la Tabla 13. 47
Tabla 15.- Incremento de flujos sobre la Tabla 13. 48
Tabla 16.- Variables de entrada y salida para las subrutinas del algoritmo PDTCI. 56
Tabla 17.- Subrutinas de la función "pdtcd". 110
Tabla 18.- Flujo en la primera iteración del algoritmo PDTCD para la red de la Figura 26. 114
Tabla 19.- Variables de entrada y salida para las subrutinas del algoritmo PDTCD. 121
Tabla 20.- Subrutinas (PDTCD) ejecutadas en la resolución de la red de la Figura 33 (método 3). 133
Tabla 21.- Resumen de los algoritmos de tiempo pseudopolinomial [3]. 145
Tabla 22.- Tiempos de ejecución de los algoritmos polinomiales [7]. 151
Tabla 23.- Función de costes absolutos lineal convexa [2]. 152
Tabla 24.- Subrutinas (PDTCD) ejecutadas en la resolución de la red de la Figura 33 (método 1). 195
Tabla 25.- Subrutinas (PDTCD) ejecutadas en la resolución de la red de la Figura 33 (método 2). 225

ÍNDICE DE FIGURAS

Figura 1.- Ejemplo de red de transporte. 5
Figura 2.- Grafo de la red de la Figura 1, obtenido con la función “estandarizar”. 8
Figura 3.- Grafo realista equivalente al grafo de la Figura 2. 9
Figura 4.- Red de la Figura 1, equilibrada según el método 1. 14
Figura 5.- Red de la Figura 1, equilibrada según el método 2. 17
Figura 6.- Estructura de una red genérica, estandarizada según el método 3. 19
Figura 7.- Estructura de la tabla de transporte asociada a la red genérica de la Figura 6. 19
Figura 8.- Red de la Figura 1, estandarizada según el método 3. 25
Figura 9.- Diagrama de flujo de la función “pdtci”. 29
Figura 10.- Flujo óptimo de la red de la Figura 1 (grafo MATLAB). 53
Figura 11.- Flujo óptimo de la red de la Figura 1. 54
Figura 12.- Diagrama de flujo del algoritmo PDTCI. 55
Figura 13.- Red de transporte a resolver en el Apartado 2.6. 57
Figura 14.- Grafo de la Figura 13, estandarizado mediante el método 1. 58
Figura 15.- Grafo de la Figura 13, representado en MATLAB. 60
Figura 16.- Grafo realista equivalente al grafo de la Figura 15. 60
Figura 17.- Flujo óptimo en la red de la Figura 13 (método 1). 69
Figura 18.- Grafo realista equivalente al grafo de la Figura 17. 70
Figura 19.- Grafo de la Figura 13, estandarizado mediante el método 2. 71
Figura 20.- Flujo óptimo en la red de la Figura 13 (método 2). 79
Figura 21.- Grafo realista equivalente al grafo de la Figura 20. 80
Figura 22.- Grafo de la Figura 13, estanadarizado mediante el método 3. 81
Figura 23.- Flujo óptimo en la red de la Figura 13 (método 3). 91
Figura 24.- Grafo realista equivalente al grafo de la Figura 23. 91
Figura 25.- Ejemplo de red de transporte con arcos con costes dependientes del flujo. 97
Figura 26.- Red de transporte de la Figura 25, estandarizada según el método 3. 106
Figura 27.- Grafo de la Figura 25, representado en MATLAB. 107
Figura 28.- Grafo realista equivalente al grafo de la Figura 27. 108
Figura 29.- Diagrama de flujo de la función “pdtcd”. 111
Figura 30.- Flujo óptimo de la red de la Figura 25. 118
Figura 31.- Grafo realista equivalente al grafo de la Figura 30. 119
Figura 32.- Diagrama de flujo del algoritmo PDTCD. 120
Figura 33.- Red de transporte a resolver en el Apartado 3.7. 122
Figura 34.- Grafo de la Figura 33, estandarizado según el método 3. 124

Figura 35.- Grafo de la Figura 33, representado en MATLAB. 126
Figura 36.- Grafo realista equivalente al grafo de la Figura 35. 126
Figura 37.- Flujo óptimo, calculado con MATLAB, para la red de la Figura 33. 135
Figura 38.- Grafo realista equivalente al grafo de la Figura 37. 135
Figura 39.- Pseudocódigo del algoritmo de cancelación de ciclo [3]. 140
Figura 40.- Pseudocódigo del algoritmo de ruta mínima sucesiva [3]. 142
Figura 41.- Pseudocódigo del algoritmo de relajación [3]. 144
Figura 42.- Pseudocódigo de las subrutinas del algoritmo de relajación [3]. 144
Figura 43.- Pseudocódigo del algoritmo de escalado en capacidades [7]. 147
Figura 44.- Pseudocódigo del algoritmo de escalado en costes [7]. 148
Figura 45.- Pseudocódigo del proceso de aproximación y mejora en el escalado en costes [7]. 148
Figura 46.- Pseudocódigo del proceso de aproximación y mejora en el doble escalado [7]. 149
Figura 47.- Datos en los arcos paralelos correspondientes al arco (i,j) [2]. 152
Figura 48.- Red de transporte a resolver en el Apéndice B. 159
Figura 49.- Grafo de la Figura 48, estandarizado mediante el método 1. 161
Figura 50.- Grafo de la Figura 48, representado en MATLAB. 161
Figura 51.- Grafo realista equivalente al grafo de la Figura 50. 162
Figura 52.- Flujo óptimo en la red de la Figura 48 (método 1). 169
Figura 53.- Grafo realista equivalente al grafo de la Figura 52. 169
Figura 54.- Grafo de la Figura 48, estandarizado mediante el método 2. 170
Figura 55.- Flujo óptimo en la red de la Figura 48 (método 2). 178
Figura 56.- Grafo realista equivalente al grafo de la Figura 55. 178
Figura 57.- Grafo de la Figura 48, estandarizado mediante el método 3. 179
Figura 58.- Flujo óptimo en la red de la Figura 48 (método 3). 188
Figura 59.- Grafo realista equivalente al grafo de la Figura 58. 189
Figura 60.- Grafo de la Figura 33, estandarizado mediante el método 1. 193
Figura 61.- Flujo óptimo en la red de la Figura 33 (método 1). 223
Figura 62.- Grafo realista equivalente al grafo de la Figura 61. 223
Figura 63.- Grafo de la Figura 33, estandarizado mediante el método 2. 224
Figura 64.- Flujo óptimo en la red de la Figura 33 (método 2). 254
Figura 65.- Grafo realista equivalente al grafo de la Figura 64. 255

Notación

G	Grafo orientado
n	Número de nodos de un grafo G
l	Número de arcos de un grafo G
K_j	Nivel de oferta/demanda de un nodo j
y_{ij}	Flujo sobre un arco (i,j)
L_{ij}	Capacidad inferior de un arco (i,j)
U_{ij}	Capacidad superior de un arco (i,j)
c_{ij}	Coste absoluto unitario de un arco (i,j)
z	Función objetivo
z'	Función objetivo estandarizada
x_{ij}	Flujo sobre un arco (i,j) en una red estándar
u_{ij}	Capacidad superior de un arco (i,j) en una red estándar
k_j	Nivel de oferta/demanda de un nodo j en una red estándar
$A(j)$	Conjunto de arcos con destino en el nodo j
$D(j)$	Conjunto de arcos con origen en el nodo j
r_{ij}	Coste relativo del arco (i,j)
Δ	Incremento de flujo admisible
∂z_{ij}	Coste marginal de un arco (i,j)
λ_j	Variable dual asociada a la restricción de conservación del nodo j
α_{ij}	Variable dual asociada a la restricción de capacidad del arco (i,j)
e_j	Desequilibrio del nodo j
E	Conjunto de nodos con exceso
D	Conjunto de nodos con déficit
$w(\lambda)$	Función objetivo del problema relajado
$S(\Delta)$	Conjunto de nodos con exceso mayor que Δ
$T(\Delta)$	Conjunto de nodos con déficit mayor que Δ

1 INTRODUCCIÓN

El interés por dar solución a los problemas de transporte, tanto de personas como de materiales, es tan antiguo como la propia humanidad [35]

- Francisco García Benítez -

El problema de flujo máximo a coste mínimo es uno de los distintos tipos de problemas de transporte que existen, en el que el objetivo es determinar el mínimo coste para enviar un determinado flujo de un bien desde una serie de nodos de oferta hacia una serie de nodos de demanda, a través de una red orientada donde cada arco tiene asignado un coste y una restricción de capacidad. En este sentido, el problema de flujo máximo a coste mínimo juega un papel esencial en la optimización de redes, pues otros problemas como el problema de transporte, el problema de flujo máximo o el de ruta mínima, pueden considerarse como casos particulares de éste.

El problema se modela mediante una red de transporte, en la que los distintos centros de transporte (nodos) se unen según unas ciertas rutas definidas (arcos). Estas redes de transporte pueden representar multitud de problemas reales, como el transporte de mercancías o personas, flujo en tuberías, redes de distribución eléctrica o de telecomunicaciones, planificación temporal de las tareas a ejecutar en un determinado proyecto, entre otros.

La red de transporte no será más que un grafo orientado G de *n* nodos, a cada uno de los cuales se le asocia un valor K_j, que indica el nivel de oferta o demanda de dicho nodo. Si $K_j > 0$, j es un nodo de oferta. Si $K_j < 0$, entonces j es un nodo de demanda. Si $K_j = 0$, el nodo es de transbordo.

Además, a cada arco (i,j) del grafo G se le asocia una variable y_{ij}, no negativa, que indicará el flujo que circula por dicho arco, que se encontrará acotado inferior y superiormente por unos límites definidos con las variables L_{ij} y U_{ij}, respectivamente. Así mismo, el costo unitario de cada arco vendrá representado por la variable c_{ij}.

Por otra parte, según las leyes de Kirchhoff, el flujo en cada nodo debe conservarse. Esto implica que, para cada nodo, se define una restricción que impone que el flujo que sale es igual al que entra más el producido menos el consumido.

Finalmente, el objetivo del problema es minimizar el coste total de la red, por lo que el modelo matemático que define el problema de flujo máximo a coste mínimo vendrá dado por:

$$\begin{aligned} & Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot y_{ij} \\ & s.a. \sum_{k\in D(j)} y_{jk} - \sum_{i\in A(j)} y_{ij} = K_j \quad \forall j = 1,2,\dots,n \\ & L_{ij} \le y_{ij} \le U_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{1}$$

La resolución de este problema puede realizarse mediante diversos algoritmos de programación lineal recogidos en la bibliografía y que aparecerán a lo largo del este trabajo. Entre ellos, se destaca inicialmente el algoritmo Primal-Dual aplicado a la Tabla de Transporte, que será el que aquí se emplee para resolver el problema.

Sin embargo, es habitual en las aplicaciones reales que el coste c_{ij} asociado a cada arco (i,j) no sea constante, sino que tenga una cierta dependencia del flujo. Por ejemplo, en redes de tráfico, es crucial considerar que la presencia de un gran flujo de vehículos en un cierto trayecto encarece los costes de transporte a través del mismo.

La formulación matemática de este problema será:

$$\begin{aligned} & Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}(y_{ij}) \cdot y_{ij} \\ & s.a. \sum_{k\in D(j)} y_{jk} - \sum_{i\in A(j)} y_{ij} = K_j \quad \forall j = 1,2,\dots,n \\ & L_{ij} \le y_{ij} \le U_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{2}$$

En ese caso, el problema se convierte en no lineal, por lo que los algoritmos de programación lineal no son aplicables de manera directa. Una posible estrategia a seguir es transformar este problema con costes dependientes del flujo en sucesivos problemas que puedan resolverse en el ámbito de los costes constantes. Otra opción es aplicar directamente algoritmos de programación no lineal.

El objetivo principal del presente trabajo consistirá en abordar la implementación en MATLAB de la resolución del problema (2) mediante el algoritmo Primal-Dual aplicado a la Tabla de transporte.

Para ello, en el Capítulo 2 se abordará la implementación del algoritmo Primal-Dual aplicado a la Tabla de transporte con Costes Independientes del flujo (en adelante, de manera abreviada, algoritmo PDTCI).

En el Capítulo 3, se explicarán las modificaciones y añadidos al código MATLAB del algoritmo PDTCI para permitir la resolución del problema con costes no constantes. De esta manera, se obtiene el algoritmo Primal-Dual aplicado a la Tabla de transporte con Costos Dependientes del flujo (en adelante, algoritmo PDTCD).

En el Capítulo 4 se indicarán otros algoritmos recogidos en la bibliografía, presentando una revisión descriptiva del estado del arte actual. Se describirán tanto algoritmos lineales como algoritmos no lineales, haciendo una comparación entre ellos desde el punto de vista computacional.

El Capítulo 5 recogerá las conclusiones alcanzadas en el desarrollo de este Trabajo Fin de Máster.

Finalmente, se incluyen como Apéndice ejemplos de resolución de una red con cada uno de los algoritmos (PDTCI y PDTCD).

2 ALGORITMO PRIMAL-DUAL CON COSTOS INDEPENDIENTES DEL FLUJO

Como se ha comentado anteriormente, la resolución del problema de flujo máximo a coste mínimo con costos dependientes del flujo, exige la resolución de varios problemas de flujo máximo a coste mínimo con costos constantes de manera sucesiva.

Por esta razón, a lo largo de este apartado se abordará la implementación en MATLAB de la resolución del problema mediante el algoritmo Primal-Dual aplicado a la Tabla de transporte con Costos Independientes del flujo (algoritmo PDTCI).

Para ello, en primer lugar, se hará un desarrollo explicativo de la filosofía seguida para la programación del algoritmo, indicando el código de los programas y comentándolos. El capítulo finalizará con la aplicación práctica de dicho programa de MATLAB para la resolución de una red de acuerdo con el algoritmo PDTCI.

Haciendo una breve recapitulación teórica, las distintas etapas que deben ejecutarse para resolver el problema con costes constantes según el algoritmo PDTCI son los siguientes:

1. Construir la tabla de transbordo asociada a la red que se va a resolver, y a continuación, estandarizar dicha tabla, realizando los cambios de variable pertinentes para convertir las capacidades inferiores, L_{ij}, en cero, y equilibrando y orlando ofertas y demandas para permitir la asignación de flujos.
2. Calcular la matriz reducida de la matriz de costes (es decir, los costos relativos), para obtener las celdas que forman parte de la red básica y asignar flujos en dichas celdas de la tabla, si es posible.
3. Realizar el proceso de marcaje de los nodos de acuerdo al algoritmo Ford-Fulkerson. Si, finalizado el proceso de marcaje, se alcanza un nodo de entrada, ir al paso 4. En caso contrario, ir al paso 5.
4. Como se ha alcanzado un nodo de entrada desde el nodo de salida, incrementar flujo en la tabla. Volver al paso 3.
5. Al no alcanzar un nodo de entrada desde el nodo de salida, modificar la matriz reducida para definir nuevas celdas en la red básica. Comprobar si, en estas nuevas celdas, puede asignarse flujo de manera directa, y volver al paso 3.
6. El algoritmo finaliza bien cuando todas las ofertas y demandas estén saturadas, bien cuando no sea posible modificar la matriz reducida.

El desarrollo teórico y la justificación matemática de cada uno de los pasos anteriores pueden encontrarse de manera muy detallada en la literatura sobre problemas de redes de transporte.

Respecto a la implementación en MATLAB que se presentará, las funciones que permiten resolver el problema son las siguientes:

```
[U,C,of,dem]=estandarizar(arcos,K,metodo)
[flujo,mr]=pdtci(C,of,dem,U)
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

Es decir, que introduciendo los datos de la red en las variables "arcos" y "K", de la forma en que se explicará a continuación, y copiando en la ventana de comandos de MATLAB las tres líneas de código anteriores, se puede obtener el flujo máximo a coste mínimo que transcurre por la red, así como los niveles reales de oferta y demanda de cada nodo de la red y el valor de la función objetivo global.

A continuación, se explicarán cada una de las funciones y subrutinas que componen cada una de las líneas de código anteriores, así como los cálculos que se realizan.

2.1. Introducción de los datos de la red.

Sea una red de transporte representada por un grafo G de *n* nodos y *l* arcos, con demandas K_j para cada nodo j, y capacidades inferiores L_{ij}, capacidades superiores U_{ij} y costes c_{ij} para cada arco (i,j), la información de los arcos de la red se introducirá en MATLAB definiendo las variables "arcos" y "K", cuyo contenido se describe a continuación:

- Matriz "arcos":

  La información relativa a todos los arcos de la red se transformará en una matriz de *l* filas y 5 columnas, de manera que cada fila de esta matriz recoge la información completa de un arco de la red. En concreto, la información se distribuye en cada columna de la matriz "arcos" de acuerdo a la estructura presentada a continuación.

$$arcos = \begin{bmatrix} nodo\ origen & nodo\ destino & capacidad\ inferior & capacidad\ superior & coste \\ i & j & L_{ij} & U_{ij} & c_{ij} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad (3)$$

  Así, además de tener la información de cada arco por filas, se tiene, por ejemplo, un vector de costes en la última columna de la matriz. Esto permitirá simplificar la programación del cálculo de la matriz de costes o de capacidades en la función "estandarizar", como podrá verse en el apartado correspondiente a dicha función.

- Vector "K"

  Los niveles de oferta y demanda K_j asociados a cada nodo se recogen en un vector de *n* elementos.

$$K = [K_1 \quad K_2 \quad \dots \quad K_n] \quad (4)$$

Ejemplo.- *Sea la red de transporte representada con el siguiente grafo:*

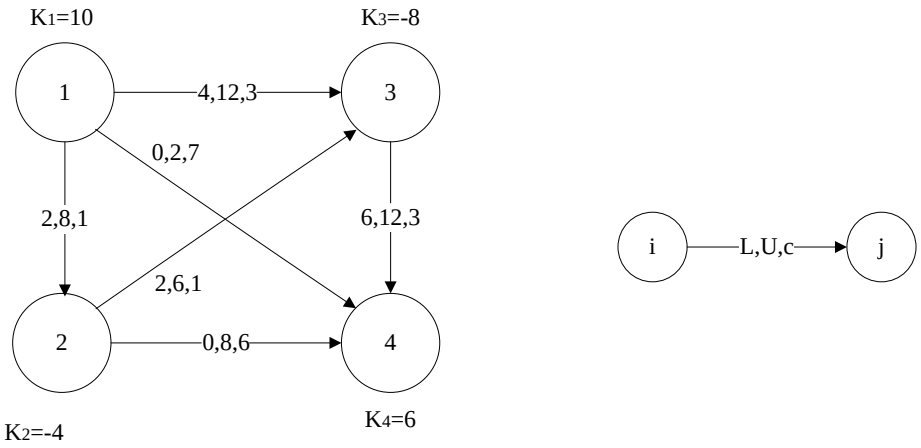


Figura 1.- Ejemplo de red de transporte.

La matriz "arcos"(3) y el vector "K" (4) correspondientes, serían:

$$arcos = \begin{bmatrix} 1 & 2 & 2 & 8 & 1 \\ 1 & 3 & 4 & 12 & 3 \\ 1 & 4 & 0 & 2 & 7 \\ 2 & 3 & 2 & 6 & 1 \\ 2 & 4 & 0 & 8 & 6 \\ 3 & 4 & 6 & 12 & 3 \end{bmatrix}$$

$$K = [10 \quad -4 \quad -8 \quad 6]$$

2.2. Estandarización del problema.

El paso previo al proceso algorítmico consiste esencialmente en convertir el grafo G en una red de transbordo y definir la tabla de transporte asociada y su estandarización. Para ello, de acuerdo al modelo matemático que describe el problema,

$$\begin{aligned} & Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot y_{ij} \\ & s.a. \sum_{k \in D(j)} y_{jk} - \sum_{i \in A(j)} y_{ij} = K_j \quad \forall j = 1,2,\dots,n \\ & L_{ij} \le y_{ij} \le U_{ij} \qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{1}$$

debe realizarse el cambio de variable:

$$x_{ij} = y_{ij} - L_{ij} \qquad (5)$$

que implica los siguientes cálculos:

$$u_{ij} = U_{ij} - L_{ij} \qquad (6)$$

$$k_j = K_j - \left(\sum_{k \in D(j)} L_{jk} - \sum_{i \in A(j)} L_{ij} \right) \qquad (7)$$

quedando el modelo como

$$\begin{aligned} & Min\ z' = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} \cdot x_{ij} \\ & s.a. \sum_{k \in D(j)} x - \sum_{i \in A(j)} x_{ij} = k_j \quad \forall j = 1,2,\dots,n \\ & 0 \le x_{ij} \le u_{ij} \qquad \forall i,j = 1,2,\dots,n \end{aligned} \qquad (8)$$

que será al que se le aplique el algoritmo.

El siguiente paso es transformar la red original en una red en la que ofertas y demandas coincidan, lo que se conoce como red equilibrada, para conseguir que la tabla de transporte asociada tenga solución.

Para ello, existen 3 métodos:

- Método 1: Equilibrado de ofertas y demandas con un nodo adicional y arcos adicionales.

  Si el problema no está equilibrado, es decir, si la suma de ofertas no coincide con la suma de demandas, debe añadirse un nodo adicional ficticio, que será un nodo de demanda en caso de haber exceso de oferta, o un nodo de oferta en caso de haber exceso de demanda. El nodo ficticio absorberá el exceso de oferta/demanda que tenga la red.

  Si el nodo ficticio es un nodo de oferta, éste estará unido únicamente a los nodos de demanda mediante arcos ficticios con capacidad infinita y coste absoluto nulo.

  Si el nodo ficticio es un nodo de demanda, éste estará unido únicamente a los nodos de oferta mediante arcos ficticios con capacidad infinita y coste absoluto nulo.

Tabla 1.- Equilibrado de redes.

EQUILIBRADO DE REDES			
Si A = B		Problema equilibrado	
Si A > B	Exceso de oferta	Nodo adicional de demanda	$k_{n+1} = -(A - B)$
Si A < B	Exceso de demanda	Nodo adicional de oferta	$k_{n+1} = A - B$

$$A = \sum_{j \in N} k_j \ \ (\forall k_j > 0) \qquad B = \sum_{j \in N} |k_j| \ \ (\forall k_j < 0)$$

(Oferta total) (Demanda total)

- Método 2: Equilibrado de ofertas y demandas con un nodo adicional y arcos adicionales.

  Nuevamente habrá que añadir un nodo ficticio adicional si el problema no está equilibrado de la misma manera que se indica en el método 1. La diferencia ahora radica en que este nodo ficticio adicional, independientemente de que sea de oferta o de demanda, se une a todos los nodos de la red mediante arcos ficticios con capacidad infinita y coste absoluto nulo.

- Método 3: Nodos de entrada y salida adicionales.

  En este método, se incluyen dos nodos ficticios adicionales:

  - Un nodo de salida que se une solamente a los nodos de oferta mediante arcos con capacidad igual al nivel de oferta de cada nodo y coste absoluto nulo.

  - Un nodo de entrada que se une solamente a los nodos de demanda mediante arcos con capacidad igual al nivel de demanda de cada nodo y coste absoluto nulo.

  Puesto que, por definición, el nodo ficticio de salida nunca demandará flujo, ni el nodo ficticio de entrada nunca ofertará flujo, pueden eliminarse de la tabla de transporte la fila asociada al nodo E y la columna asociada al nodo S, quedando una matriz cuadrada de orden n+1.

  En este supuesto, todos los nodos de la red original se convierten en nodos de transbordo, y la red modificada solo posee un único nodo de oferta (el de salida) y un único nodo de demanda (el de entrada).

La determinación de la matriz de costes de acuerdo a cada uno de estos métodos se detallará con ejemplos cuando se expliquen las funciones de MATLAB que realizan dichos cálculos.

A continuación, se indica el código que constituye la función "estandarizar" implementada en MATLAB, dividido en fragmentos que se irán comentando sucesivamente

```
function [U,C,of,dem]=estandarizar(arcos,K,metodo)
```

La función “estandarizar” recibe como datos las variables “arcos” y “K” explicadas previamente, así como la variable “metodo”, que indica el método a emplear para crear la red de transbordo, y valdrá 1, 2 ó 3, según sea el método elegido.

El resultado será la matriz de costes de la red, calculada acorde al modelo seleccionado para el equilibrado de las ofertas y demandas, la matriz de capacidades, y los vectores de oferta y demanda asociados, respectivamente, a las filas y columnas de la tabla de transporte.

En primer lugar, se guarda en distintas variables la información de las columnas de la matriz “arcos”, así como calcular el número de nodos y el número de arcos. Esto permitirá, un poco más adelante, el cálculo de la matriz de costes y de la matriz de capacidades, necesarias para aplicar el algoritmo.

```
nodosi=arcos(:,1);
nodosj=arcos(:,2);
Lb=arcos(:,3);
Ub=arcos(:,4);
Cij=arcos(:,5);
n=length(K);
narc=length(nodosi);
```

Este siguiente tramo de código permite obtener un grafo de la red original, en la que en cada arco se indica su coste absoluto.

```
G=digraph(nodosi,nodosj,Cij);
red=plot(G,'EdgeLabel',G.Edges.Weight);
red.NodeColor='r';
red.MarkerSize=5;
```

Ejemplo.- *Para la red de la Figura 1, y su correspondiente matriz “arcos”, la función “estandarizar” devuelve el siguiente grafo orientado:*

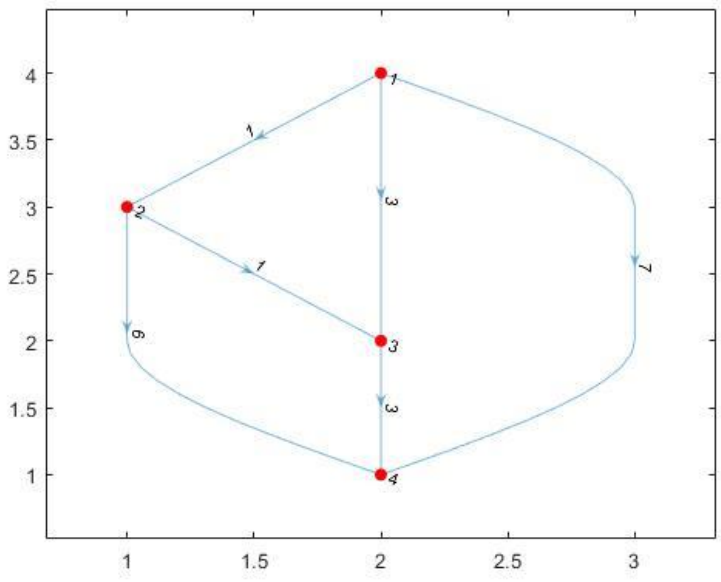


Figura 2.- Grafo de la red de la Figura 1, obtenido con la función “estandarizar”.

En cada arco, se indica su coste absoluto. Este grafo, permite visualizar en MATLAB la red a resolver, además de identificar posibles errores a la hora de introducir la matriz "arcos" (ya sea algún error numérico a la hora de introducir los datos, algún arco que no se haya incluido, etc.).

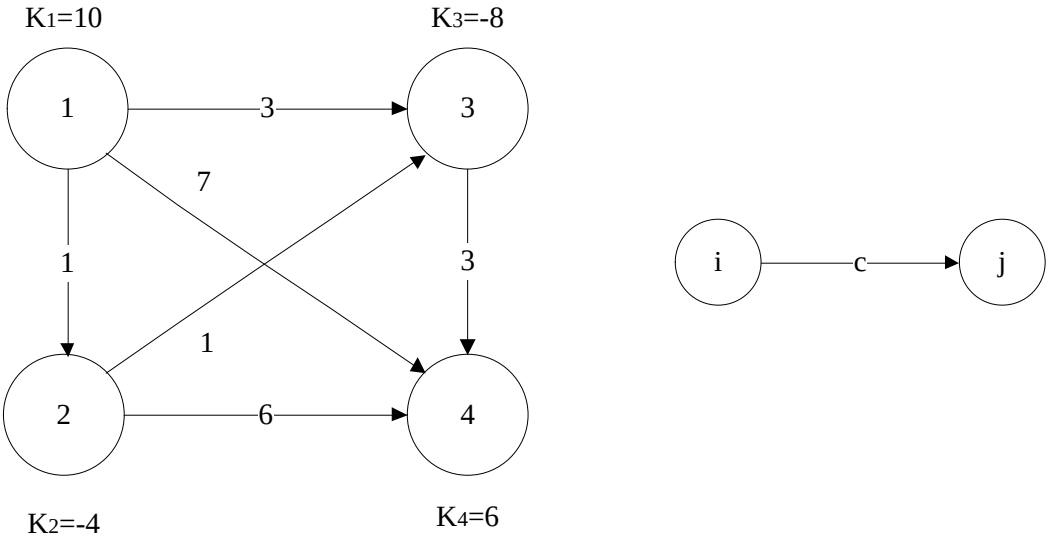


Figura 3.- Grafo realista equivalente al grafo de la Figura 2.

Las siguientes líneas corresponden al cálculo del vector k_j de acuerdo a la ecuación (7).

```
k=zeros(1,n);
for ii=1:n
    Liij=sum(Lb(nodosi==ii));
    Ljji=sum(Lb(nodosj==ii));
    k(1,ii)=K(ii)-(Liij-Ljji);
end
```

Ejemplo.- *A partir del vector "K" correspondiente a la red de la Figura 1*

$$K = [10 \quad -4 \quad -8 \quad 6]$$

Se obtiene el vector "k"

$$k = [4 \quad -4 \quad -8 \quad 12]$$

Para poder aplicar el algoritmo PDTCI, se necesita definir para cada fila de la tabla de transporte asociada, un nivel de oferta, y para cada columna, un nivel de demanda, que deberán satisfacerse en el proceso algorítmico. Esto implica que, a partir del vector k_j, que indica ofertas y demandas de todos los nodos, deben calcularse otros dos vectores que indiquen, por separado la oferta y demanda de cada nodo.

Por otra parte, la matriz de costes (sin considerar nodos adicionales) será una matriz cuadrada de orden n formada por los siguientes elementos:

- Costes c_{ij} para las celdas que representan al arco (i,j).
- Costes 0 para las celdas de la diagonal.
- Costes M (en MATLAB, infinito) para el resto de celdas.

Las líneas de código que se muestran a continuación crean esta matriz inicial de los costes (y la guarda en la variable "costes"). Para ello, se parte de una matriz $n\ x\ n$ con todos sus elementos igual a infinito, que luego sustituye los elementos de la diagonal por ceros, y los elementos (i,j) por los valores c_{ij}.

```
costes=inf*ones(n,n);
for ii=1:n
    for jj=1:n
        if ii==jj
            costes(ii,jj)=0;
        end
    end
end
for ii=1:narc
    costes(nodosi(ii),nodosj(ii))=Cij(ii);
end
```

El último paso para completar el proceso de estandarización, es aplicar el método seleccionado (1, 2 ó 3) para añadir los nodos ficticios, si procede, y calcular finalmente la matriz de costes, la matriz de capacidades superiores modificadas, y los vectores de oferta y demanda.

En este sentido, la función "estandarizar" toma 3 posibles caminos para el cálculo en función del valor de la variable "metodo". Los cálculos asociados a cada método se indican en el correspondiente apartado.

Ejemplo.- *Para la red de la Figura 1, la matriz inicial de costes, de orden n x n, sería:*

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 \\ M & 0 & 1 & 6 \\ M & M & 0 & 3 \\ M & M & M & 0 \end{bmatrix}$$

Esta matriz es la que se obtendría al ejecutar las líneas de código anteriores. Sobre esta matriz se harán los cálculos pertinentes, según se opte por el método 1, 2 ó 3, para obtener definitivamente la matriz de costes con la que se aplicará el algoritmo PDTCI.

2.2.1 Cálculo de ofertas y demandas.

El cálculo del vector de ofertas y demandas se hará conjuntamente con el cálculo de la matriz de costes y de la de capacidades. Para este cálculo, se ha implementado otra función en MATLAB, denominada "ofdem" (en referencia a los términos oferta y demanda)

```
function [of,dem,s]=ofdem(k)
```

Esta función recibe el vector "k" calculado anteriormente, y lo separa inicialmente en otros dos vectores del mismo tamaño:

- Vector "of", de ofertas, en el que se guardan los valores k_j positivos
- Vector "dem", de demandas, en el que se guardan, en valor absoluto, los valores k_j negativos.

Además, la función guarda en la variable "s", la suma de todas las k_j, de manera que el valor de "s" indicará el desequilibrio en oferta o demanda de la red.

```
s=sum(k); n=length(k);
of=zeros(1,n);
dem=zeros(1,n);
for ii=1:n
    if k(ii)>0
        of(ii)=k(ii);
    else
        dem(ii)=abs(k(ii));
    end
end
```

Por último, la misma función realiza el equilibrado de la red dependiendo del valor de "s", de acuerdo a las condiciones de la Tabla 1

- Si $s = 0$, el problema está equilibrado, y no es necesario añadir nodos adicionales.
- Si $s > 0$, entonces la oferta total es mayor que la demanda total, es decir, existe exceso de oferta. Por lo tanto, debe añadirse un nodo adicional de demanda cuya demanda sea "s" y cuya oferta sea nula.
- Si $s < 0$, entonces la oferta total es menor que la demanda total, es decir, existe exceso de demanda. Por lo tanto, debe añadirse un nodo adicional de oferta cuya oferta sea el valor absoluto de "s" y cuya demanda sea nula.

Esto se consigue con las siguientes líneas de código:

```
if s>0
    of=[of 0];
    dem=[dem s];
end
if s<0
    of=[of abs(s)];
    dem=[dem 0];
end
```

Ejemplo.- *Siguiendo con la red de la Figura 1, en la que*

$$k = [4 \quad -4 \quad -8 \quad 12]$$

Según el código de la función "ofdem", en primer lugar, los vectores "of" y "dem" serán

$$of = [4 \quad 0 \quad 0 \quad 12]$$
$$dem = [0 \quad 4 \quad 8 \quad 0]$$

Además, como $s = 4 + (-4) + (-8) + 12 = 4$, *hay exceso de oferta y debe añadirse un nodo ficticio de demanda igual a 4. Finalmente, el resultado de aplicar en Matlab*

```
k=[4 -4 -8 12];
[of,dem,s]=ofdem(k)
```

es

```
of =

     4     0     0    12     0


dem =

     0     4     8     0     4


s =

     4
```

2.2.2 Método 1. Equilibrado de la red.

La primera posibilidad para el equilibrado de la red es, como se ha indicado anteriormente, añadir un nodo adicional que absorba el exceso de oferta/demanda, si lo hubiere y unirlo solo a los arcos de demanda/oferta. Esto implica a las siguientes líneas de código dentro de la función "estandarizar".

```
if metodo==1
    [of,dem,s]=ofdem(k);
    U=full(sparse(nodosi,nodosj,Ub-Lb,length(of),length(dem)));
    U(U==0)=inf;
    of=of+sum(of);
    dem=dem+sum(dem);
    C=metodo1(k,costes,s);
end
```

En resumidas cuentas, se emplea la función "ofdem" para calcular los vectores de oferta y demanda, que, además, se orlarán.

Respecto a la matriz de capacidades superiores, tendrá las siguientes características:

- Si las ofertas y demandas están equilibradas, la matriz será cuadrada de orden *n*. En caso de desequilibrio, debe añadirse un nodo adicional, quedando una matriz cuadrada de orden *(n+1)*.
- Contendrá elementos iguales a $u_{ij} = U_{ij} - L_{ij}$ en las celdas que representan a los arcos (i,j).
- Contendrá elementos iguales a M (infinito), en el resto de celdas.

Esta matriz se obtiene al ejecutar las líneas de código anteriores.

Respecto a la matriz de costes, cabe recordar que la función "estandarizar" calcula inicialmente la matriz de costes de orden *n* (referente a los *n* nodos de la red original). Si el problema está desequilibrado hay que añadir una fila y una columna a dicha matriz, con los nuevos costes asociados a los arcos que aparecen al añadir un nodo ficticio.

Para ello, se emplea la función "metodo1", cuyo funcionamiento y código es el siguiente.

```
function costes=metodo1(k,costes,s)
```

Si se ha añadido un nodo adicional de demanda, se debe añadir una fila de costes infinitos a la matriz de costes (pues el nodo de demanda no puede ser origen de ningún arco). Además, el coste asociado al elemento diagonal (celda [n+1,n+1]), es nulo.
De acuerdo al método descrito, el resto de elementos de la última columna serán ceros (si el nodo correspondiente es un nodo de oferta) o infinitos (si el nodo correspondiente es un nodo de demanda).

```
[n,~]=size(costes);
if s>0
    costes(n+1,:)=inf;
    costes(n+1,n+1)=0;
    for ii=1:n
        if k(ii)>0
            costes(ii,n+1)=0;
        else
            costes(ii,n+1)=inf;
        end
    end
end
```

Si se ha añadido un nodo adicional de oferta, se debe añadir una columna de costes infinitos a la matriz de costes (pues el nodo de oferta no puede ser destino de ningún arco). Además, el coste asociado al elemento diagonal (celda [n+1,n+1]), es nulo.
De acuerdo al método descrito, el resto de elementos de la última fila serán ceros (si el nodo correspondiente es un nodo de demanda) o infinitos (si el nodo correspondiente es un nodo de oferta).

```
if s<0
    costes(:,n+1)=inf;
    costes(n+1,n+1)=0;
    for ii=1:n
        if k(ii)<0
            costes(n+1,ii)=0;
        else
            costes(n+1,ii)=inf;
        end
    end
end
```

Ejemplo.- *Para la red de la Figura 1, que como se ha visto tiene un exceso de oferta de 4 unidades, la red equilibrada de acuerdo al método 1, expuesto en este apartado sería:*

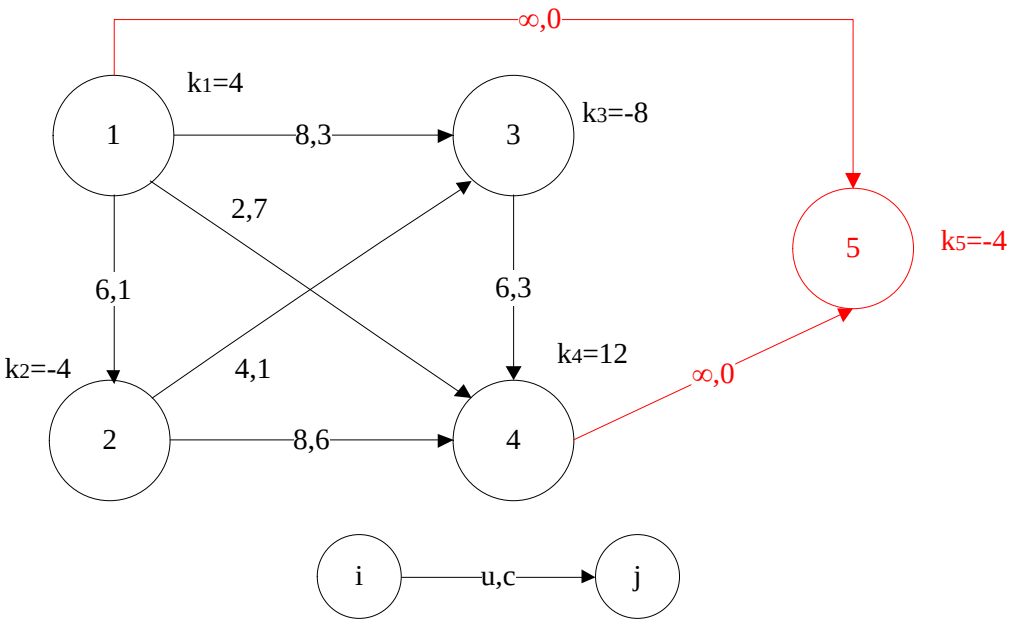


Figura 4.- Red de la Figura 1, equilibrada según el método 1.

Cuya tabla de transporte asociada es:

Tabla 2.- Tabla de transporte asociada a la Figura 4.

0	1	6	3	8	7	2	0	
M	0		1	4	6	8	M	
M	M		0		3	6	M	
M	M		M		0		0	
M	M		M		M		0	

cij	uij

Los cálculos que realiza la función "metodo1", a la matriz "costes" indicada anteriormente, se indican a continuación:

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 \\ M & 0 & 1 & 6 \\ M & M & 0 & 3 \\ M & M & M & 0 \end{bmatrix}$$

Para este caso, como s>0, se añade una fila adicional de costes infinitos

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 \\ M & 0 & 1 & 6 \\ M & M & 0 & 3 \\ M & M & M & 0 \\ M & M & M & M \end{bmatrix}$$

A continuación, se impone que el elemento (5,5) sea nulo, lo que en MATLAB implica crear una columna adicional de ceros.

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & 0 \\ M & 0 & 1 & 6 & 0 \\ M & M & 0 & 3 & 0 \\ M & M & M & 0 & 0 \\ M & M & M & M & 0 \end{bmatrix}$$

Por útlimo, para cada nodo i, se mantiene el coste nulo del arco (i,5) si es un nodo de oferta; o se modifica por un coste infinito si es un nodo de demanda. Como los nodos de demanda son 2 y 3, finalmente la matriz de costes es:

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & 0 \\ M & 0 & 1 & 6 & M \\ M & M & 0 & 3 & M \\ M & M & M & 0 & 0 \\ M & M & M & M & 0 \end{bmatrix}$$

El resultado que se obtiene en MATLAB al introducir las siguientes líneas de código:

```
arcos=[1 2 2 8 1; 1 3 4 12 3;1 4 0 2 7; 2 3 2 6 1;2 4 0 8 6;3 4 6 12 3];
K=[10 -4 -8 6];
[U,C,of,dem]=estandarizar(arcos,K,1)
```

es

```
U =

   Inf     6     8     2   Inf
   Inf   Inf     4     8   Inf
   Inf   Inf   Inf     6   Inf
   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf

C =

     0     1     3     7     0
   Inf     0     1     6   Inf
   Inf   Inf     0     3   Inf
   Inf   Inf   Inf     0     0
   Inf   Inf   Inf   Inf     0

of =

    20    16    16    28    16

dem =

    16    20    24    16    20
```

Recuérdese que los vectores "of" y "dem", calculados mediante la función "ofdem" se orlan para poder realizar la asignación de flujo en el proceso algorítmico.

2.2.3 Método 2. Equilibrado de la red.

El segundo método también implica introducir un nodo adicional de oferta/demanda si el problema está desequilibrado. Sin embargo, en este caso, dicho nodo adicional se une a todos los nodos de la red original mediante arcos ficticios, lo que simplifica el procedimiento de cálculo respecto al del método 1.

El código que emplea la función "estandarizar" para realizar las operaciones de acuerdo a este método es completamente análogo al método anterior.

```
if metodo==2
    [of,dem,s]=ofdem(k);
    U=full(sparse(nodosi,nodosj,Ub-Lb,length(of),length(dem)));
    U(U==0)=inf;
    of=of+sum(of);
    dem=dem+sum(dem);
    C=metodo2(costes,s);
end
```

La obtención de los vectores de oferta y demanda, así como de la matriz "U", son idénticas al caso anterior. La diferencia está en que, como el nodo ficticio se une a todos los nodos originales, la función "metodo2" no necesita al vector "k" como entrada.

```
function costes=metodo2(costes,s)
```

Si se añade un nodo adicional de demanda (exceso de oferta, s>0), a la matriz de costes se le añade una fila de costes infinitos y una columna de costes nulos. El elemento diagonal tiene coste nulo.

Si se añade un nodo adicional de oferta (exceso de demanda, s<0), a la matriz de costes se le añade una columna de costes infinitos y una fila de costes nulos. El elemento diagonal tiene coste nulo.

```
[n,~]=size(costes);
if s>0
    costes(n+1,:)=inf;
    costes(:,n+1)=0;
end
if s<0
    costes(:,n+1)=inf;
    costes(n+1,:)=0;
end
```

Ejemplo.- *Para la red de la Figura 1, que tiene un exceso de oferta de 4 unidades, la red equilibrada de acuerdo al método 2, expuesto en este apartado, sería:*

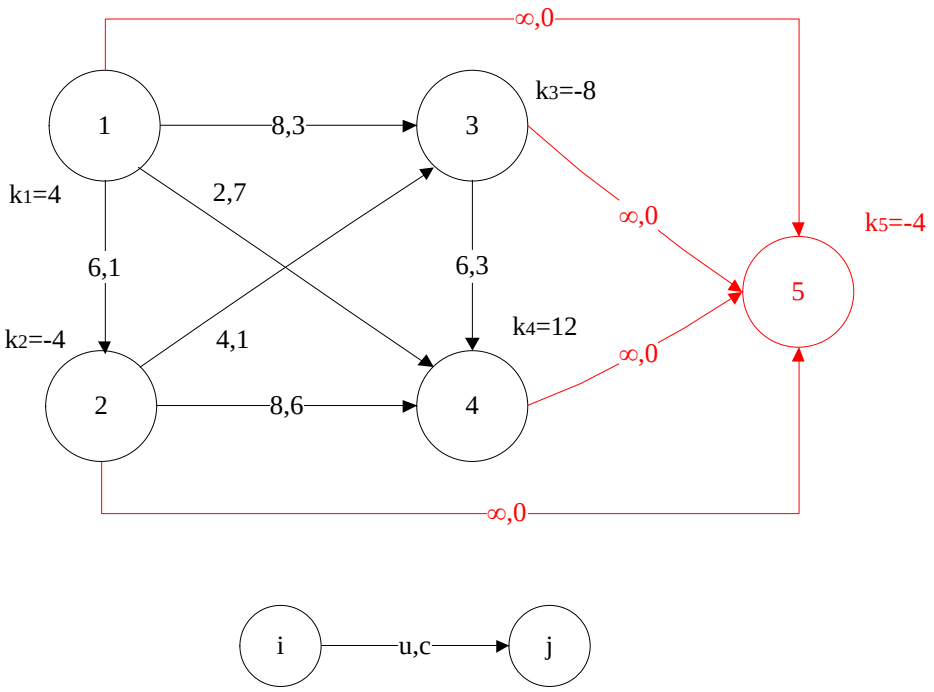


Figura 5.- Red de la Figura 1, equilibrada según el método 2.

Cuya tabla de transporte asociada es:

Tabla 3.- Tabla de transporte asociada a la red de la Figura 5.

0	1	6	3	8	7	2	0
M	0		1	4	6	8	0
M	M		0		3	6	0
M	M		M		0		0
M	M		M		M		0

cij	uij

Esta tabla se diferencia de la Tabla 2 en los costes asociados a los arcos (2,5) y (3,5), que corresponden a los arcos que unen los nodos de demanda con el nodo ficticio adicional. Según el método 1, estos arcos no se incluían al equilibrar la red (lo que implica coste infinito), y según el método 2, estos arcos sí aparecen en el grafo (con coste nulo).

Desde el punto de vista de los cálculos que realiza "metodo2" a la matriz "costes"

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 \\ M & 0 & 1 & 6 \\ M & M & 0 & 3 \\ M & M & M & 0 \end{bmatrix}$$

Como s>0, al ser el nodo ficticio adicional un nodo de demanda, primero se añade una fila de costes infinitos

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 \\ M & 0 & 1 & 3 \\ M & M & 0 & 3 \\ M & M & M & 0 \\ M & M & M & M \end{bmatrix}$$

y luego una columna de costes nulos, obteniendo la matriz de costes correspondiente al grafo estandarizado.

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & 0 \\ M & 0 & 1 & 3 & 0 \\ M & M & 0 & 3 & 0 \\ M & M & M & 0 & 0 \\ M & M & M & M & 0 \end{bmatrix}$$

El resultado que se obtiene en MATLAB al ejecutar la función "estandarizar" aplicando el método 2 es idéntico al obtenido al aplicar el método 1 para las variables "U", "of" y "dem".

```
arcos=[1 2 2 8 1; 1 3 4 12 3;1 4 0 2 7; 2 3 2 6 1;2 4 0 8 6;3 4 6 12 3];
K=[10 -4 -8 6];
[U,C,of,dem]=estandarizar(arcos,K,2)
```

La matriz "C" que se obtiene es:

```
C =

     0     1     3     7     0
   Inf     0     1     6     0
   Inf   Inf     0     3     0
   Inf   Inf   Inf     0     0
   Inf   Inf   Inf   Inf     0
```

pues el nodo adicional ficticio es un nodo de demanda.

En caso de haber sido un nodo de oferta, la última fila de la matriz de costes sería una fila de ceros y la última columna sería de infinitos. El elemento diagonal seguiría siendo nulo.

2.2.4 Método 3. Conversión en una red de transbordo.

La tercera opción para convertir el problema original en un problema estándar es transformar la red de manera que todos los nodos de la red se conviertan en nodos de transbordo. Esta misma estrategia es la que se sigue a la hora de aplicar el algoritmo Primal-Dual gráfico y consiste en añadir un nodo de salida S (que se unirá a todos los nodos de oferta, $k_j > 0$, con arcos de coste nulo y capacidad igual a k_j) y un nodo de entrada E (que se unirá a todos los nodos de demanda, $k_j < 0$, con arcos de coste nulo y capacidad igual a $|k_j|$) ficticios.

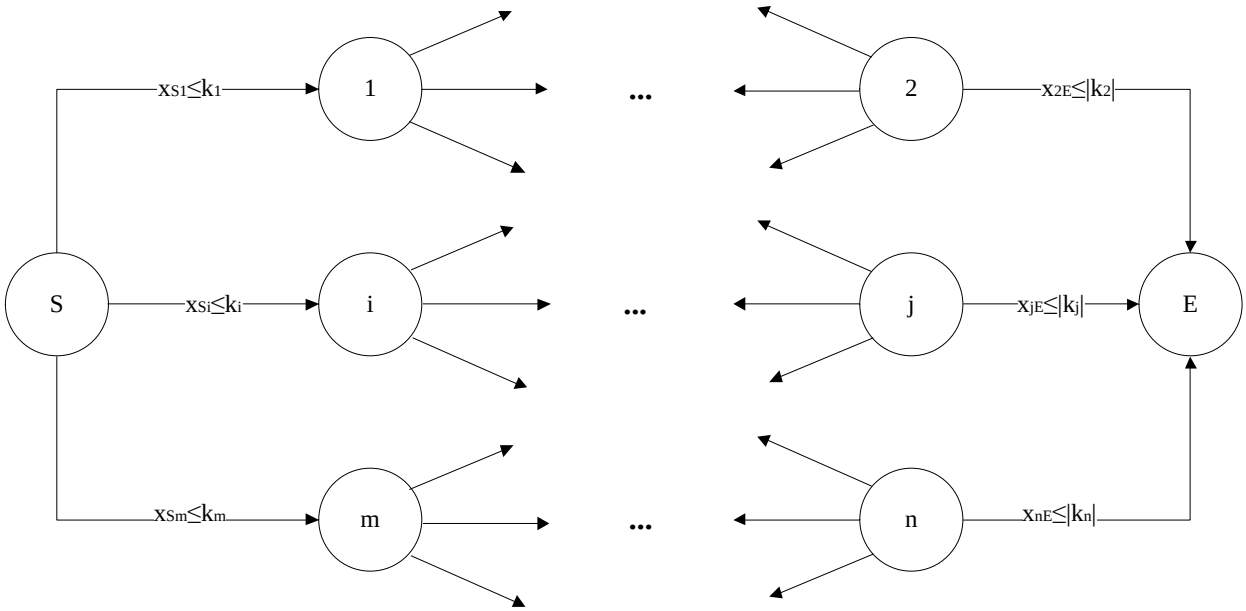


Figura 6.- Estructura de una red genérica, estandarizada según el método 3.

De esta manera, la tabla de transporte asociada al grafo de la red original tendría $(n+2)\ x\ (n+2)$ celdas. Sin embargo, la columna asociada al nodo de salida S puede eliminarse de la tabla, pues éste nunca será destino de ningún arco. El razonamiento es análogo para la fila asociada al nodo E. Así, puede simplificarse la tabla de transporte (y por tanto las matrices de costes y capacidades) hasta quedar la siguiente estructura:

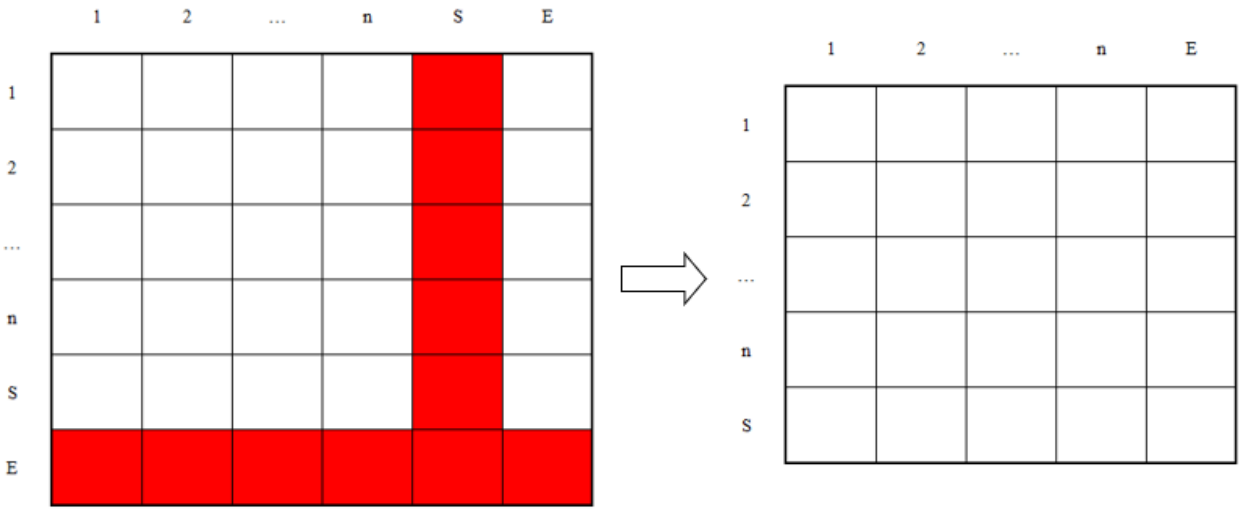


Figura 7.- Estructura de la tabla de transporte asociada a la red genérica de la Figura 6.

En resumen, la tabla de transporte siempre estará compuesta por $(n+1)\ x\ (n+1)$ celdas. Además, ahora en la red únicamente hay un nodo de oferta (S) y un nodo de demanda (E). Los vectores de oferta y demanda siempre tendrán (n+1) elementos, cuyo cálculo se explica más adelante en este apartado.

Para calcular las matrices de costes y capacidades, así como los vectores de oferta y demanda, la función "estandarizar" emplea las siguientes líneas de código.

```
if metodo==3
    U=full(sparse(nodosi,nodosj,Ub-Lb,n+1,n+1));
    U(U==0)=inf;
    [C,U,of,dem]=metodo3(k,costes,U);
end
```

Cabe recordar que la matriz "costes" es una matriz cuadrada de orden *n*, que incluye los costos absolutos de los arcos de la red original. La función "metodo3" se encargará de añadirle una fila y columna adicionales, tal como se ha comentado en párrafos anteriores, para calcular la matriz "C".

Además, se crea una matriz de capacidades "U", en la que se recogen capacidades iguales a u_{ij} para las celdas correspondientes a los arcos de la red original y costes infinitos para el resto de celdas. La misma función "metodo3" modificará el valor de las celdas correspondientes a los arcos ficticios y les asignará la capacidad correspondiente.

Ejemplo.- *Para la red de la Figura 1, las líneas de código anteriores crearán inicialmente la matriz "U" siguiente:*

$$U = \begin{bmatrix} M & 6 & 8 & 2 & M \\ M & M & 4 & 8 & M \\ M & M & M & 6 & M \\ M & M & M & M & M \\ M & M & M & M & M \end{bmatrix}$$

Sobre esta matriz se sustituirán los valores de la última fila y la última columna, según sean los arcos ficticios que se incluyan en la red, como se verá más adelante.

La función "metodo3" implementada, sigue la siguiente sintaxis:

```
function [costes,U,of,dem]=metodo3(k,costes,U)
```

A partir de las matrices "costes" y "U" calculadas en pasos anteriores, y en función de los valores del vector "k", esta función calcula las matrices de costes y capacidades que definirán la tabla de transporte del problema, y que serán las que se empleen en la aplicación del algoritmo PDTCI.

```
[n,~]=size(costes);
for ii=1:n
    if k(ii)>0
        costes(ii,n+1)=inf;
        U(n+1,ii)=k(ii);
    else
        if k(ii)<0
            costes(n+1,ii)=inf;
            U(ii,n+1)=abs(k(ii));
        else
            costes(ii,n+1)=inf;
            costes(n+1,ii)=inf;
        end
    end
end
costes(n+1,n+1)=inf;
```

Este script parte de las matrices "costes" y "U" anteriores, y realiza, para cada nodo *j*, las siguientes operaciones:

- Si el nodo *j* es de oferta ($k_j > 0$), impone coste infinito al arco (j,E), y una capacidad igual a k_j al arco (S,j).
- Si el nodo *j* es de demanda ($k_j < 0$), impone coste infinito al arco (S,j), y una capacidad igual a $|k_j|$ al arco (j,E)
- Si el nodo j ya era un nodo de transbordo ($k_j = 0$), impone costes infinitos a los arcos (S,j) y (j,E)

Por último, como el nodo S en ningún caso se unirá de manera directa al nodo E, se impone que el último elemento diagonal (que irá asociado al arco (S,E)) tenga coste infinito.

Recuérdese que la matriz "costes" es una matriz cuadrada de orden *n*. Al evaluar el nodo 1, el script impone un coste infinito a un determinado arco, que será el (1,E) si 1 es un nodo de oferta o el (S,1) si es un nodo de demanda. Al introducir este elemento en la fila o columna *n + 1*, MATLAB crea de manera automática una fila o columna completa en la que todos sus elementos serán nulos, salvo el que se corresponde al arco en cuestión. De ahí que, a la hora de programar la función, no se opte por imponer coste nulo a los arcos ficticios que aparecen en la red de transbordo, puesto que esos costes nulos ya aparecen cuando MATLAB crea la fila o columna adicional de manera automática.

En el siguiente ejemplo, se indican detalladamente los cálculos que la función "metodo3" realiza sobre las matrices "costes" y "U".

Ejemplo.- *Las variables de entrada para la función "metodo3", serán las matrices "costes" y "U" indicadas previamente, así como el vector "k", con el que se identificará cada nodo como nodo de oferta o demanda.*

$$k = [4 \quad -4 \quad -8 \quad 12] \qquad costes = \begin{bmatrix} 0 & 1 & 3 & 7 \\ M & 0 & 1 & 6 \\ M & M & 0 & 3 \\ M & M & M & 0 \end{bmatrix} \qquad U = \begin{bmatrix} M & 6 & 8 & 2 & M \\ M & M & 4 & 8 & M \\ M & M & M & 6 & M \\ M & M & M & M & M \\ M & M & M & M & M \end{bmatrix}$$

El bucle for *recorrerá los 4 nodos de la red original.*

El nodo 1, con $k_1 = 4$*, es un nodo de oferta. En este caso, se impone que el coste del arco (1,E) es infinito, es decir, se crea una columna adicional en la matriz "costes" en la que el elemento (1,E) vale M y el resto de elementos son nulos. La capacidad del arco (S,1) será 4.*

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & M \\ M & 0 & 1 & 6 & 0 \\ M & M & 0 & 3 & 0 \\ M & M & M & 0 & 0 \end{bmatrix} \qquad U = \begin{bmatrix} M & 6 & 8 & 2 & M \\ M & M & 4 & 8 & M \\ M & M & M & 6 & M \\ M & M & M & M & M \\ 4 & M & M & M & M \end{bmatrix}$$

El nodo 2, con $k_2 = -4$*, es un nodo de demanda. En este caso, se impone que el coste del arco (S,2) sea infinito, creándose una nueva fila en la matriz de costes. Véase también que la matriz costes anterior ya incluye el coste nulo del arco (2,E), que es el que aparece como arco ficticio al modificar la red original. Respecto a la matriz de capacidades, se sustituye la capacidad del arco (2,E) por su valor, que será 4.*

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & M \\ M & 0 & 1 & 6 & 0 \\ M & M & 0 & 3 & 0 \\ M & M & M & 0 & 0 \\ 0 & M & 0 & 0 & 0 \end{bmatrix} \qquad U = \begin{bmatrix} M & 6 & 8 & 2 & M \\ M & M & 4 & 8 & 4 \\ M & M & M & 6 & M \\ M & M & M & M & M \\ 4 & M & M & M & M \end{bmatrix}$$

El nodo 3, $k_3 = -8$*, también es un nodo de demanda, y los cambios en las matrices son análogos a los realizados para el nodo 2.*

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & M \\ M & 0 & 1 & 6 & 0 \\ M & M & 0 & 3 & 0 \\ M & M & M & 0 & 0 \\ 0 & M & M & 0 & 0 \end{bmatrix} \qquad U = \begin{bmatrix} M & 6 & 8 & 2 & M \\ M & M & 4 & 8 & 4 \\ M & M & M & 6 & 8 \\ M & M & M & M & M \\ 4 & M & M & M & M \end{bmatrix}$$

Para finalizar el bucle, el nodo 4, $k_4 = 12$, también es un nodo de oferta, y los cambios en las matrices son análogos a los realizados para el nodo 1.

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & M \\ M & 0 & 1 & 6 & 0 \\ M & M & 0 & 3 & 0 \\ M & M & M & 0 & M \\ 0 & M & M & 0 & 0 \end{bmatrix} \quad U = \begin{bmatrix} M & 6 & 8 & 2 & M \\ M & M & 4 & 8 & 4 \\ M & M & M & 6 & 8 \\ M & M & M & M & M \\ 4 & M & M & 12 & M \end{bmatrix}$$

Por último, se impone que el elemento diagonal tenga coste M

$$costes = \begin{bmatrix} 0 & 1 & 3 & 7 & M \\ M & 0 & 1 & 6 & 0 \\ M & M & 0 & 3 & 0 \\ M & M & M & 0 & M \\ 0 & M & M & 0 & M \end{bmatrix} \quad U = \begin{bmatrix} M & 6 & 8 & 2 & M \\ M & M & 4 & 8 & 4 \\ M & M & M & 6 & 8 \\ M & M & M & M & M \\ 4 & M & M & 12 & M \end{bmatrix}$$

Con estos cálculos, se han obtenido las matrices de costes y capacidades que permitirán aplicar el algoritmo PDTCI.

Además, la función "metodo3", también permite calcular los vectores de oferta y demanda a partir del vector "k". Sin embargo, el procedimiento para su obtención es ligeramente diferente al de los dos métodos anteriores.

El único nodo de oferta, S, tendrá que ser capaz de introducir en la red como máximo, la oferta total, es decir:

$$k_S = \sum_{j \in N} k_j \quad (\forall k_j > 0) \qquad (9)$$

El único nodo de demanda, E, tendrá que ser capaz de absorber de la red, como máximo, la demanda total:

$$k_E = -\sum_{j \in N} |k_j| \quad (\forall k_j < 0) \qquad (10)$$

Para que el problema pueda resolverse, debe estar equilibrado, es decir, $k_S = -k_E$. Sin embargo, si la oferta total y la demanda total no coinciden, el flujo máximo en la red está acotado por el valor mínimo entre k_S y $|k_E|$, que será el valor que se imponga como oferta del nodo S y como demanda del nodo E. Finalmente, estas ofertas y demandas se orlarán para toda la tabla, lo que permitirá su resolución.

Desde el punto de vista de la programación, estos cálculos se realizan aprovechando los resultados de la función "ofdem" explicada en el Apartado 2.2.1 de esta memoria. El código implementado dentro de la función "metodo3", que permite obtener los vectores de oferta y demanda se indica a continuación.

```
[kpos,kneg]=ofdem(k);
pos=sum(kpos(1:n));
neg=sum(kneg(1:n));
f=min(pos,neg);
of(n+1)=f;
dem(n+1)=f;
of=of+sum(of);
dem=dem+sum(dem);
```

Ejemplo.- *Para el cálculo de los vectores de oferta y demanda, se parte del vector "k"*

$$k = [4 \quad -4 \quad -8 \quad 12]$$

Se aprovecha la función "ofdem" para separar las componentes de estos vectores según sean positivas (ofertas) o negativas (demandas). Así, al calcular

```
k=[4 -4 -8 12];
[kpos,kneg]=ofdem(k)
```

Se obtienen los vectores:

$$kpos = [4 \quad 0 \quad 0 \quad 12 \quad 0] \qquad kneg = [0 \quad 4 \quad 8 \quad 0 \quad 4]$$

Recordar que la función "ofdem" realiza el equilibrado de las ofertas y demandas, según se tenga exceso de oferta o exceso de demanda. No obstante, aquí solo interesa haber separado ofertas y demandas en dos vectores independientes. Los primeros 4 elementos de "kpos" muestran las ofertas de todos los nodos originales, mientras que los primeros 4 elementos de "kneg" muestran las demandas de todos los nodos originales. El elemento 5 de cada vector hace referencia al exceso de oferta o demanda que, para estos cálculos, no tienen relevancia alguna.

Como se ha comentado anteriormente, el flujo máximo, f, en la red viene acotado por el mínimo entre la oferta total y la demanda total. La variable "pos", calcula la suma de los n primeros elementos de "kpos", es decir, la oferta total, y la variable "neg", calcula la suma de los n primeros elementos de "kneg", es decir, la demanda total.

$$pos = 4 + 0 + 0 + 12 = 16$$
$$neg = 0 + 4 + 8 + 0 = 12$$
$$f = min(pos, neg) = min(16{,}12) = 12$$

Este flujo máximo, f, es que el se asigna como oferta del nodo S y como demanda del nodo E. De esta manera, el problema está equilibrado.

$$of = [0 \quad 0 \quad 0 \quad 0 \quad 12] \qquad dem = [0 \quad 0 \quad 0 \quad 0 \quad 12]$$

Y orlando, se obtienen los vectores de oferta y demanda con los que se puede resolver el problema mediante la función "pdtci" que se verá en el siguiente Apartado de este Capítulo.

$$of = [12 \quad 12 \quad 12 \quad 12 \quad 24] \qquad dem = [12 \quad 12 \quad 12 \quad 12 \quad 24]$$

Para finalizar, se muestran, a modo de resumen, los resultados obtenidos al aplicar el método 3 a la red de transporte de la Figura 1.

Ejemplo.- *La red de la Figura 1, estandarizada de acuerdo al método 3, quedaría:*

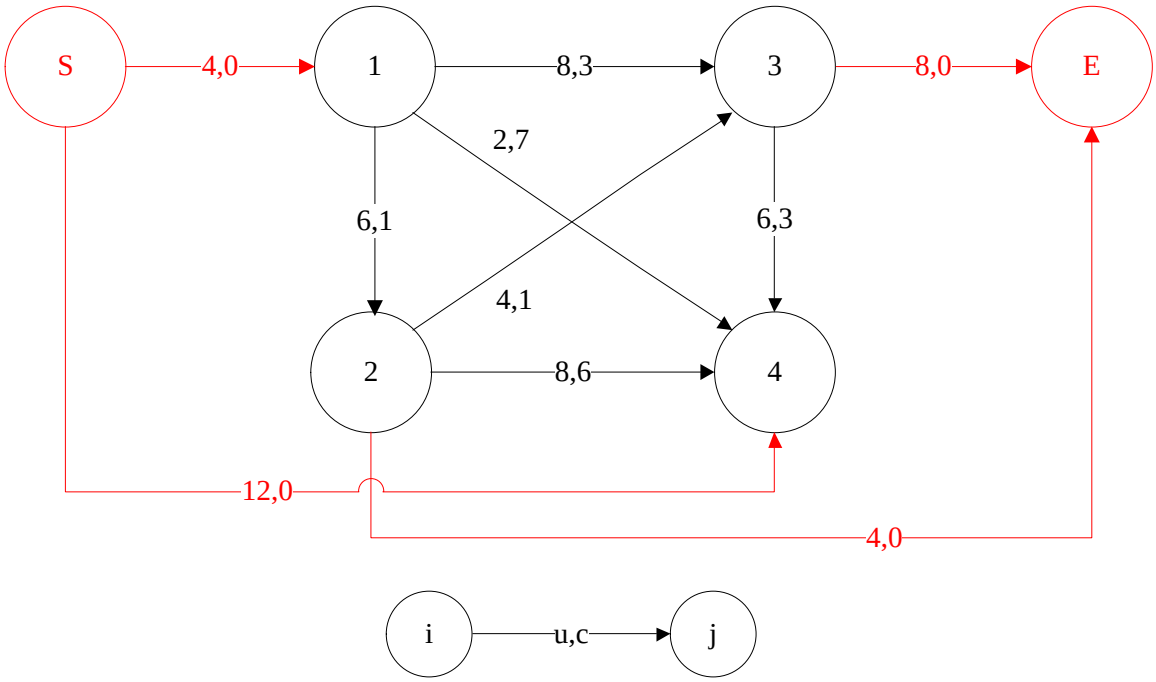


Figura 8.- Red de la Figura 1, estandarizada según el método 3.

En esta red, todos los nodos que pertenecen a la red original ahora son nodos de transbordo.

Su tabla de transporte asociada es:

Tabla 4.- Tabla de transporte asociada a la red de la Figura 8.

0		1	6	3	8	7	2	M	
M		0		1	4	6	8	0	4
M		M		0		3	6	0	8
M		M		M		0		M	
0	4	M		M		0	12	M	

cij	uij

El resultado obtenido en MATLAB al ejecutar la función de estandarización empleando el método 3

```
arcos=[1 2 2 8 1; 1 3 4 12 3;1 4 0 2 7; 2 3 2 6 1;2 4 0 8 6;3 4 6 12 3];
K=[10 -4 -8 6];
[U,C,of,dem]=estandarizar(arcos,K,3)
```

es:

```
U =

   Inf     6     8     2   Inf
   Inf   Inf     4     8     4
   Inf   Inf   Inf     6     8
   Inf   Inf   Inf   Inf   Inf
     4   Inf   Inf    12   Inf

C =

     0     1     3     7   Inf
   Inf     0     1     6     0
   Inf   Inf     0     3     0
   Inf   Inf   Inf     0   Inf
     0   Inf   Inf     0   Inf

of =

    12    12    12    12    24

dem =

    12    12    12    12    24
```

2.3. Algoritmo Primal-Dual con costes constantes.

Hasta ahora lo que se ha conseguido es adecuar los datos de la red original para convertirla en una red estándar, de manera que pueda resolverse aplicando un algoritmo de transporte. El siguiente paso, es aplicar el algoritmo Primal-Dual aplicado a la Tabla de transporte para Costos Independientes del flujo. El algoritmo consta de las etapas que se indicaron al comienzo de este capítulo, y que brevemente se recuerdan a continuación.

En cuanto a la programación, la función "pdtci" implementa distintas subrutinas con las que se realizan las distintas etapas del algoritmo. A partir de las matrices "C" y "U", y los vectores de oferta y demanda, calculados al estandarizar el grafo de la red original, las distintas subrutinas permiten obtener el flujo máximo a coste mínimo de la red estandarizada.

```
function [flujo,mr]=pdtci(C,of,dem,U)
[mr,~]=matred(C);
[m,n]=size(mr);
flujo=zeros(m,n);
while true
    [flujo,of,dem]=asignaflujo(mr,flujo,of,dem,U);
    if sum(of)==0, break; end
    [I,J,ent]=marcajeff(mr,flujo,of,dem,U);
    if ent~=0
        [flujo,of,dem]=aumentaflujo(flujo,of,dem,I,J,ent);
        if sum(of)==0, break; end
    else
        [mr,modmr]=modificaMR(mr,I,J,flujo,U);
        if modmr==0, break; end
    end
end
```

El código inicia los cálculos hallando la matriz reducida (función "matred") de la matriz de costes "C", que identificará la red básica sobre la que empezar el algoritmo. Además, se inicializa la matriz de flujos con todos los flujos a cero, que será la solución factible desde la que se comienza el proceso algorítmico.

El bucle while realiza los cálculos del algoritmo propiamente dichos. En primer lugar, se realiza una asignación directa de flujos (mediante la función "asignaflujo"), en caso de que, para una celda básica de la matriz de flujo exista oferta y demanda que pueda satisfacerse. En este punto, si se han saturado todas las ofertas y demandas, se ha alcanzado el óptimo y, por tanto, se finaliza el bucle while. En caso contrario, se prosigue el algoritmo.

A continuación, se realiza el proceso de marcaje del algoritmo Ford-Fulkerson (función "marcajeff"). El resultado del marcaje se recoge en las matrices "I" y "J", cuyo contenido y forma de calcularlas en MATLAB se describirán en el Apartado 2.3.3. Además, se identificará mediante la variable "ent" si se ha alcanzado un nodo de entrada o no. La función de marcaje asigna a "ent" el número del nodo de entrada que se ha alcanzado al marcar, o bien un valor de 0 si no se ha alcanzado ningún nodo de entrada.

Aquí el código tiene una bifurcación.

- Si $ent \neq 0$, entonces, mediante la función "aumentaflujo", puede aumentarse el flujo en la tabla. Además, se actualizan los vectores de oferta y demanda. Para ello, se emplea la información de las matrices "I" y "J". Una vez realizado el aumento de flujos, se vuelve a comprobar si se han satisfecho todas las ofertas y demandas, en cuyo caso se habría alcanzado el óptimo.

  Una vez aumentado los flujos, puede volver a realizarse el proceso de marcaje, en busca de una nueva ruta que vaya desde uno de los nodos de salida hacia uno de los nodos de entrada.

- Si $ent = 0$, entonces debe modificarse la matriz reducida (empleando la función "modificaMR"). Para ello, se parte del resultado del proceso de marcaje. Como se explicará en el Apartado 2.3.5 de esta Memoria, hay casos en los que la matriz reducida no puede modificarse. Esto se identifica en la función "modificaMR" con la variable de decisión "modmr", que tomará el valor 1 si se ha podido realizar la modificación de la matriz reducida, y 0 en caso contrario. En este último supuesto, se ha alcanzado el óptimo, pues el flujo en la red no puede aumentarse, aunque queden ofertas y demandas sin satisfacer.

  La modificación de la matriz reducida implica la aparición de nuevas celdas básicas sobre las que podrá asignarse flujo nuevamente y con ello continuar el algoritmo.

En resumidas cuentas, la siguiente tabla recoge una breve descripción de cada una de las funciones o subrutinas que forman parte de la función "pdtci" y que serán desarrolladas y analizadas de manera individual a lo largo de las distintas secciones de este Apartado.

Tabla 5.- Subrutinas de la función "pdtci".

FUNCIÓN "pdtci"	
matred	Determinación de los arcos básicos de la red estandarizada
asignaflujo	Asignación directa de flujos sobre la tabla de transporte
marcajeff	Marcaje de los nodos según el algoritmo de Ford-Fulkerson
aumentaflujo	Aumento de flujos en la red básica según el algoritmo de Ford-Fulkerson
modificaMR	Modificación de la matriz reducida

Para facilitar la comprensión del funcionamiento del código anterior, así como del proceso algorítmico que sigue MATLAB al aplicar la función "pdtci" para resolver el problema de flujo máximo a coste mínimo independiente del flujo, se muestra el siguiente diagrama de flujo.

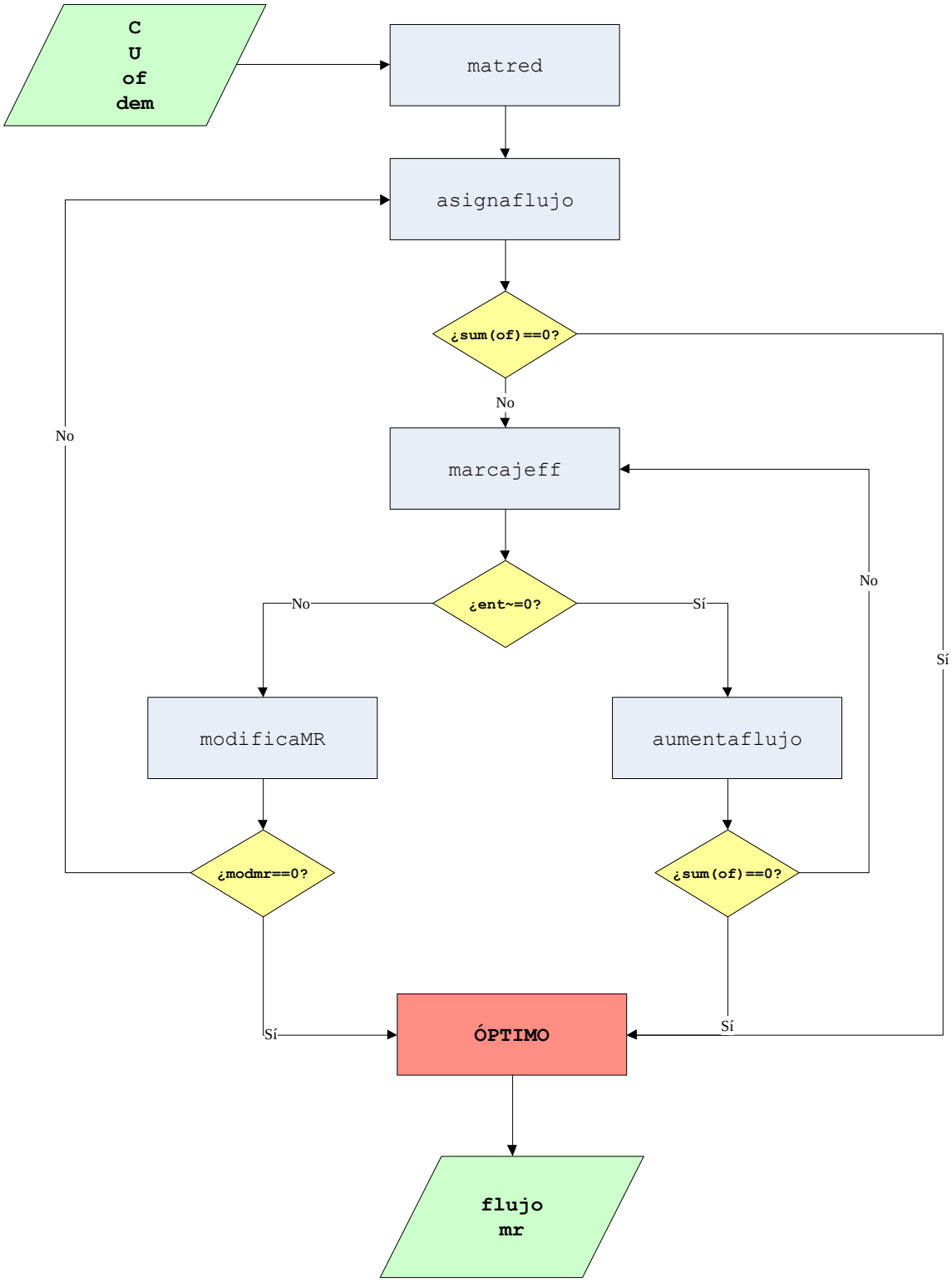


Figura 9.- Diagrama de flujo de la función "pdtci".

En dicho diagrama se indican: en color verde, los datos de entrada y salida de la función; en color azul, las distintas subrutinas que se aplican; y en color amarillo, las variables de decisión.

2.3.1 Cálculo de la matriz reducida.

Para el cálculo de la matriz reducida, se emplea la función "matred", cuyo código es:

```
function [C,B]=matred(A)
[m,n]=size(A);
for ii=1:m
    u=min(A(ii,:));
    B(ii,:)=A(ii,:)-u;
end
for jj=1:n
    v=min(B(:,jj));
    C(:,jj)=B(:,jj)-v;
end
```

Para una matriz "A" cualquiera, "matred" calcula la matriz reducida primero por filas y luego por columnas y la guarda en la variable "C". La función permite la posibilidad de obtener también la matriz intermedia "B".

Ejemplo.- *Sea la matriz "A" siguiente:*

$$A = \begin{bmatrix} 4 & 5 & 8 & 9 \\ -1 & 2 & 3 & 4 \\ 6 & 5 & 4 & 3 \\ 2 & 6 & -3 & 3 \end{bmatrix}$$

En primer lugar, se obtiene la matriz "B" restando a cada fila de la matriz "A", el valor mínimo de la misma.

$$B = \begin{bmatrix} 0 & 1 & 4 & 5 \\ 0 & 3 & 4 & 5 \\ 3 & 2 & 1 & 0 \\ 5 & 9 & 0 & 6 \end{bmatrix}$$

En segundo lugar, se obtiene la matriz "C" restando a cada columna de la matriz "B", el valor mínimo de la misma.

$$C = \begin{bmatrix} 0 & 0 & 4 & 5 \\ 0 & 2 & 4 & 5 \\ 3 & 1 & 1 & 0 \\ 5 & 8 & 0 & 6 \end{bmatrix}$$

Al ejecutar en MATLAB:

```
A=[4 5 8 9;-1 2 3 4;6 5 4 3;2 6 -3 3];
[C,B]=matred(A)
```

Se obtiene

```
C =                         B =

     0     0     4     5         0     1     4     5
     0     2     4     5         0     3     4     5
     3     1     1     0         3     2     1     0
     5     8     0     6         5     9     0     6
```

2.3.2 Asignación directa de flujos.

La asignación directa de flujos supone el primer paso del proceso algorítmico y consiste, esencialmente, en comprobar si alguna de las celdas básicas corresponde a un arco cuyo origen sea uno de los nodos de oferta y cuyo destino sea uno de los nodos de demanda. De esta manera, ese mismo arco representa una ruta que une un nodo de oferta con un nodo de demanda, a través del cual, puede aumentarse flujo en la red.

A efectos prácticos, esta etapa podría suprimirse del algoritmo, pues mediante el algoritmo de Ford-Fulkerson también podrían encontrarse estas rutas y aumentar el flujo. En este Trabajo, la idea es implementar el algoritmo de manera que represente lo más fielmente posible la resolución a mano. Es por ello, que se mantiene en el algoritmo como paso inicial a cada iteración de Ford-Fulkerson.

En el planteamiento de la resolución del problema a mano, esta etapa consiste básicamente en una inspección visual de la tabla de transporte, para asignar los flujos que sean posibles. En cuanto a la programación en MATLAB de esta asignación, se incluye en la función "asignaflujo", cuyo código es el siguiente:

```
function [flujo,of,dem]=asignaflujo(mr,flujo,of,dem,U)
[m,n]=size(flujo);
for ii=1:m
    for jj=1:n
        if mr(ii,jj)==0
            inc=min(of(ii),dem(jj));
            inc=min(inc,U(ii,jj)-flujo(ii,jj));
            flujo(ii,jj)=flujo(ii,jj)+inc;
            of(ii)=of(ii)-inc;
            dem(jj)=dem(jj)-inc;
        end
    end
end
```

La función recibe como argumentos de entrada la matriz reducida de los costes ("mr"), la matriz de capacidades superiores ("U") y los vectores de oferta y demanda.

El programa realiza un barrido por todas las celdas de la tabla, y para cada celda, se comprueba si es básica a través de la matriz reducida. En caso afirmativo, se calcula el incremento de flujo admisible (Δ; en MATLAB, "inc") para dicha celda como el valor mínimo entre su oferta, su demanda, y el flujo que puede aumentarse hasta saturarse dicha celda:

$$\Delta = \min\left(\mathrm{of_i}, \mathrm{dem}_j, u_{ij} - x_{ij}\right) \quad (11)$$

Por último, se aumenta el flujo de dicha celda en Δ unidades, y se disminuyen, tanto la oferta de su fila como la demanda de su columna, en Δ unidades.

Cabe destacar que, como puede verse en la llamada a la función de asignación, ésta lo que hará es actualizar los valores de la matriz de flujos y los vectores de oferta y demanda, eliminando de la memoria del programa los valores previos a cada cálculo. Esta misma filosofía se sigue en todas las subrutinas incluidas en "pdtci", calculando en cada paso los nuevos valores del flujo en la tabla, ofertas y demandas, en sustitución de los anteriores.

Ejemplo.- *Tomando la red inicial que se viene resolviendo a lo largo del presente Apartado (Figura 1), estandarizada según el método 1 (ver Figura 4 y Tabla 2), su matriz de costes es la siguiente:*

$$C = \begin{bmatrix} 0 & 1 & 3 & 7 & 0 \\ M & 0 & 1 & 6 & M \\ M & M & 0 & 3 & M \\ M & M & M & 0 & 0 \\ M & M & M & M & 0 \end{bmatrix}$$

Para obtener las celdas básicas, se calcula la matriz reducida, con la función "matred"

```
[mr,~]=matred(C);
```

$$mr = \begin{bmatrix} 0 & 1 & 3 & 7 & 0 \\ M & 0 & 1 & 6 & M \\ M & M & 0 & 3 & M \\ M & M & M & 0 & 0 \\ M & M & M & M & 0 \end{bmatrix}$$

Como puede observarse, la matriz "C" ya tenía la forma de una matriz reducida, por lo que al aplicarle la función "matred", sus valores no se han visto modificados. Esto es habitual en redes sin costos negativos. La tabla de transporte con la que se comienza la primera iteración del algoritmo, quedaría como:

Tabla 6.- Primera iteración "pdtci" para la red de la Figura 4. Celdas básicas.

	16	20	24	16	20	dem_j
20	0	1 | 6	3 | 8	7 | 2	0	of_i
16	M	0	1 | 4	6 | 8	M	c_{ij} | u_{ij} x_{ij}
16	M	M	0	3 | 6	M	
28	M	M	M	0	0	
16	M	M	M	M	0	

Mediante una simple inspección visual de la tabla, es fácil darse cuenta de que, por ejemplo, puede asignarse un flujo de 16 unidades en la celda (1,1). Ese flujo corresponde al mínimo entre la oferta y demanda disponibles. También debe tenerse en cuenta el flujo admisible por cada arco, en caso de que estén acotados superiormente (en este caso no aplica, pues la celda (1,1) no tiene límite superior). Al asignar dicho flujo a la celda (1,1), la demanda en la columna 1 de la tabla se satura y la oferta en la fila 1 disminuye a 4 unidades.

Igualmente puede procederse para el resto de arcos básicos (indicados con un cuadrado rojo en la tabla anterior), obteniéndose por asignación directa los siguientes flujos:

Tabla 7.- Primera iteración "pdtci" para la red de la Figura 4. Asignación directa.

	4	8			
	0 **16**	1 6	3 8	7 2	0 **4**
	M	0 **16**	1 4	6 8	M
	M	M	0 **16**	3 6	M
	M	M	M	0 **16**	0 **12**
12	M	M	M	M	0 **4**

Sobre esta tabla se realizará el proceso de marcaje del algoritmo Ford-Fulkerson.

En MATLAB, al ejecutar:

```
[mr,~]=matred(C)
[flujo,of,dem]=asignaflujo(mr,flujo,of,dem,U)
```

Se obtiene:

```
flujo =

    16     0     0     0     4
     0    16     0     0     0
     0     0    16     0     0
     0     0     0    16    12
     0     0     0     0     4

of =

     0     0     0     0    12

dem =

     0     4     8     0     0
```

2.3.3 Algoritmo de Ford-Fulkerson. Proceso de marcaje.

Una vez asignado flujos, el siguiente paso es aplicar el algoritmo de Ford-Fulkerson. Dicho algoritmo puede dividirse en dos etapas bien diferenciadas. En primer lugar, el algoritmo realiza una serie de asignaciones a cada nodo de la red, con el objetivo de buscar una ruta por la que llevar flujo desde un nodo de oferta hacia un nodo de demanda. Es lo que aquí se ha denominado, marcaje de la red. En segundo lugar, si se ha encontrado una ruta, se procede al aumento de flujo en los arcos de dicha ruta.

El procedimiento para realizar el marcaje o exploración de los nodos de la red, en una tabla de transporte, aparece recogido en la bibliografía sobre el algoritmo Primal-Dual, y se indica a continuación mediante un ejemplo numérico.

Ejemplo.- *A partir de la Tabla 7, se realiza el proceso de marcaje según el algoritmo de Ford-Fulkerson.*

En primer lugar, las filas asociadas a los nodos de oferta se marcan con (-,of$_i$). En este caso, el único nodo de oferta es el nodo 5:

Tabla 8.- Ejemplo de marcaje (Ford-Fulkerson). Paso 1.

		4	8		
	0 **16**	1 6	3 8	7 2	0 **4**
	M	0 **16**	1 4	6 8	M
	M	M	0 **16**	3 6	M
	M	M	M	0 **16**	0 **12**
12	M	M	M	M	0 **4** (-,12)

A continuación, se realiza el llamado marcaje hacia adelante. Para ello, a partir de la fila 5, puede aumentarse flujo en la columna 5 (la única celda básica disponible en esa fila). Por tanto, se marcaría la columna 5 con (5,12), puesto que desde la fila 5 puede aumentarse el flujo en 12 unidades. El nodo 5 se marcaría como explorado, con un asterisco.

Tabla 9.- Ejemplo de marcaje (Ford-Fulkerson). Paso 2.

	4		8		
	0 **16**	1 6	3 8	7 2	0 **4**
	M	0 **16**	1 4	6 8	M
	M	M	0 **16**	3 6	M
	M	M	M	0 **16**	0 **12**
12	M	M	M	M	0 **4** (-,12)*
					(5,12)

A continuación, se realiza el marcaje hacia detrás. Como la única columna con etiqueta es la 5, se realiza el marcaje desde ella. Las celdas básicas disponibles son las correspondientes a las filas 1 y 4, que se marcarán con la etiqueta (5,·), indicando el flujo que puede disminuirse. La columna 5 quedaría explorada.

Tabla 10.- Ejemplo de marcaje (Ford-Fulkerson). Paso 3.

	4		8		
	0 **16**	1 6	3 8	7 2	0 **4** (5,4)
	M	0 **16**	1 4	6 8	M
	M	M	0 **16**	3 6	M
	M	M	M	0 **16**	0 **12** (5,12)
12	M	M	M	M	0 **4** (-,12)*
					(5,12)*

En este caso, la fila 1 se ha marcado con (5,4) porque desde la columna 5 solo puede disminuirse un flujo de 4 unidades. El proceso de marcaje continúa realizando sucesivamente el marcaje hacia adelante y hacia detrás hasta que se alcance un nodo de demanda, hasta que se hayan explorado todos los nodos de la red o bien, hasta que no se puedan marcar más filas y columnas. El marcaje proporciona finalmente el siguiente resultado:

Tabla 11.- Ejemplo de marcaje (Ford-Fulkerson). Paso final.

	4		8		
	0 **16**	1 6	3 8	7 2	0 **4**	(5,4)*
	M	0 **16**	1 4	6 8	M	
	M	M	0 **16**	3 6	M	
	M	M	M	0 **16**	0 **12**	(5,12)*
12	M	M	M	M	0 **4**	(-,12)*
	(1,4)*			(4,12)*	(5,12)*	

En este ejemplo, tras el marcaje no se ha encontrado una ruta por la que aumentar el flujo, pues no se ha asignado una etiqueta a ninguno de los nodos de demanda. El siguiente paso del algoritmo sería modificar la matriz reducida de los costes para incluir nuevas celdas básicas.

Como puede verse en la Tabla 11, el resultado del proceso de marcaje no es más que un conjunto de etiquetas en ciertas filas y columnas de la tabla de transporte.

En la información que contienen las etiquetas correspondientes a las filas pueden diferenciarse tres elementos:

- El nodo desde el que se marca dicha fila en el marcaje hacia detrás, o bien un guión (“-“), en caso de que dicha fila se marque por corresponder a un nodo de oferta.
- El flujo que puede disminuirse.
- Un asterisco, en caso de que la fila esté explorada.

La estrategia para representar estas etiquetas en MATLAB, será agrupar los elementos anteriores en una matriz “I”, construida como la unión de tres vectores columna, que se denominarán:

- Vector “j”: En la primera columna de “I”, contiene para cada fila, el nodo desde el que se produce el marcaje. Para el nodo de oferta se le asigna un valor 0. Dicho valor corresponderá a la columna de dicha fila sobre la que puede disminuirse el flujo.
- Vector “deltai”: En la segunda columna de “I”, contiene los distintos valores de los flujos para cada fila.
- Vector “marcai”: En la tercera columna de “I”, contiene un valor 1 ó 0 para cada fila, según el nodo esté explorado o no.

Análogamente, en la información que contienen las etiquetas correspondientes a las columnas, pueden diferenciarse tres elementos:

- El nodo desde el que se marca dicha columna en el marcaje hacia adelante.
- El flujo que puede aumentarse.
- Un asterisco, en caso de que el nodo esté explorado.

De igual manera, la estrategia a seguir en MATLAB, será agrupar estos elementos en una matriz "J", construida como la unión de tres vectores fila, que se denominarán:

- Vector "i": En la primera fila de "J", contiene, para cada columna, el nodo desde los que se produce el marcaje. Dicho valor corresponderá a la fila de dicha columna sobre la que puede aumentarse el flujo.
- Vector "deltaj": En la segunda fila de "J", contiene los distintos valores de los flujos para cada columna.
- Vector "marcaj": En la tercera fila de "J", contiene un valor 1 ó 0, según la columna esté explorada o no.

Para una red estándar de *n* nodos, "I" será una matriz *n x 3*, y "J" será una matriz *3 x n*, con la siguiente estructura general:

$$\mathrm{I} = \begin{bmatrix} columna & flujo & * \\ j & deltai & marcai \\ \vdots & \vdots & \vdots \end{bmatrix} \qquad (12)$$

$$\mathrm{J} = \begin{bmatrix} fila & i & \dots \\ flujo & deltaj & \dots \\ * & marcaj & \dots \end{bmatrix} \qquad (13)$$

Ejemplo.- *Para el marcaje indicado en la Tabla 11, las matrices "I" (12) y "J" (13) que pueden definirse serán:*

$$I = \begin{bmatrix} 5 & 4 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 5 & 12 & 1 \\ 0 & 12 & 1 \end{bmatrix} \qquad J = \begin{bmatrix} 1 & 0 & 0 & 4 & 5 \\ 4 & 0 & 0 & 12 & 12 \\ 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$

En definitiva y en lo que se refiere a la implementación del algoritmo PDTCI en MATLAB, la función "marcajeff" es la que permite calcular los 6 vectores indicados anteriormente, y a partir de ellos construir las matrices "I" y "J" que reflejan el resultado obtenido en el proceso de marcaje. Además, calculará la variable "ent" cuyo valor coincidirá con el nodo de demanda alcanzado en el marcaje, ó 0 si no se ha alcanzado ninguno.

```
function [I,J,ent]=marcajeff(mr,flujo,of,dem,U)
```

Para ello, esta subrutina recibe como entradas las matrices de flujo y capacidades, la matriz reducida de los costes (o las siguientes modificadas) y los vectores de oferta y demanda.

En primer lugar, la función transforma la matriz reducida en una matriz lógica (denominada "ppr") en la que las celdas correspondientes a celdas básicas toman un valor de 1 y el resto, valor 0. Además, inicializa los 6 vectores indicados anteriormente con valores nulos.

```
ppr=~logical(mr);
m=length(of);
n=length(dem);
j=zeros(m,1);
deltai=zeros(m,1);
marcai=zeros(m,1);
i=zeros(1,n);
deltaj=zeros(1,n);
marcaj=zeros(1,n);
```

A continuación, se realizan de manera iterativa los marcajes hacia adelante y hacia detrás mediante un bucle while. Para detener el bucle se emplea una variable bandera ("flag"), que en cada iteración se inicia en un valor igual a 0 y a la que se le suma 1 por cada fila o columna que se marque. El bucle finalizará cuando el valor de esta variable bandera se mantenga en 0 tras un marcaje hacia delante y hacia atrás. Esto implica que ya no pueden marcarse más filas y columnas y, por tanto, el proceso de marcaje ha finalizado.

Antes del primer marcaje hacia adelante, se asignan las ofertas disponibles al vector "deltai"

```
deltai=of';
flag=1;
while flag~=0
    flag=0;
    %%Marcaje hacia delante
    %%Marcaje hacia atrás
end
```

El marcaje hacia delante se realiza con el siguiente fragmento de código:

```
for ii=1:m
     if deltai(ii)>0 && marcai(ii)==0
        flag=flag+1;
        dispo=find(ppr(ii,:));
        len=length(dispo);
        for kk=1:len
           if marcaj(dispo(kk))==0
              i(dispo(kk))=ii;
              deltaj(dispo(kk))=min(deltai(ii),...
                 U(ii,dispo(kk))-flujo(ii,dispo(kk)));
           end
           marcai(ii)=1;
        end
     end
 end
```

El marcaje hacia detrás se realiza con el siguiente fragmento de código:

```
for jj=1:n
   if deltaj(jj)>0 && marcaj(jj)==0
      flag=flag+1;
      dispo=find(ppr(:,jj));
      len=length(dispo);
      for ll=1:len
         if flujo(dispo(ll),jj)>0 && marcai(dispo(ll))==0
            j(dispo(ll))=jj;
            deltai(dispo(ll))=min(deltaj(jj),flujo(dispo(ll),jj));
         end
         marcaj(jj)=1;
       end
    end
 end
```

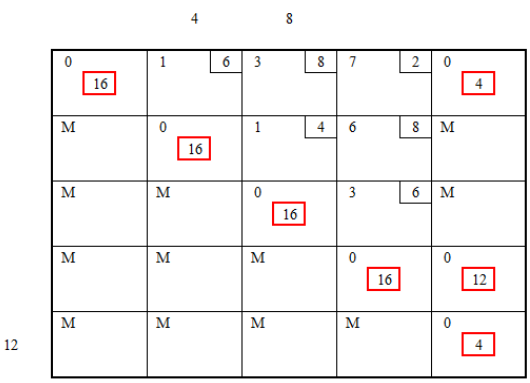
Finalmente, el programa construye las matrices "I" y "J" a partir de los vectores correspondientes, y calcula la variable de decisión "ent"

```
I=[j,deltai,marcai];
J=[i;deltaj;marcaj];
ent=0;
for aa=1:n
    if dem(aa)>0 && marcaj(aa)>0
        ent=aa;
        break;
    end
end
```

Nótese que el programa solo finaliza cuando no añade etiquetas nuevas en una iteración del bucle while, independientemente de que antes ya se haya alcanzado un nodo de demanda. Esta programación se ha decidido así debido a que MATLAB no implementa una función para bucles anidados, por lo que añadir esta funcionalidad de manera manual aumentaría la complejidad de comprensión del código.

Ejemplo.- *Los cálculos que realiza MATLAB en el marcaje de la Tabla 7 se indican en este ejemplo. En primer lugar, se calcula la matriz "ppr" y se inicializan los 6 vectores con valores nulos,*

Tabla 7



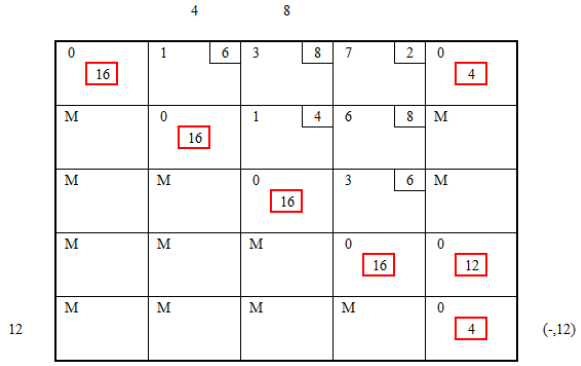
$$ppr = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$j = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad deltai = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad marcai = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$i = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$
$$deltaj = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$
$$marcaj = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$

y, a continuación, se asignan los flujos de oferta en "deltai", obteniéndose la asignación de la Tabla 8.

Tabla 8



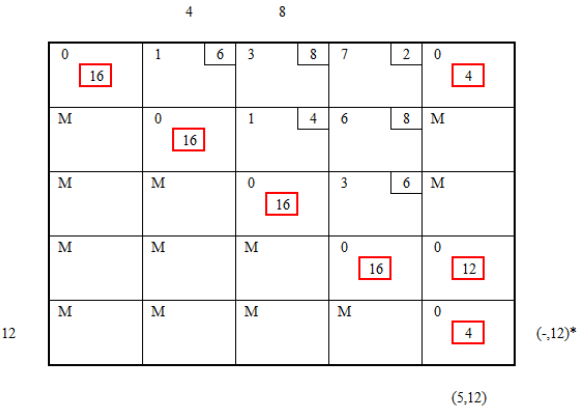
$$j = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad deltai = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 12 \end{bmatrix} \quad marcai = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$i = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$
$$deltaj = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$
$$marcaj = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$

El proceso de marcaje hacia adelante comienza haciendo un barrido con un bucle for en busca de la primera fila no explorada ($marcai = 0$) con un cierto flujo $deltai > 0$. En este ejemplo, esto solamente ocurre para la fila 5. Para esta fila, el código calcula, mediante la matriz "ppr", las columnas de las celdas básicas disponibles sobre las que realizar el marcaje, en este caso, corresponde únicamente a la columna 5, por lo que resulta $dispo = [5]$, vector con longitud $len = 1$

Luego, se identifica para cada elemento de "dispo", si la columna está o no explorada. En este caso, la columna 5 no está explorada (es decir, $marcaj(5) = 0$), entonces se asigna el número de fila en la variable "i", y el flujo admisible en la variable "deltaj". En ese caso, como la celda (5,5) no está acotada superiormente, pueden asignarse las 12 unidades de flujo. La fila 5 se marca como explorada. Con esto se obtienen los resultados de la Tabla 9

Tabla 9



$$j = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad deltai = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 12 \end{bmatrix} \quad marcai = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$i = [0 \quad 0 \quad 0 \quad 0 \quad 5]$$
$$deltaj = [0 \quad 0 \quad 0 \quad 0 \quad 12]$$
$$marcaj = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$

El proceso de marcaje hacia detrás es completamente análogo. Se realiza un barrido por las columnas de la tabla en busca de una columna marcada (deltaj>0) y no explorada (marcaj=0). La columna 5 es la única que cumple estas condiciones. Para esta columna, se calcula el vector de filas disponibles a partir de la matriz "ppr". Se obtiene $dispo = [1 \quad 4 \quad 5]$, de longitud $len = 3$.

Ahora, para cada una de esas filas, se identifica si tiene flujo distinto de 0 para poder disminuirlo, y si la fila asociada no está marcada. La fila 1 cumple estas condiciones, por lo que se le asigna en "i" el número de la columna y en "deltai" el flujo que puede disminuirse, que será el mínimo entre el flujo de esa celda y el flujo en "deltaj", es decir, min(4,12)=4.

La fila 4 también cumple las condiciones para que sea marcada, por lo que se procede de igual manera que para la fila 1. En el caso de la fila 5, ésta ya está marcada como explorada, por lo que no se ve afectada. Por último, la columna 5, se marca como explorada. Con todo esto se obtienen los resultados de la Tabla 10.

Tabla 10

	4		8		
0 16	1 6	3 8	7 2	0 4	(5,4)
M	0 16	1 4	6 8	M	
M	M	0 16	3 6	M	
M	M	M	0 16	0 12	(5,12)
12 M	M	M	M	0 4	(-,12)*
		(5,12)*			

$$j = \begin{bmatrix} 5 \\ 0 \\ 0 \\ 5 \\ 0 \end{bmatrix} \quad deltai = \begin{bmatrix} 4 \\ 0 \\ 0 \\ 12 \\ 12 \end{bmatrix} \quad marcai = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$i = [0 \quad 0 \quad 0 \quad 0 \quad 5]$$
$$deltaj = [0 \quad 0 \quad 0 \quad 0 \quad 12]$$
$$marcaj = [0 \quad 0 \quad 0 \quad 0 \quad 1]$$

Finalmente, repitiendo de manera cíclica el marcaje hacia adelante y hacia detrás hasta que no se modifique la variable "flag" se consiguen obtener las matrices "I" y "J", correspondientes a la Tabla 11.

Ejecutando

```
[I,J,ent]=marcajeff(mr,flujo,of,dem,U)
```

Con los valores de "mr", "flujo", "of", "dem" y "U" obtenidos al ejecutar las funciones anteriores del algoritmo ("estandarizar", "matred", "asignaflujo"), se obtiene:

Tabla 11

	4		8		
0 16	1 6	3 8	7 2	0 4	(5,4)*
M	0 16	1 4	6 8	M	
M	M	0 16	3 6	M	
M	M	M	0 16	0 12	(5,12)*
12 M	M	M	M	0 4	(-,12)*
	(1,4)*		(4,12)*	(5,12)*	

```
I =
     5     4     1
     0     0     0
     0     0     0
     5    12     1
     0    12     1

ent =
     0

J =
     1     0     0     4     5
     4     0     0    12    12
     1     0     0     1     1
```

La variable "ent" toma valor 0 pues no se ha alcanzado ningún nodo de demanda en el marcaje. El algoritmo prosigue con la modificación de la matriz reducida (ver diagrama de flujo, Figura 9).

2.3.4 Modificación de la matriz reducida.

Si en el proceso de marcaje no se ha alcanzado un nodo de demanda, entonces deben modificarse los costes relativos de los arcos (recogidos en la matriz reducida de los costes). En este caso, se habría determinado el flujo máximo en la red básica, habiéndose definido una cortadura $(X, \bar{X})$.

El conjunto X contendrá nodos de oferta (denotados por el conjunto I) y nodos de demanda (denotados por J). Análogamente, el conjunto $\bar{X}$ contendrá nodos de oferta (denotados por el conjunto $\bar{I}$) y nodos de demanda (denotados por $\bar{J}$).

A efectos prácticos, los conjuntos I, J, $\bar{I}$ y $\bar{J}$ estarán formados por:

Tabla 12.- Conjuntos I, J, $\bar{I}$ y $\bar{J}$.

Celdas que forman parte de los conjuntos I, J, $\bar{I}$ y $\bar{J}$	
I	Filas marcadas
$\bar{I}$	Filas no marcadas
J	Columnas marcadas
$\bar{J}$	Columnas no marcadas

La modificación de los costes relativos se hará según la expresión:

$$\overline{r_{ij}} = \begin{cases} r_{ij} - r^* & \forall i \in I, \forall j \in \bar{J} \\ r_{ij} + r^* & \forall i \in \bar{I}, \forall j \in J \\ r_{ij} & otros\ casos \end{cases} \qquad (14)$$

Siendo r* el coste relativo mínimo entre las filas marcadas y columnas no marcadas:

$$r^* = \min r_{ij}\ \forall i \in I, \forall j \in \bar{J} \qquad (15)$$

Las celdas correspondientes a arcos saturados, es decir, a arcos básicos a nivel superior (en los que el flujo coincide con la capacidad), no deben tenerse en cuenta en el cálculo de r*, pues el criterio de optimalidad para estas celdas es que tengan costos relativos negativos.

Según la disposición de la red y la capacidad de sus arcos, es posible que en algúna iteración se obtenga un r*=∞. En ese caso, los costos relativos no pueden modificarse, lo que implica que el flujo en la red no puede aumentarse más. Se habría alcanzado el óptimo, sin satisfacer todas las ofertas y demandas.

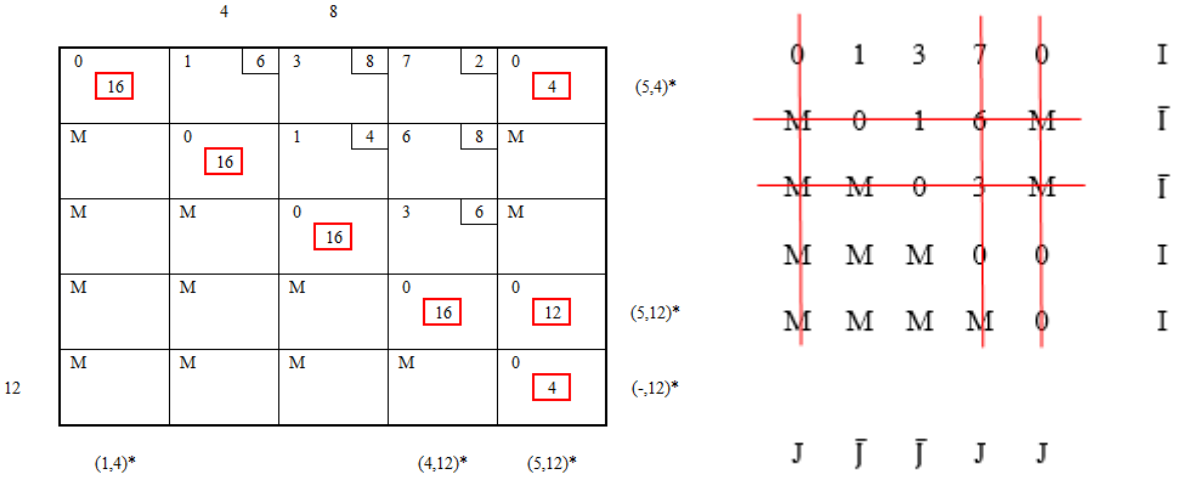
La justificación matemática de este procedimiento puede encontrarse de manera detallada y explícita en la literatura sobre el algoritmo Primal-Dual.

Ejemplo.- *Desde los resultados del marcaje en la Tabla 11, puede procederse a la modificación de la matriz reducida.*

Definiendo los conjuntos I, J, $\bar{I}$ y $\bar{J}$, se realiza una tachadura en las filas no marcadas y en las columnas marcadas

.

Tabla 11



De esta forma, r será el valor mínimo entre las celdas no tachadas de la matriz reducida:*

$$r^* = 1$$

Los nuevos valores $\overline{r_{ij}}$ se obtendrán restando r a las celdas no tachas y sumando r* a las celdas doblemente tachadas. La nueva matriz reducida resulta:*

$$mr = \begin{bmatrix} 0 & 0 & 2 & 7 & 0 \\ M & 0 & 1 & 7 & M \\ M & M & 0 & 4 & M \\ M & M & M & 0 & 0 \\ M & M & M & M & 0 \end{bmatrix}$$

Como puede observarse, la celda correspondiente a la variable x_{12} ahora es una celda básica. Se continua el algoritmo intentando asignar flujos de manera directa y realizando de nuevo el proceso de marcaje.

En MATLAB, esta modificación de los costes relativos se implementa en la función “modificaMR”

```
function [nuevaMR,modmr]=modificaMR(mr,I,J,flujo,U)
```

Esta función sigue la misma filosofía de cálculo que la empleada en el ejemplo, estableciendo unas tachaduras en la matriz y calculando a partir de ellas r*, y finalmente la nueva matriz reducida.

En primer lugar, a partir de las matrices “I” y “J” obtenidas de “marcajeff”, se calcula una matriz que indique el número de veces que se ha tachado una celda. Dicha matriz de denominará “tachadura”

```
[m,n]=size(mr);
marcai=I(:,3);
marcaj=J(3,:);
II=~repmat(marcai,1,n);
JJ=repmat(marcaj,m,1);
tachadura=II+JJ;
```

Para el cálculo de r*, se crea un vector “r” en el que se guardan todos los valores de la matriz reducida asociados a celdas con valor de tachadura igual a 0 (y que representan las celdas no tachadas). Se excluyen de este cálculo las celdas básicas a nivel superior (cuyo flujo coincide con su capacidad). r* será el valor mínimo entre los contenidos en el vector “r”.

```
r=[];
for ii=1:m
    for jj=1:n
        if tachadura(ii,jj)==0 && flujo(ii,jj)<U(ii,jj)
            r=[r mr(ii,jj)];
        end
    end
end
rmin=min(r);
```

Finalmente, se calcula la nueva matriz reducida si $r^* \neq \infty$. Además, se añade una variable que informará de si se podido realizar la modificación de la matriz reducida, “modmr”, que también será una variable de decisión en el algoritmo, como se ve en la Figura 9.

```
if rmin~=inf
    nuevaMR=mr+rmin*(tachadura-1);
    modmr=1;
else
    nuevaMR=mr;
    modmr=0;
end
```

Ejemplo.- *En este ejemplo se continúa con la resolución de la red de la Figura 4. Tras el marcaje se tiene las matrices "I" y "J" siguientes:*

$$I=\begin{bmatrix}5&4&1\\0&0&0\\0&0&0\\5&12&1\\0&12&1\end{bmatrix}\qquad J=\begin{bmatrix}1&0&0&4&5\\4&0&0&12&12\\1&0&0&1&1\end{bmatrix}$$

A partir de los vectores "marcai" y "marcaj" (recordar función "marcajeff", Apartado 2.3.3), se crean las matrices "II" y "JJ". La matriz "II" tiene valores 1 en todos los elementos de las filas no marcadas y 0 en las filas marcadas. La matriz "JJ" tiene valores 1 en todos los elementos de las columnas marcadas y 0 en las columnas no marcadas. La suma de estas dos matrices permite calcular "tachadura"

$$tachadura=II+JJ=\begin{bmatrix}0&0&0&0&0\\1&1&1&1&1\\1&1&1&1&1\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}+\begin{bmatrix}1&0&0&1&1\\1&0&0&1&1\\1&0&0&1&1\\1&0&0&1&1\\1&0&0&1&1\end{bmatrix}=\begin{bmatrix}1&0&0&1&1\\2&1&1&2&2\\2&1&1&2&2\\1&0&0&1&1\\1&0&0&1&1\end{bmatrix}$$

El vector "r" guarda los valores de "mr" de las celdas con tachadura igual a 0.

$$mr=\begin{bmatrix}0&1&3&7&0\\M&0&1&6&M\\M&M&0&3&M\\M&M&M&0&0\\M&M&M&M&0\end{bmatrix}\qquad r=[1\quad M\quad M\quad 3\quad M\quad M]\qquad r^*=min[r]=1$$

Finalmente, la nueva matriz reducida se calcula como:

$$nuevaMR=mr+r^*\ast(tachadura-1)=\begin{bmatrix}0&1&3&7&0\\M&0&1&6&M\\M&M&0&3&M\\M&M&M&0&0\\M&M&M&M&0\end{bmatrix}+1\ast\left[\begin{bmatrix}1&0&0&1&1\\2&1&1&2&2\\2&1&1&2&2\\1&0&0&1&1\\1&0&0&1&1\end{bmatrix}-1\right]$$

$$=\begin{bmatrix}0&1&3&7&0\\M&0&1&6&M\\M&M&0&3&M\\M&M&M&0&0\\M&M&M&M&0\end{bmatrix}+1\ast\begin{bmatrix}0&-1&-1&0&0\\1&0&0&1&1\\1&0&0&1&1\\0&-1&-1&0&0\\0&-1&-1&0&0\end{bmatrix}=\begin{bmatrix}0&0&2&7&0\\M&0&1&7&M\\M&M&0&4&M\\M&M&M&0&0\\M&M&M&M&0\end{bmatrix}$$

El resultado de ejecutar en MATLAB

```
[nuevaMR,modmr]=modificaMR(mr,I,J,flujo,U)
```

es

```
nuevaMR =

     0     0     2     7     0
   Inf     0     1     7   Inf
   Inf   Inf     0     4   Inf
   Inf   Inf   Inf     0     0
   Inf   Inf   Inf   Inf     0


modmr =

     1
```

2.3.5 Algoritmo de Ford Fulkerson. Aumento de flujos.

El aumento de flujos en la red podrá darse cuando se haya encontrado una ruta que comunique un nodo de oferta con un nodo de demanda. Esto ocurrirá cuando se marque una columna con demanda no satisfecha al realizar el marcaje de las filas y columnas.

Por tanto, si el nodo de demanda j alcanzado se ha marcado con un etiqueta (i,Δ), en la red podrá realizarse un incremento de flujo de Δ unidades. A continuación, se procede:

- Decrementando la demanda de la columna j en Δ unidades.
- Incrementando el flujo de la celda (i,j) en Δ unidades.
- En la fila i, marcada con (k,·), decrementando el flujo en la celda (i,k) en Δ unidades.
- En la columna k, marcada con (l,·), incrementar el flujo en la celda (l,k) en Δ unidades.
- Y repitiendo estos dos últimos pasos hasta alcanzar una fila no saturada, cuya oferta se decrementará en Δ unidades.

Para la programación en MATLAB de esta etapa del algoritmo, se implementa la función “aumentaflujo”. Dicha función se ha programado siguiendo exactamente los pasos indicados en el párrafo:

```
function [flujo,of,dem]=aumentaflujo(flujo,of,dem,I,J,ent)
inc=min(dem(ent),J(2,ent));
j=I(:,1);
i=J(1,:);

dem(ent)=dem(ent)-inc;
flujo(i(ent),ent)=flujo(i(ent),ent)+inc;
fila=i(ent);

while of(fila)==0
    flujo(fila,j(fila))=flujo(fila,j(fila))-inc;
    flujo(i(j(fila)),j(fila))=flujo(i(j(fila)),j(fila))+inc;
    fila=i(j(fila));
end
of(fila)=of(fila)-inc;
```

Sobre este script cabe recordar que la variable “ent” guarda el nodo de demanda alcanzado en el proceso de marcaje. A partir de dicho nodo, se realizan los incrementos y decrementos de flujo indicados.

Ejemplo.- *Siguiendo con la resolución de la red de la Figura 4, y una vez realizado el marcaje recogido en la Tabla 11, puede modificarse la matriz reducida, obteniéndose una nueva tabla de transporte.*

Tabla 13.- Nueva tabla de transporte tras modificar la matriz reducida (Figura 4).

		4		8		
	0 16	0 6	2 8	7 2	0 4	
	M	0 16	1 4	7 8	M	
	M	M	0 16	4 6	M	
	M	M	M	0 16	0 12	
12	M	M	M	M	0 4	

Puede observarse que, por asignación directa, no puede incrementarse flujos. Se realiza, pues, un nuevo marcaje sobre esta tabla, cuyo resultado será, empleando la función "marcajeff":

```
I =

     5     4     1     J =
     2     4     1
     0     0     0           1     1     0     4     5        ent =
     5    12     1           4     4     0    12    12
     0    12     1           1     1     0     1     1             2
```

Tabla 14.- Algoritmo de Ford-Fulkerson. Marcaje sobre la Tabla 13.

		4		8			
	0 16	0 6	2 8	7 2	0 4		(5,4)*
	M	0 16	1 4	7 8	M		
	M	M	0 16	4 6	M		
	M	M	M	0 16	0 12		(5,12)
12	M	M	M	M	0 4		(-,12)*
	(1,4)	(1,4) ↑			(5,12)*	Δ=4	

En este punto, hay que recordar que la función "marcajeff" no distingue el hecho de alcanzar un nodo de entrada y continúa realizando asignaciones hasta que todos los nodos estén explorados, de ahí la diferencia entre las etiquetas de la Tabla 14, y las matrices "I" y "J". Sin embargo, esto no afecta al aumento de flujo, pues la ruta encontrada en el marcaje finaliza en el nodo "ent".

Se ha obtenido un incremento $\Delta = 4$. *Realizando los pasos indicados al comienzo de este apartado:*

Tabla 15.- Incremento de flujos sobre la Tabla 13.

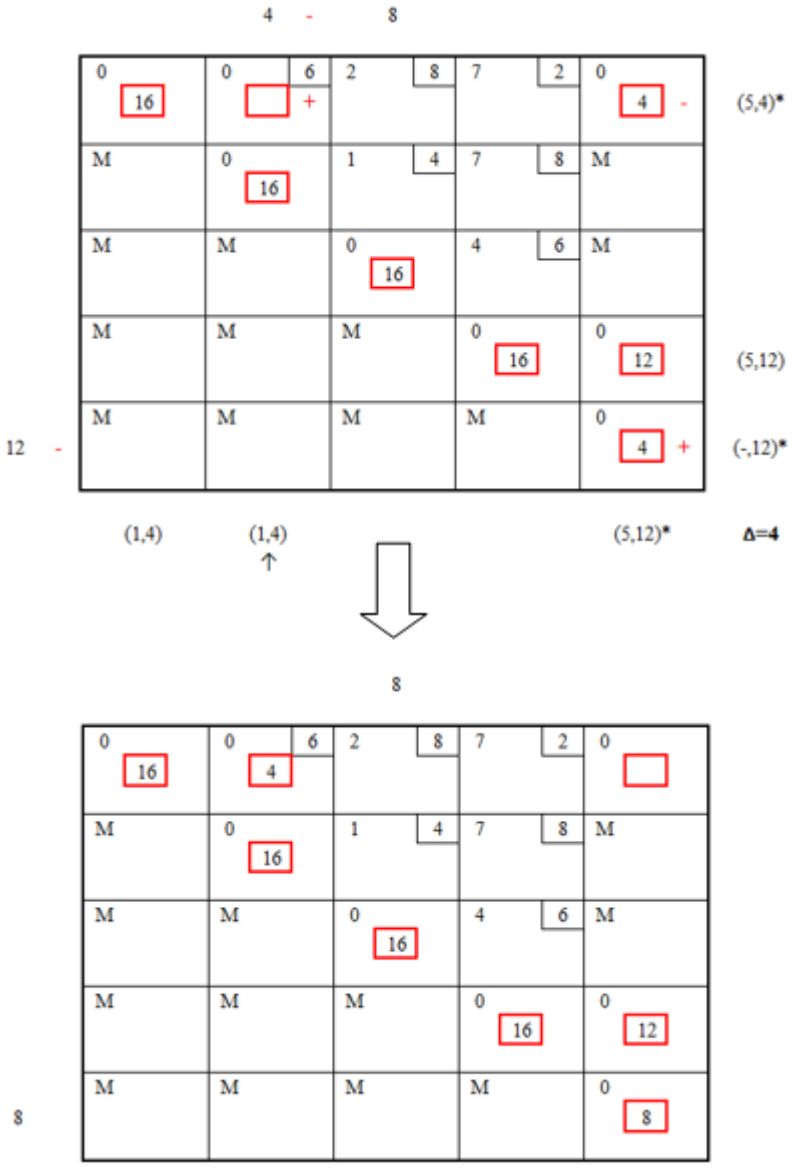


Tabla que se corresponde con los resultados obtenidos al ejecutar "aumentaflujo"

```
[flujo,of,dem]=aumentaflujo(flujo,of,dem,I,J,ent)
```

```
flujo =

    16     4     0     0     0
     0    16     0     0     0
     0     0    16     0     0
     0     0     0    16    12
     0     0     0     0     8

of =

     0     0     0     0     8

dem =

     0     0     8     0     0
```

2.4. Post-procesado del resultado.

La ejecución de las distintas subrutinas en el algoritmo PDTCI (Figura 9), permite la obtención del óptimo del problema estandarizado, cuyo modelo responde a la ecuación (8):

$$\begin{aligned} &Min\ z' = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot x_{ij} \\ &s.a. \sum_{k\in D(j)} x_{jk} - \sum_{i\in A(j)} x_{ij} = k_j \quad \forall j = 1,2,\dots,n \\ &0 \le x_{ij} \le u_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{8}$$

Sin embargo, la obtención del óptimo del problema original pasa por deshacer los cambios de variable, y obtener la solución para el problema modelado según la ecuación (1):

$$\begin{aligned} &Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot y_{ij} \\ &s.a. \sum_{k\in D(j)} y_{jk} - \sum_{i\in A(j)} y_{ij} = K_j \quad \forall j = 1,2,\dots,n \\ &L_{ij} \le y_{ij} \le U_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{1}$$

En resumidas cuentas, una vez resuelto el algoritmo PDTCI, se requiere de un post-procesado de los datos para obtener el óptimo del problema original y_{ij}, que se calculará a partir del cambio de variable (5):

$$x_{ij} = y_{ij} - L_{ij} \qquad (5)$$

En MATLAB, este post-procesado de los resultados, se ejecuta con la función "postproc"

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

Esta función calculará, no solo el flujo óptimo sobre el problema original, sino también el valor de la función objetivo y los niveles de oferta y demanda reales sobre la red original, que pueden diferir de los niveles originales en caso de que se hayan relajado las restricciones de demanda.

Para calcular el flujo real sobre la red (óptimo del problema original, y_{ij}), "postproc" crea la matriz de cotas inferiores a partir de los datos de la red original (matriz "arcos", ver Apartado 2.1). Esta matriz se suma a la matriz "flujo", que indica el flujo a coste mínimo óptimo del problema estandarizado.

```
[m,n]=size(flujo);
nodosi=arcos(:,1);
nodosj=arcos(:,2);
Lb=arcos(:,3);
L=full(sparse(nodosi,nodosj,Lb,m,n));
flujoreal=flujo+L;
```

La función objetivo, "z", se calcula multiplicando elemento a elemento las matrices de costes "C" y el flujo real calculado con las líneas de código anteriores:

```
z=sum(sum(C.*flujoreal,'omitnan'));
```

La etiqueta 'omitnan' se introduce para que en el cálculo se ignoren elementos que den lugar a error, que generalmente ocurre en los arcos con coste infinito y flujo nulo (arcos no definidos en la red original).

Los niveles de oferta y demanda reales se calculan a partir de los flujos y_{ij}, según la siguiente expresión:

$$K_j = \sum_{k \in D(j)} y_{jk} - \sum_{i \in A(j)} y_{ij} \quad \forall j = 1,2,\dots,n \qquad (16)$$

En cuanto al cálculo en MATLAB, se crea un vector "y" que recoge los valores de flujo real para los arcos de la red original (los índices de dichos arcos están guardados en las dos primeras columnas de la matriz "arcos").

Luego, con los valores de ese vector "y" se calculan los sumatorios de la ecuación (16), y finalmente las K_j.

```
for ii=1:length(nodosi)
    y(ii)=flujoreal(nodosi(ii),nodosj(ii));
end
Kreales=zeros(1,n);
for ii=1:n
    Di=sum(y(nodosi==ii));
    Ai=sum(y(nodosj==ii));
    Kreales(1,ii)=Di-Ai;
end
```

Adicionalmente, la función "postproc" representa el óptimo mediante un grafo orientado en el que en cada arco aparece su flujo y_{ij}:

```
G=digraph(nodosi,nodosj,y);
red=plot(G,'EdgeLabel',G.Edges.Weight);
red.NodeColor='r';
red.MarkerSize=5;
```

Ejemplo.- *El flujo óptimo correspondiente a la red de la Figura 4, puede obtenerse ejecutando el siguiente código:*

```
arcos=[1 2 2 8 1; 1 3 4 12 3;1 4 0 2 7; 2 3 2 6 1;2 4 0 8 6;3 4 6 12 3];
K=[10 -4 -8 6];
[U,C,of,dem]=estandarizar(arcos,K,1);
[flujo,mr]=pdtci(C,of,dem,U);
```

obteniéndose

```
flujo =

    16     4     0     0     0
     0    16     0     0     0
     0     0    16     0     0
     0     0     0    16    12
     0     0     0     0     8
```

Como puede observarse, este flujo coincide con el representado en la Tabla 15. Al realizar el marcaje sobre esta tabla, se acaba obteniendo un $r^* = M$, por lo que no puede modificarse la matriz reducida, habiendose alcanzado el óptimo.

En cuanto a los cálculos realizados por "postproc", en primer lugar, se calcula la matriz "L" de límites inferiores a partir de la información de "arcos"

$$arcos = \begin{bmatrix} 1 & 2 & 2 & 8 & 1 \\ 1 & 3 & 4 & 12 & 3 \\ 1 & 4 & 0 & 2 & 7 \\ 2 & 3 & 2 & 6 & 1 \\ 2 & 4 & 0 & 8 & 6 \\ 3 & 4 & 6 & 12 & 3 \end{bmatrix} \rightarrow L = \begin{bmatrix} 0 & 2 & 4 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

El óptimo se obtiene sumando:

$$flujoreal = flujo + L = \begin{bmatrix} 16 & 4 & 0 & 0 & 0 \\ 0 & 16 & 0 & 0 & 0 \\ 0 & 0 & 16 & 0 & 0 \\ 0 & 0 & 0 & 16 & 12 \\ 0 & 0 & 0 & 0 & 8 \end{bmatrix} + \begin{bmatrix} 0 & 2 & 4 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 16 & 6 & 4 & 0 & 0 \\ 0 & 16 & 2 & 0 & 0 \\ 0 & 0 & 16 & 6 & 0 \\ 0 & 0 & 0 & 16 & 12 \\ 0 & 0 & 0 & 0 & 8 \end{bmatrix}$$

Para calcular el valor de la función objetivo se multiplican, elemento a elemento, la matriz "C" por la matriz "flujoreal":

$$C = \begin{bmatrix} 0 & 1 & 3 & 7 & 0 \\ M & 0 & 1 & 6 & M \\ M & M & 0 & 3 & M \\ M & M & M & 0 & 0 \\ M & M & M & M & 0 \end{bmatrix} \rightarrow \; z = 1 \cdot 6 + 3 \cdot 4 + 7 \cdot 0 + 1 \cdot 2 + 6 \cdot 0 + 3 \cdot 6 = 38$$

Para calcular los niveles de oferta y demanda reales, se emplea la ecuación (16). Para ello, en primer lugar se calcula el vector "y", que guarda, para los arcos (i,j) de la red original, los valores de "flujoreal". Es decir:

$$y = \begin{bmatrix} 6 \\ 4 \\ 0 \\ 2 \\ 0 \\ 6 \end{bmatrix}$$

Luego, para cada nodo, usando las dos primeras columnas de "arcos" (que se corresponden con la identificación (i,j) de cada arco de la red original), puede aplicarse la ecuación (16).

Por ejemplo, para el nodo 2:

$$K_2 = \sum_{k \in D(2)} y_{2k} - \sum_{i \in A(2)} y_{i2} \; = (2 + 0) - (6) = -4$$

Para el resto de nodos, se obtiene:

$$Kreales = [10 \quad -4 \quad 0 \quad -6 \quad 0]$$

Así, al ejecutar:

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

Se obtiene finalmente el óptimo sobre la red original:

```
flujoreal =

    16     6     4     0     0
     0    16     2     0     0
     0     0    16     6     0
     0     0     0    16    12
     0     0     0     0     8

Kreales =

    10    -4     0    -6     0

z =

    38
```

y el grafo que representa dichos flujos óptimos, y_{ij}:

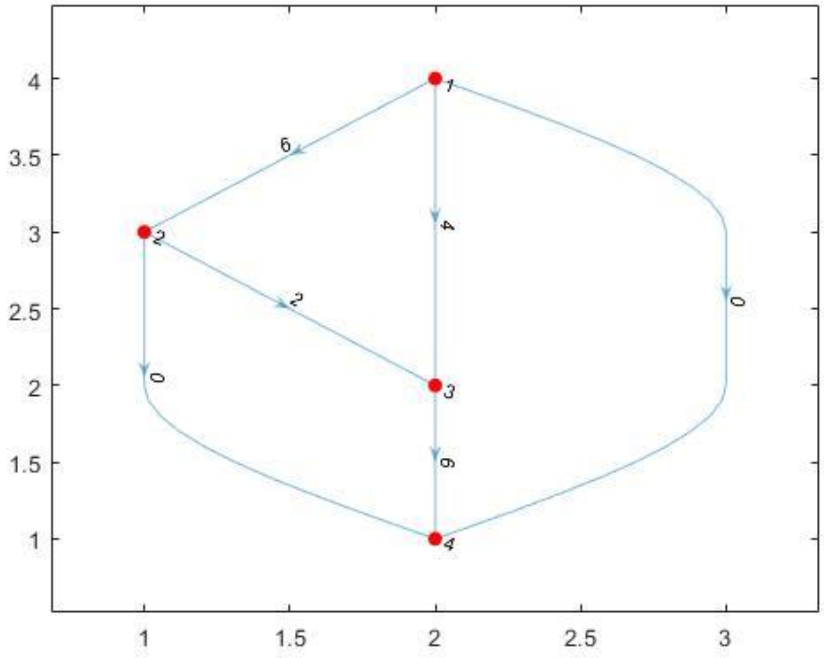


Figura 10.- Flujo óptimo de la red de la Figura 1 (grafo MATLAB).

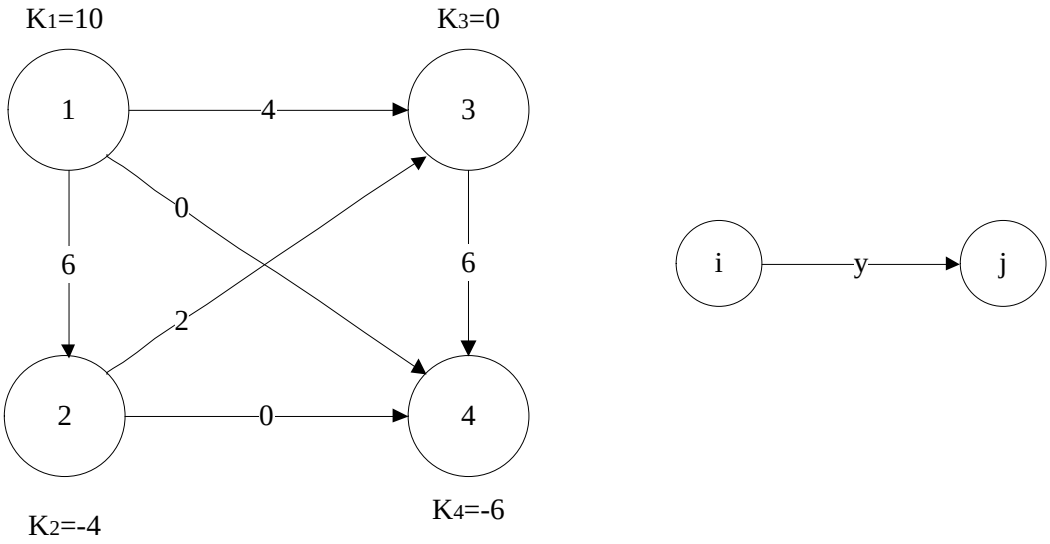


Figura 11.- Flujo óptimo de la red de la Figura 1.

2.5. Resumen de la programación del algoritmo PDTCI.

Llegados a este punto, se ha resuelto el problema de flujo máximo a coste mínimo independiente del flujo aplicando el algoritmo Primal-Dual sobre la tabla de transporte. Para ello, se han creado en MATLAB varias funciones o rutinas que realizan de manera iterativa los cálculos necesarios para la resolución del problema.

El objetivo de este apartado es mostrar las distintas etapas del programa MATLAB que aplica de manera completa el algoritmo PDTCI descrito a lo largo de todo el capítulo, a través del siguiente diagrama de flujo:

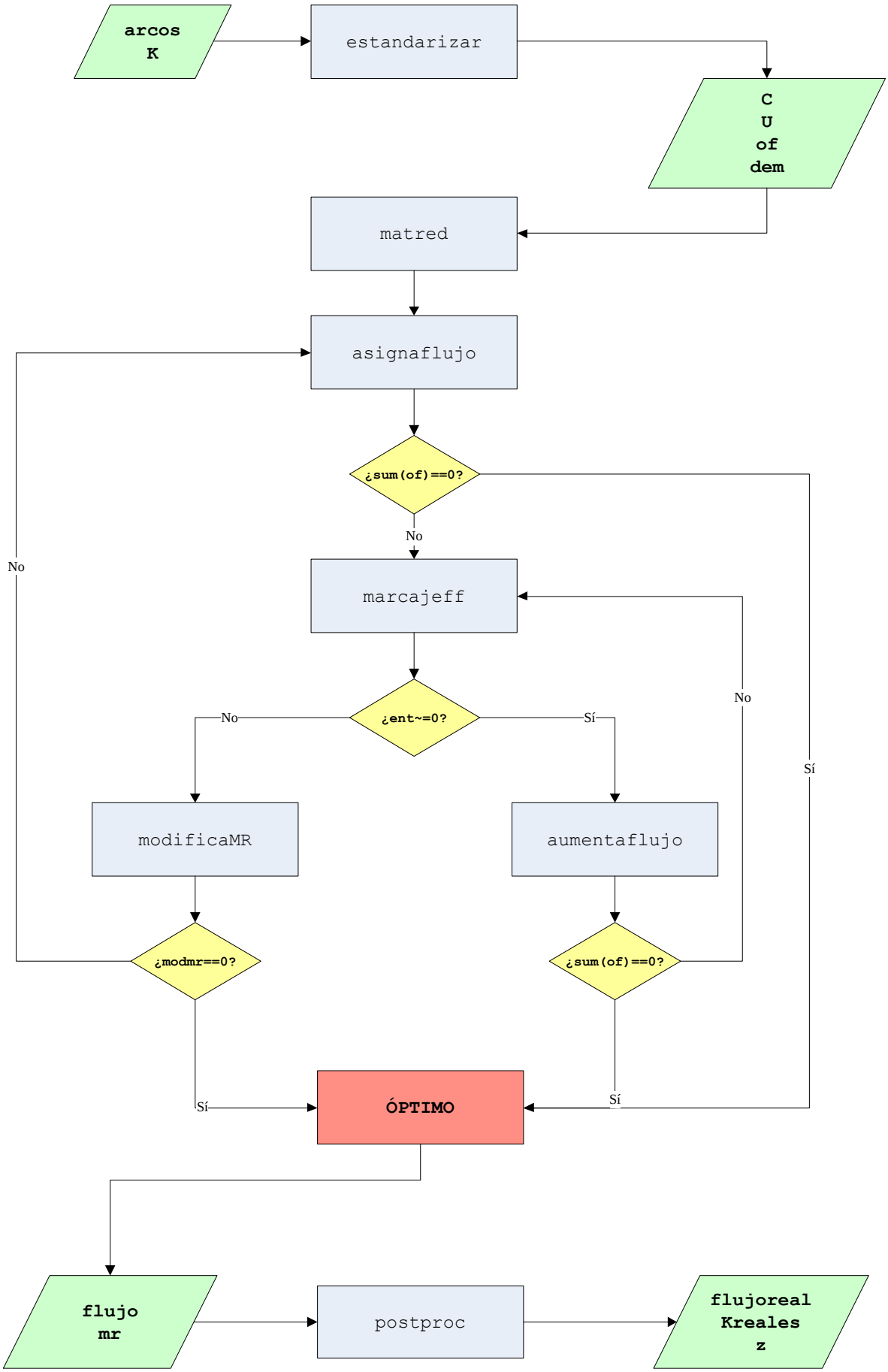


Figura 12.- Diagrama de flujo del algoritmo PDTCI.

Además, se recuerdan en la siguiente tabla, las variables de entrada y salida que intervienen en cada una de las funciones implementadas:

Tabla 16.- Variables de entrada y salida para las subrutinas del algoritmo PDTCI.

Función	Descripción	Variables de salida	Variables de entrada
estandarizar	Tranformación de la red original en una resoluble realizando los cambios de variable adecuados	`U,C,of,dem`	`arcos,K, metodo`
ofdem	Cálculo de ofertas y demandas (función interna en "estandarizar")	`of,dem,s`	`k`
metodo1	Estandarización de la red original según el método 1 (función interna en "estandarizar")	`costes`	`k,costes,s`
metodo2	Estandarización de la red original según el método 2 (función interna en "estandarizar")	`costes`	`costes,s`
metodo3	Estandarización de la red original según el método 3 (función interna en "estandarizar")	`costes,U, of,dem`	`k,costes,U`
pdtci	Resolución de la red estandarizada aplicando el algoritmo PDTCI	`flujo,mr`	`C,of,dem,U`
matred	Determinación de los arcos básicos de la red estandarizada (función interna en "pdtci")	`C,B`	`A`
asignaflujo	Asignación directa de flujos sobre la tabla de transporte (función interna en "pdtci")	`flujo,of,dem`	`mr,flujo, of,dem,U`
marcajeff	Marcaje de los nodos según el algoritmo de Ford-Fulkerson (función interna en "pdtci")	`I,J,ent`	`mr,flujo, of,dem,U`
aumentaflujo	Aumento de flujos en la red básica según el algoritmo de Ford-Fulkerson (función interna en "pdtci")	`flujo,of,dem`	`flujo,of,dem, I,J,ent`
modificaMR	Modificación de la matriz reducida (función interna en "pdtci")	`nuevaMR, modmr`	`mr,I,J, flujo,U`
postproc	Obtención del flujo óptimo en la red original	`flujoreal, Kreales,z`	`flujo,C, arcos`

2.6. Aplicación práctica del código implementado.

En este apartado se muestra la resolución completa de un problema de flujo máximo a coste mínimo, mostrando paso a paso los resultados en cada una de las iteraciones del algoritmo PDTCI implementado en MATLAB. Además, en el Apéndice B, se expone la resolución completa de un segundo problema.

Para conseguir que MATLAB muestre por pantalla los resultados de cada uno de los cálculos, se añaden unas líneas de código en la función "pdtci", empleando las funciones de MATLAB "disp" y "pause" para conseguir ese resultado. El código completo se puede conseguir solicitándolo al autor.

Se resolverá la red representada por la siguiente figura:

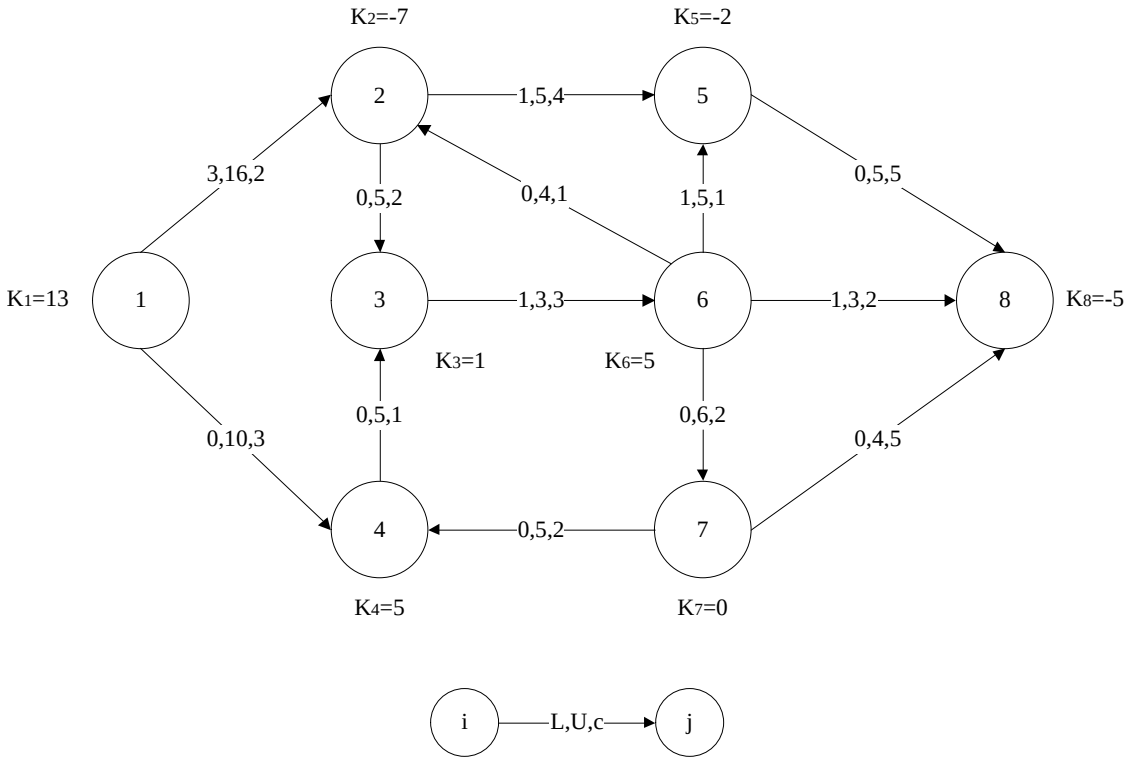


Figura 13.- Red de transporte a resolver en el Apartado 2.6.

En primer lugar, a partir de la red se obtienen la matriz "arcos" y el vector "K"

$$arcos = \begin{bmatrix} 1 & 2 & 3 & 16 & 2 \\ 1 & 4 & 0 & 10 & 3 \\ 2 & 3 & 0 & 5 & 2 \\ 2 & 5 & 1 & 5 & 4 \\ 3 & 6 & 1 & 3 & 3 \\ 4 & 3 & 0 & 5 & 1 \\ 5 & 8 & 0 & 5 & 5 \\ 6 & 2 & 0 & 4 & 1 \\ 6 & 5 & 1 & 5 & 1 \\ 6 & 7 & 0 & 6 & 2 \\ 6 & 8 & 1 & 3 & 2 \\ 7 & 4 & 0 & 5 & 2 \\ 7 & 8 & 0 & 4 & 5 \end{bmatrix} \qquad K = [13 \quad -7 \quad 1 \quad 5 \quad -2 \quad 5 \quad 0 \quad -5]$$

A continuación, se muestra de manera detallada la resolución de esta red para cada uno de los métodos de estandarización posibles.

2.6.1 Resolución del problema estandarizando la red con el método 1.

Al ejecutar

```
arcos=[1 2 3 16 2;1 4 0 10 3;2 3 0 5 2;2 5 1 5 4;3 6 1 3 3;4 3 0 5 1;5 8 0
5 5;6 2 0 4 1;6 5 1 5 1;6 7 0 6 2;6 8 1 3 2;7 4 0 5 2;7 8 0 4 5];
K=[13 -7 1 5 -2 5 0 -5];
[U,C,of,dem]=estandarizar(arcos,K,1)
```

Se realiza el cambio de variable asociado a eliminar las cotas inferiores, que implica calcular el vector "k" según las expresiones siguientes.

$$u_{ij} = U_{ij} - L_{ij} \qquad (6)$$

$$k_j = K_j - \left(\sum_{k \in D(j)} L_{jk} - \sum_{i \in A(j)} L_{ij} \right) \qquad (7)$$

A partir de dicho vector "k", se calculan las ofertas y demandas de cada nodo y se redefine el grafo de la Figura 13.

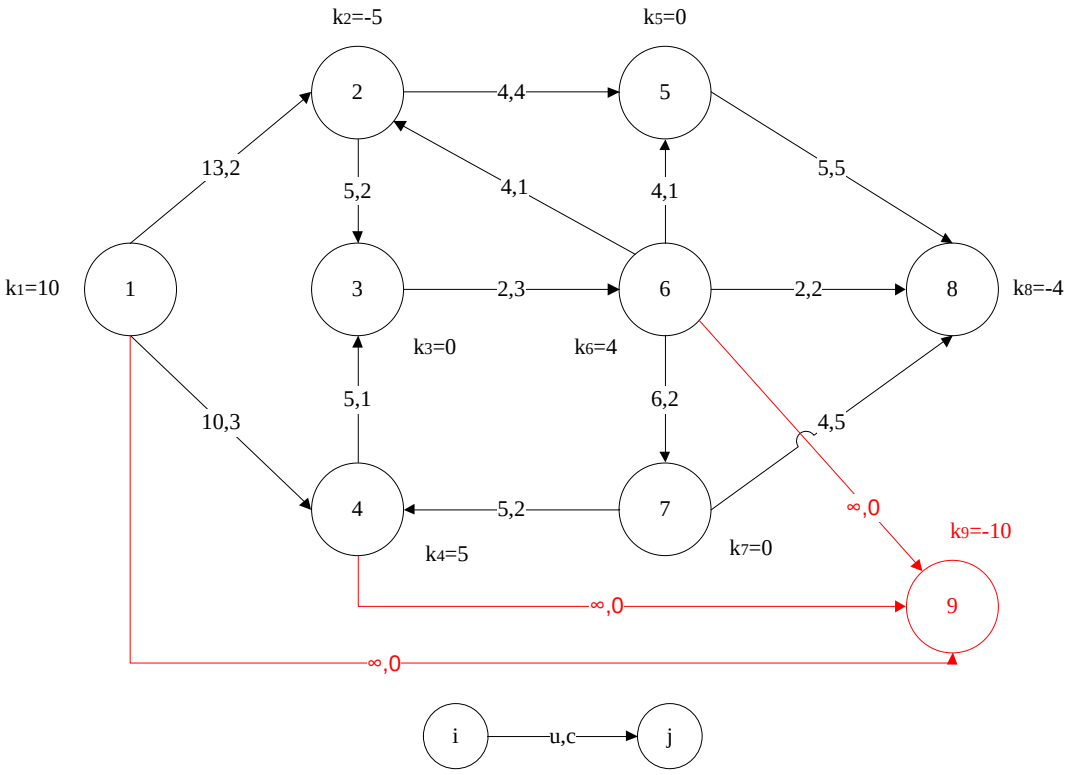


Figura 14.- Grafo de la Figura 13, estandarizado mediante el método 1.

Se observa que la red tiene un exceso de oferta de 10 unidades por lo que se crea un nodo ficticio, 9, que absorba dicho exceso de oferta, y que estará unido a los nodos de oferta mediante arcos ficticios de coste nulo y capacidad infinita. La ejecución de la función estandarizar proporciona los siguientes resultados:

```
U =

   Inf    13   Inf    10   Inf   Inf   Inf   Inf   Inf
   Inf   Inf     5   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf     2   Inf   Inf   Inf
   Inf   Inf     5   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     5   Inf
   Inf     4   Inf   Inf     4   Inf     6     2   Inf
   Inf   Inf   Inf     5   Inf   Inf   Inf     4   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf


C =

     0     2   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     1     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     1   Inf   Inf     1     0     2     2     0
   Inf   Inf   Inf     2   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0


of =

    29    19    19    24    19    23    19    19    19


dem =

    19    24    19    19    19    19    19    23    29
```

También se obtiene el siguiente grafo, que también representa a la Figura 13, indicando los costes de cada arco.

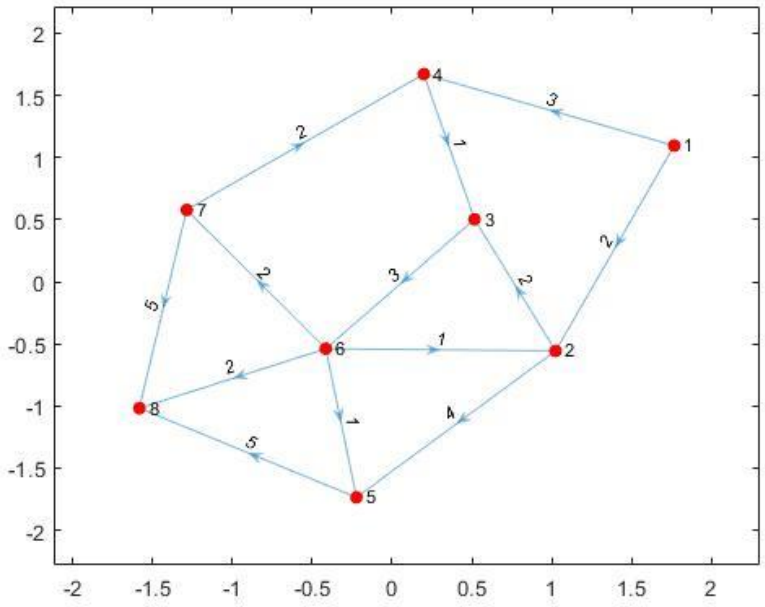


Figura 15.- Grafo de la Figura 13, representado en MATLAB.

Este grafo que aparece en pantalla, corresponde al grafo de la Figura 13, indicando únicamente los costes de cada arco, por lo que es independiente del método de estandarización empleado. Por ello, se omitirá este grafo en el desarrollo de la resolución del problema según los otros dos métodos.

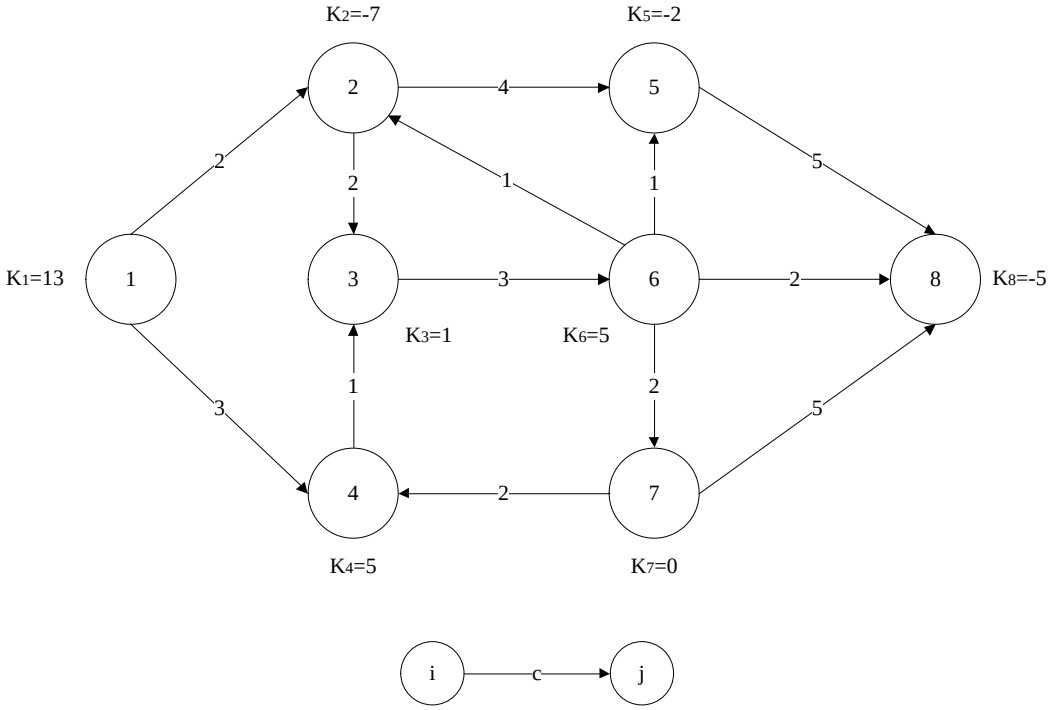


Figura 16.- Grafo realista equivalente al grafo de la Figura 15.

El siguiente paso es ejecutar la función “pdtci”

```
[flujo,mr]=pdtci(C,of,dem,U)
```

En primer lugar, se muestra por pantalla la matriz reducida de los costes o matriz de costes relativos.

```
ITERACIÓN 1
Costos relativos
     0     2   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     1     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     1   Inf   Inf     1     0     2     2     0
   Inf   Inf   Inf     2   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0
```

Luego se realiza la asignación directa de los costes, actualizando los vectores de oferta y demanda:

```
Flujo actual
    19     0     0     0     0     0     0     0    10
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     0     0     0     0    19     0     0     4
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    10

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     9

     0     5     0     0     0     0     0     4     0
```

A continuación, se realiza el marcaje de filas y columnas sobre la tabla:

```
Marcaje del algoritmo FF
     9     9     1
     0     0     0
     0     0     0
     9     5     1
     0     0     0
     9     4     1
     0     0     0
     0     0     0
     0     9     1

     1     0     0     4     0     6     0     0     9
     9     0     0     5     0     4     0     0     9
     1     0     0     1     0     1     0     0     1
```

Como no se ha alcanzado un nodo de demanda, se modifica la matriz reducida, pasando a la segunda iteración del algoritmo PDTCI.

```
ITERACIÓN 2
Costos relativos
     0     1   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     4   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1     0
   Inf   Inf   Inf     3   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0
```

Se realiza nuevamente el proceso de asignación directa de flujos, aunque en este caso, no se puede aumentar flujo por inspección visual, por lo que se obtiene la misma matriz de flujos y las mismas ofertas y demandas anteriores. Aunque MATLAB mostrará en cada iteración el resultado de aplicar la función “asignaflujo”, aquí se omitirán los resultados de aquellas iteraciones en las que no se asigne flujo de manera directa.

Realizando un nuevo marcaje, se obtiene:

```
Marcaje del algoritmo FF
     9     9     1
     2     4     1
     3     5     1
     9     5     1
     5     4     1
     9     4     1
     0     0     0
     0     0     0
     0     9     1

     1     6     4     4     6     6     0     0     9
     9     4     5     5     4     4     0     0     9
     1     1     1     1     1     1     0     0     1
```

Habiéndose alcanzado el nodo 2, que es uno de los nodos con demanda no nula, por lo que se puede aumentar el flujo en la red.

```
Flujo actual
    19     0     0     0     0     0     0     0    10
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     4     0     0     0    19     0     0     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    14

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     5

     0     1     0     0     0     0     0     4     0
```

Se inicia la tercera iteración del algoritmo. MATLAB mostrará por pantalla la matriz de costos relativos al inicio de cada iteración. En adelante, se muestran, sin comentar, los sucesivos resultados que ofrece MATLAB en su ventana de comandos, en la ejecución del algoritmo PDTCI hasta la obtención del óptimo:

```
ITERACIÓN 3
Costos relativos
     0     1   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     4   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1     0
   Inf   Inf   Inf     3   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     5     1
     0     0     0
     3     5     1
     9     5     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     5     1

     1     0     4     4     0     0     0     0     9
     5     0     5     5     0     0     0     0     5
     1     0     1     1     0     0     0     0     1


ITERACIÓN 4
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1     1
   Inf   Inf   Inf     4   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0
```

```
Marcaje del algoritmo FF
     9     5     1
     2     5     1
     3     5     1
     9     5     1
     5     4     1
     2     4     1
     0     0     0
     0     0     0
     0     5     1

     1     1     4     4     6     6     0     0     9
     5     5     5     5     4     4     0     0     5
     1     1     1     1     1     1     0     0     1

Flujo actual
    19     1     0     0     0     0     0     0     9
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     4     0     0     0    19     0     0     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    15

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     4

     0     0     0     0     0     0     0     4     0


ITERACIÓN 5
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1     1
   Inf   Inf   Inf     4   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0
```

```
Marcaje del algoritmo FF
     9     4     1
     2     4     1
     3     4     1
     9     4     1
     5     4     1
     2     4     1
     0     0     0
     0     0     0
     0     4     1

     1     1     4     4     6     6     0     0     9
     4     4     4     4     4     4     0     0     4
     1     1     1     1     1     1     0     0     1


ITERACIÓN 6
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     4   Inf
   Inf     0   Inf   Inf     0     0     0     0     1
   Inf   Inf   Inf     5   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     4     1
     2     4     1
     3     4     1
     9     4     1
     5     4     1
     2     4     1
     7     4     1
     8     2     1
     0     4     1

     1     1     4     4     6     6     6     6     9
     4     4     4     4     4     4     4     2     4
     1     1     1     1     1     1     1     1     1
```

```
Flujo actual
    19     3     0     0     0     0     0     0     7
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     2     0     0     0    19     0     2     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    17

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     2

     0     0     0     0     0     0     0     2     0


ITERACIÓN 7
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     4   Inf
   Inf     0   Inf   Inf     0     0     0     0     1
   Inf   Inf   Inf     5   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     2     1
     2     2     1
     3     2     1
     9     2     1
     5     2     1
     2     2     1
     7     2     1
     0     0     0
     0     2     1

     1     1     4     4     6     6     6     6     9
     2     2     2     2     2     2     2     0     2
     1     1     1     1     1     1     1     0     1
```

```
ITERACIÓN 8
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     0   Inf
   Inf     0   Inf   Inf     0     0     0    -4     1
   Inf   Inf   Inf     5   Inf   Inf     0     1   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0


Marcaje del algoritmo FF
     9     2     1
     2     2     1
     3     2     1
     9     2     1
     5     2     1
     2     2     1
     7     2     1
     8     2     1
     0     2     1

     1     1     4     4     6     6     6     5     9
     2     2     2     2     2     2     2     2     2
     1     1     1     1     1     1     1     1     1
```

Finalmente, se obtiene el siguiente óptimo:

```
flujo =

    19     5     0     0     0     0     0     0     5
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    17     0     0     2     0
     0     0     0     0     2    19     0     2     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    19
```

Para finalizar el algoritmo, se ejecuta la función “postproc”.

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

Con esta función se obtiene el óptimo de la red original, así como los niveles de oferta y demanda que realmente se satisfacen en la red y el valor de la función objetivo.

```
flujoreal =

    19     8     0     0     0     0     0     0     5
     0    19     0     0     1     0     0     0     0
     0     0    19     0     0     1     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    17     0     0     2     0
     0     0     0     0     3    19     0     3     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    19


Kreales =

     8    -7     1     0    -2     5     0    -5     0


z =

    42
```

También se obtiene el siguiente grafo, en el que se indican los flujos óptimos para la red del problema propuesto:

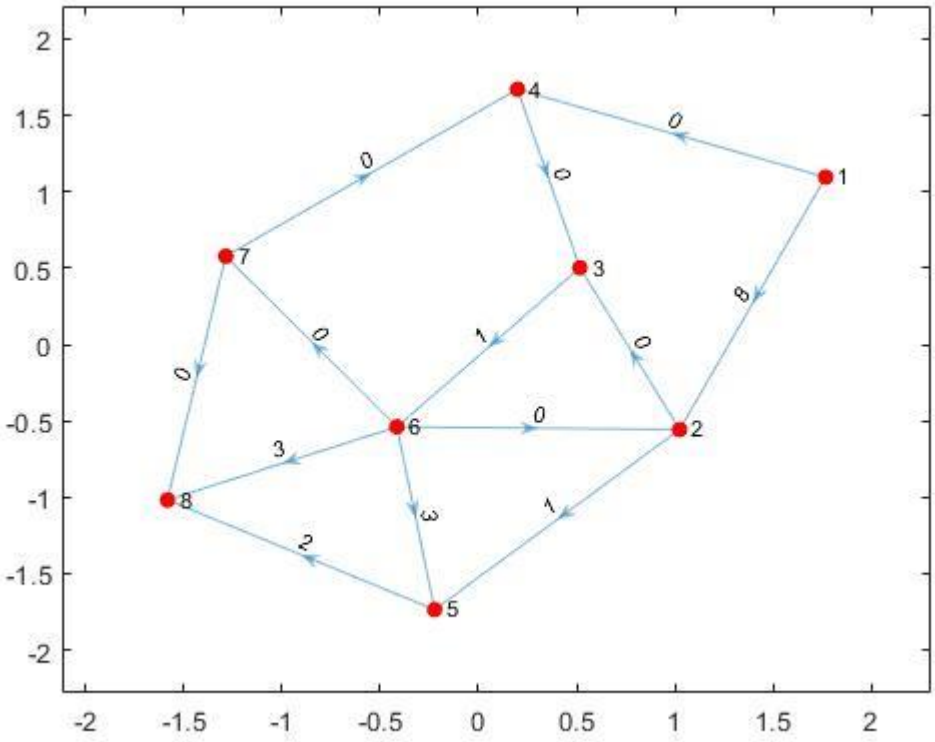


Figura 17.- Flujo óptimo en la red de la Figura 13 (método 1).

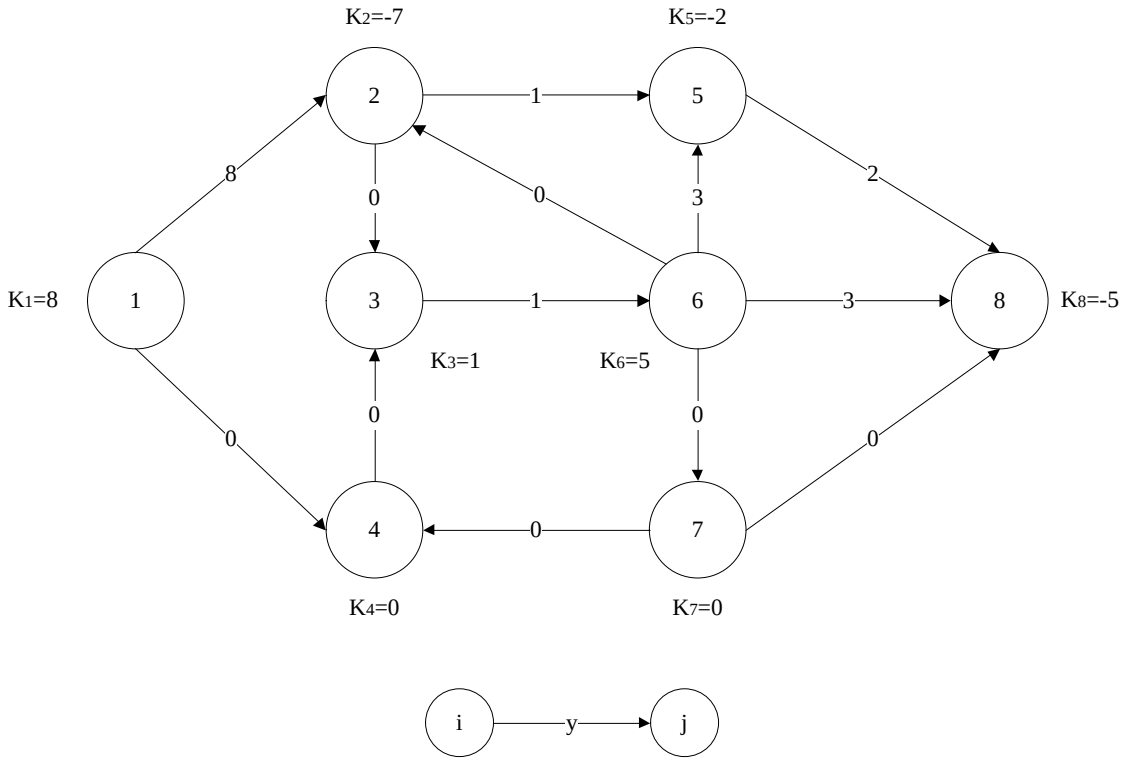


Figura 18.- Grafo realista equivalente al grafo de la Figura 17.

2.6.2 Resolución del problema estandarizando la red con el método 2.

De nuevo, al ejecutar

```
arcos=[1 2 3 16 2;1 4 0 10 3;2 3 0 5 2;2 5 1 5 4;3 6 1 3 3;4 3 0 5 1;5 8 0
5 5;6 2 0 4 1;6 5 1 5 1;6 7 0 6 2;6 8 1 3 2;7 4 0 5 2;7 8 0 4 5];
K=[13 -7 1 5 -2 5 0 -5];
[U,C,of,dem]=estandarizar(arcos,K,2)
```

Se realiza el cambio de variable asociado a eliminar las cotas inferiores, se calcula el vector “k” y se redefine el grafo de la Figura 13.

En este caso, el nodo adicional, 9, que absorbe el exceso de oferta, estará unido mediante arcos ficticios a todos los nodos de la red, resultando el siguiente grafo.

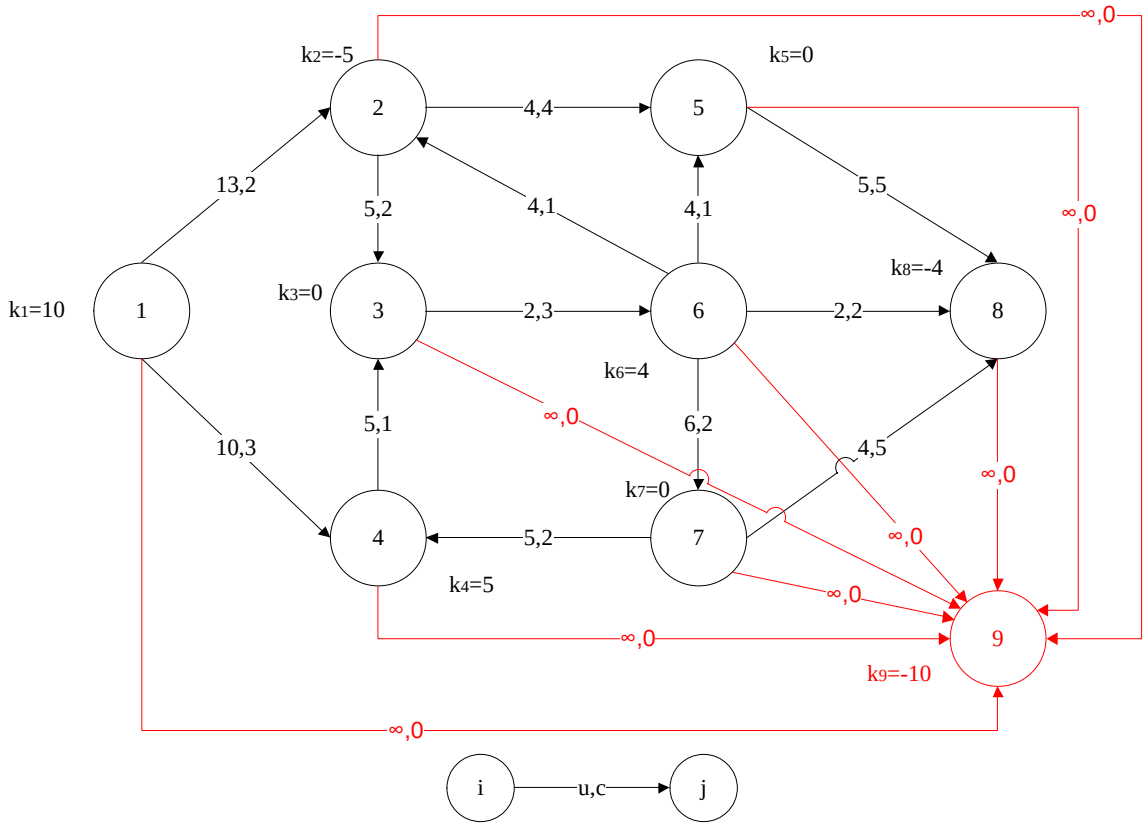


Figura 19.- Grafo de la Figura 13, estandarizado mediante el método 2.

La ejecución de la función “estandarizar” proporciona los siguientes resultados:

```
U =

   Inf    13   Inf    10   Inf   Inf   Inf   Inf   Inf
   Inf   Inf     5   Inf     4   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf     2   Inf   Inf   Inf
   Inf   Inf     5   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     5   Inf
   Inf     4   Inf   Inf     4   Inf     6     2   Inf
   Inf   Inf   Inf     5   Inf   Inf   Inf     4   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf


C =

     0     2   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     3   Inf   Inf     0
   Inf   Inf     1     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5     0
   Inf     1   Inf   Inf     1     0     2     2     0
   Inf   Inf   Inf     2   Inf   Inf     0     5     0
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0
```

```
of =

    29    19    19    24    19    23    19    19    19


dem =

    19    24    19    19    19    19    19    23    29
```

Se muestran a continuación, sin comentar, los resultados intermedios que se obtienen al ejecutar la función "pdtci"

```
[flujo,mr]=pdtci(C,of,dem,U)


ITERACIÓN 1
Costos relativos
     0     2   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     3   Inf   Inf     0
   Inf   Inf     1     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5     0
   Inf     1   Inf   Inf     1     0     2     2     0
   Inf   Inf   Inf     2   Inf   Inf     0     5     0
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Flujo actual
    19     0     0     0     0     0     0     0    10
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     0     0     0     0    19     0     0     4
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    10

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     9

     0     5     0     0     0     0     0     4     0
```

```
Marcaje del algoritmo FF
     9     9     1
     0     0     0
     0     0     0
     9     5     1
     0     0     0
     9     4     1
     0     0     0
     0     0     0
     0     9     1

     1     0     0     4     0     6     0     0     9
     9     0     0     5     0     4     0     0     9
     1     0     0     1     0     1     0     0     1


ITERACIÓN 2
Costos relativos
     0     1   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf     1
   Inf   Inf     0   Inf   Inf     4   Inf   Inf     1
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5     1
   Inf     0   Inf   Inf     0     0     1     1     0
   Inf   Inf   Inf     3   Inf   Inf     0     5     1
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     1
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0
Marcaje del algoritmo FF
     9     9     1
     2     4     1
     3     5     1
     9     5     1
     5     4     1
     9     4     1
     0     0     0
     0     0     0
     0     9     1

     1     6     4     4     6     6     0     0     9
     9     4     5     5     4     4     0     0     9
     1     1     1     1     1     1     0     0     1
```

```
Flujo actual
    19     0     0     0     0     0     0     0    10
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     4     0     0     0    19     0     0     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    14

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     5

     0     1     0     0     0     0     0     4     0


ITERACIÓN 3
Costos relativos
     0     1   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     2   Inf     4   Inf   Inf   Inf     1
   Inf   Inf     0   Inf   Inf     4   Inf   Inf     1
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5     1
   Inf     0   Inf   Inf     0     0     1     1     0
   Inf   Inf   Inf     3   Inf   Inf     0     5     1
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     1
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     5     1
     0     0     0
     3     5     1
     9     5     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     5     1

     1     0     4     4     0     0     0     0     9
     5     0     5     5     0     0     0     0     5
     1     0     1     1     0     0     0     0     1
```

```
ITERACIÓN 4
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf     2
   Inf   Inf     0   Inf   Inf     3   Inf   Inf     1
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5     2
   Inf     0   Inf   Inf     0     0     1     1     1
   Inf   Inf   Inf     4   Inf   Inf     0     5     2
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     2
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     5     1
     2     5     1
     3     5     1
     9     5     1
     5     4     1
     2     4     1
     0     0     0
     0     0     0
     0     5     1

     1     1     4     4     6     6     0     0     9
     5     5     5     5     4     4     0     0     5
     1     1     1     1     1     1     0     0     1

Flujo actual
    19     1     0     0     0     0     0     0     9
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     4     0     0     0    19     0     0     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    15

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     4

     0     0     0     0     0     0     0     4     0
```

```
ITERACIÓN 5
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf     2
   Inf   Inf     0   Inf   Inf     3   Inf   Inf     1
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     5     2
   Inf     0   Inf   Inf     0     0     1     1     1
   Inf   Inf   Inf     4   Inf   Inf     0     5     2
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     2
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     4     1
     2     4     1
     3     4     1
     9     4     1
     5     4     1
     2     4     1
     0     0     0
     0     0     0
     0     4     1

     1     1     4     4     6     6     0     0     9
     4     4     4     4     4     4     0     0     4
     1     1     1     1     1     1     0     0     1

ITERACIÓN 6
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf     2
   Inf   Inf     0   Inf   Inf     3   Inf   Inf     1
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     4     2
   Inf     0   Inf   Inf     0     0     0     0     1
   Inf   Inf   Inf     5   Inf   Inf     0     5     3
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     3
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     4     1
     2     4     1
     3     4     1
     9     4     1
     5     4     1
     2     4     1
     7     4     1
     8     2     1
     0     4     1

     1     1     4     4     6     6     6     6     9
     4     4     4     4     4     4     4     2     4
     1     1     1     1     1     1     1     1     1
```

```
Flujo actual
    19     3     0     0     0     0     0     0     7
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    19     0     0     0     0
     0     2     0     0     0    19     0     2     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    17

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     2

     0     0     0     0     0     0     0     2     0
ITERACIÓN 7
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf     2
   Inf   Inf     0   Inf   Inf     3   Inf   Inf     1
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     4     2
   Inf     0   Inf   Inf     0     0     0     0     1
   Inf   Inf   Inf     5   Inf   Inf     0     5     3
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     3
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0
Marcaje del algoritmo FF
     9     2     1
     2     2     1
     3     2     1
     9     2     1
     5     2     1
     2     2     1
     7     2     1
     0     0     0
     0     2     1

     1     1     4     4     6     6     6     6     9
     2     2     2     2     2     2     2     0     2
     1     1     1     1     1     1     1     0     1
```

```
ITERACIÓN 8
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf     0
   Inf     0     3   Inf     4   Inf   Inf   Inf     2
   Inf   Inf     0   Inf   Inf     3   Inf   Inf     1
   Inf   Inf     0     0   Inf   Inf   Inf   Inf     0
   Inf   Inf   Inf   Inf     0   Inf   Inf     0     2
   Inf     0   Inf   Inf     0     0     0    -4     1
   Inf   Inf   Inf     5   Inf   Inf     0     1     3
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     7
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     9     2     1
     2     2     1
     3     2     1
     9     2     1
     5     2     1
     2     2     1
     7     2     1
     8     2     1
     0     2     1

     1     1     4     4     6     6     6     5     9
     2     2     2     2     2     2     2     2     2
     1     1     1     1     1     1     1     1     1
```

Finalmente, se obtiene el óptimo:

```
flujo =

    19     5     0     0     0     0     0     0     5
     0    19     0     0     0     0     0     0     0
     0     0    19     0     0     0     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    17     0     0     2     0
     0     0     0     0     2    19     0     2     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    19
```

Y deshaciendo los cambios de variable con la función “postproc”:

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

```
flujoreal =

    19     8     0     0     0     0     0     0     5
     0    19     0     0     1     0     0     0     0
     0     0    19     0     0     1     0     0     0
     0     0     0    19     0     0     0     0     5
     0     0     0     0    17     0     0     2     0
     0     0     0     0     3    19     0     3     0
     0     0     0     0     0     0    19     0     0
     0     0     0     0     0     0     0    19     0
     0     0     0     0     0     0     0     0    19


Kreales =

     8    -7     1     0    -2     5     0    -5     0


z =

    42
```

MATLAB también devuelve el siguiente grafo:

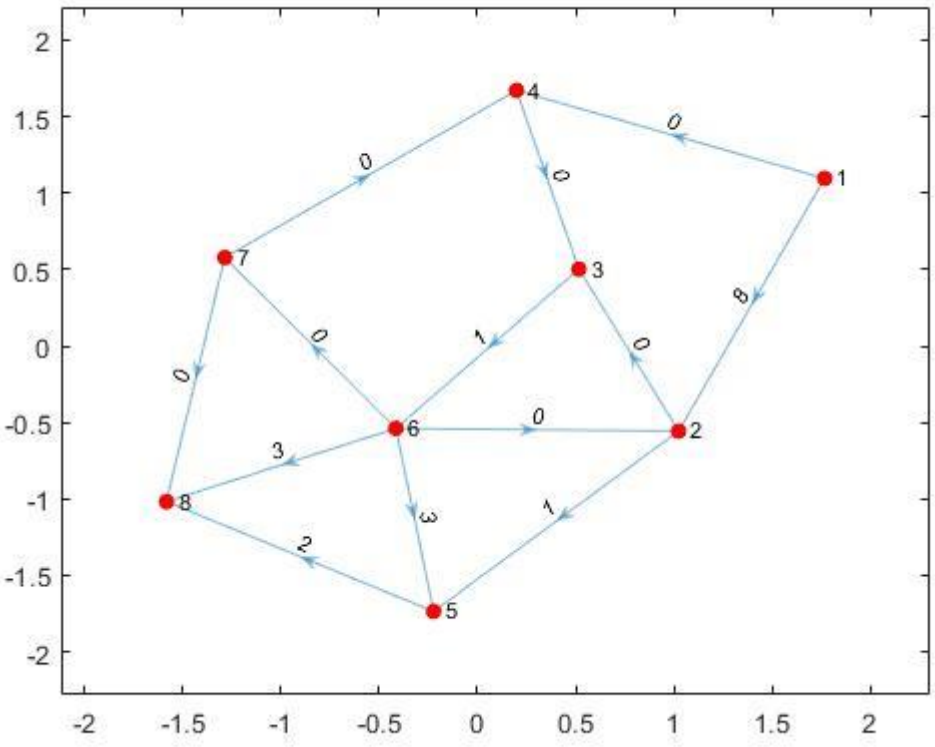


Figura 20.- Flujo óptimo en la red de la Figura 13 (método 2).

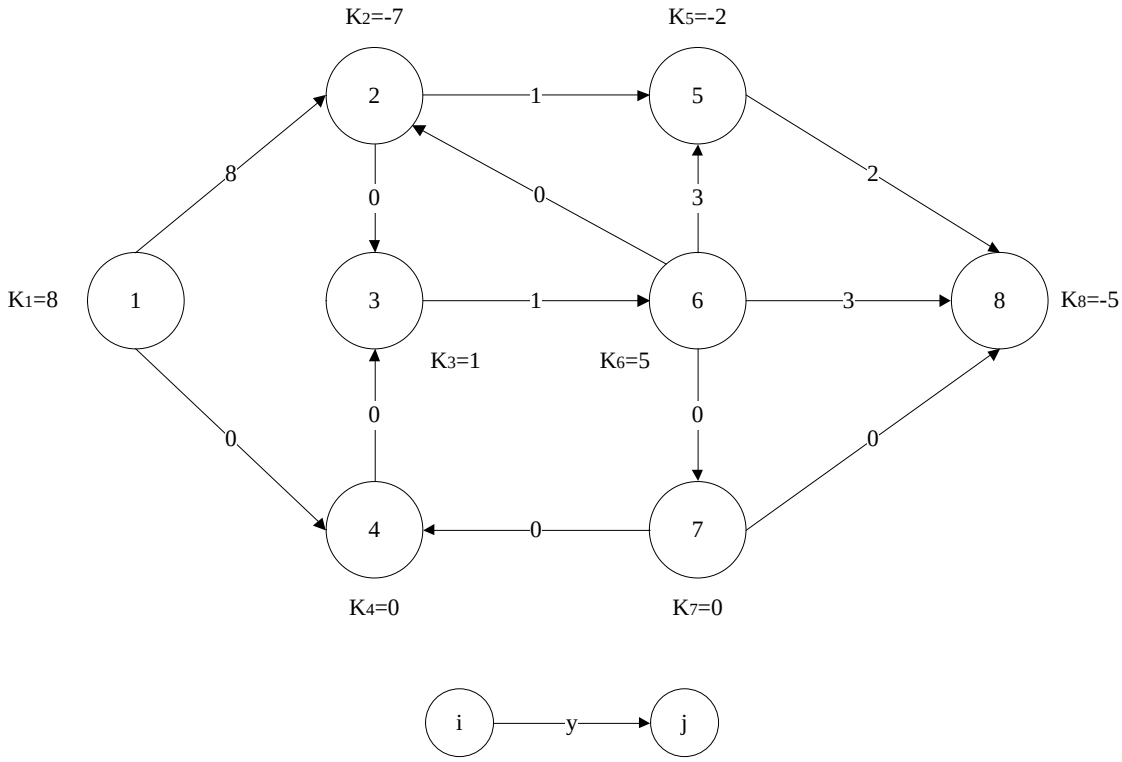


Figura 21.- Grafo realista equivalente al grafo de la Figura 20.

Puede comprobarse, con una simple comparación entre la Figura 17 y la Figura 20, que los óptimos obtenidos son idénticos como cabría esperar, al no haber en la red costes negativos.

2.6.3 Resolución del problema estandarizando la red con el método 3.

Una vez más, al ejecutar:

```
arcos=[1 2 3 16 2;1 4 0 10 3;2 3 0 5 2;2 5 1 5 4;3 6 1 3 3;4 3 0 5 1;5 8 0
5 5;6 2 0 4 1;6 5 1 5 1;6 7 0 6 2;6 8 1 3 2;7 4 0 5 2;7 8 0 4 5];
K=[13 -7 1 5 -2 5 0 -5];
[U,C,of,dem]=estandarizar(arcos,K,3)
```

Se realiza el cambio de variable asociado a eliminar las cotas inferiores, se calcula el vector “k” y se redefine el grafo de la Figura 13, de acuerdo al método 3.

En este caso, se añade un nodo adicional de salida S unido a todos los nodos de oferta mediante arcos ficiticios con capacidad igual a k_j y coste nulo; y un nodo adicional de entrada E unido a todos los nodos de demanda mediante arcos ficiticio con capacidad igual a $|k_j|$ y coste nulo.

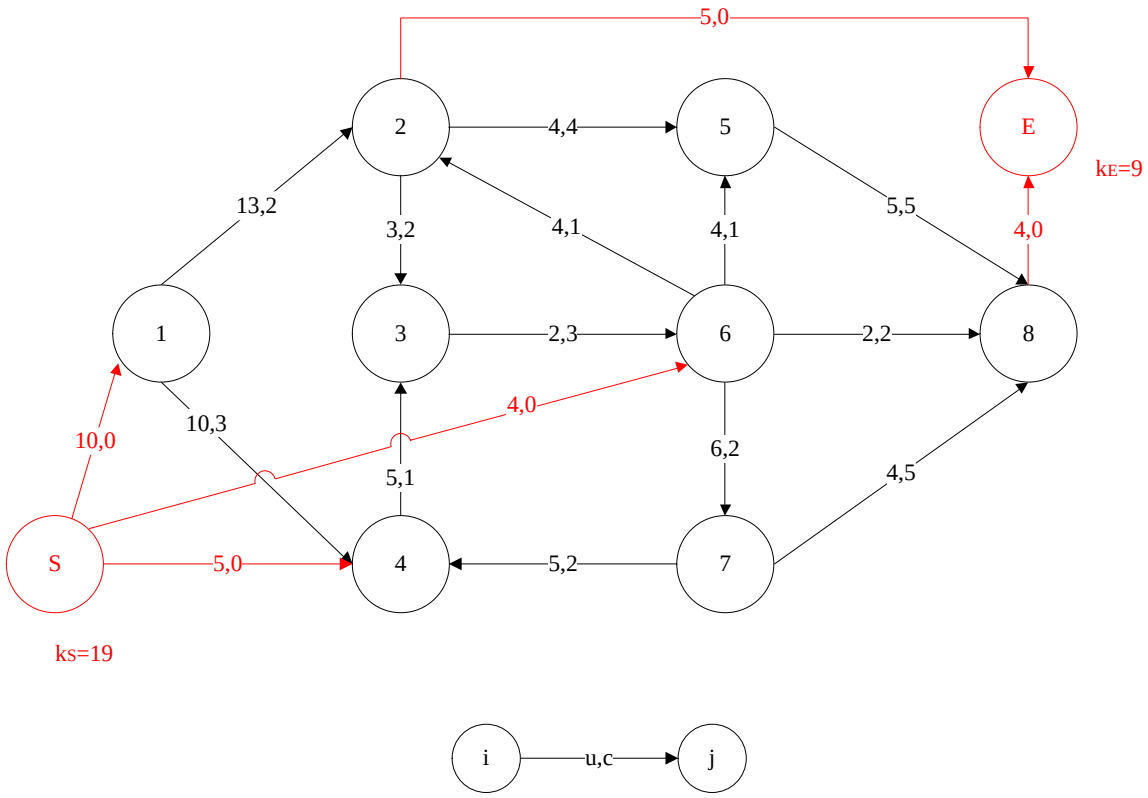


Figura 22.- Grafo de la Figura 13, estanadarizado mediante el método 3.

La ejecución de la función “estandarizar” proporciona los siguientes resultados:

```
U =

   Inf    13   Inf    10   Inf   Inf   Inf   Inf   Inf
   Inf   Inf     5   Inf     4   Inf   Inf   Inf     5
   Inf   Inf   Inf   Inf   Inf     2   Inf   Inf   Inf
   Inf   Inf     5   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     5   Inf
   Inf     4   Inf   Inf     4   Inf     6     2   Inf
   Inf   Inf   Inf     5   Inf   Inf   Inf     4   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf     4
    10   Inf   Inf     5   Inf     4   Inf   Inf   Inf


C =

     0     2   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     2   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     1     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     1   Inf   Inf     1     0     2     2   Inf
   Inf   Inf   Inf     2   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf     0   Inf   Inf   Inf
```

```
of =

     9     9     9     9     9     9     9     9    18


dem =

     9     9     9     9     9     9     9     9    18
```

Se muestran a continuación, sin comentar, los resultados intermedios que se obtienen al ejecutar la función "pdtci".

```
[flujo,mr]=pdtci(C,of,dem,U)


ITERACIÓN 1
Costos relativos
     0     2   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     2   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     1     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     1   Inf   Inf     1     0     2     2   Inf
   Inf   Inf   Inf     2   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf     0   Inf   Inf   Inf
Flujo actual
     9     0     0     0     0     0     0     0     0
     0     9     0     0     0     0     0     0     0
     0     0     9     0     0     0     0     0     0
     0     0     0     9     0     0     0     0     0
     0     0     0     0     9     0     0     0     0
     0     0     0     0     0     9     0     0     0
     0     0     0     0     0     0     9     0     0
     0     0     0     0     0     0     0     9     0
     0     0     0     0     0     0     0     0     0
Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
    18

     0     0     0     0     0     0     0     0    18
```

```
Marcaje del algoritmo FF
     1     9     1
     0     0     0
     0     0     0
     4     5     1
     0     0     0
     6     4     1
     0     0     0
     0     0     0
     0    18     1

     9     0     0     9     0     9     0     0     0
    10     0     0     5     0     4     0     0     0
     1     0     0     1     0     1     0     0     0


ITERACIÓN 2
Costos relativos
     0     1   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     2   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     4   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1   Inf
   Inf   Inf   Inf     3   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf     0   Inf   Inf   Inf
Marcaje del algoritmo FF
     1     9     1
     2     4     1
     3     5     1
     4     5     1
     5     4     1
     6     4     1
     0     0     0
     0     0     0
     0    18     1

     9     6     4     9     6     9     0     0     2
    10     4     5     5     4     4     0     0     4
     1     1     1     1     1     1     0     0     1
```

```
Flujo actual
     9     0     0     0     0     0     0     0     0
     0     5     0     0     0     0     0     0     4
     0     0     9     0     0     0     0     0     0
     0     0     0     9     0     0     0     0     0
     0     0     0     0     9     0     0     0     0
     0     4     0     0     0     5     0     0     0
     0     0     0     0     0     0     9     0     0
     0     0     0     0     0     0     0     9     0
     0     0     0     0     0     4     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
    14

     0     0     0     0     0     0     0     0    14


ITERACIÓN 3
Costos relativos
     0     1   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     2   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     4   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1   Inf
   Inf   Inf   Inf     3   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf     0   Inf   Inf   Inf


Marcaje del algoritmo FF
     1     9     1
     0     0     0
     3     5     1
     4     5     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0    14     1

     9     0     4     9     0     9     0     0     0
    10     0     5     5     0     0     0     0     0
     1     0     1     1     0     0     0     0     0
```

```
ITERACIÓN 4
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     3   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1   Inf
   Inf   Inf   Inf     4   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf    -1   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     9     1
     2     5     1
     3     5     1
     4     5     1
     5     4     1
     2     4     1
     0     0     0
     0     0     0
     0    14     1

     9     1     4     9     6     6     0     0     2
    10     9     5     5     4     4     0     0     1
     1     1     1     1     1     1     0     0     1

Flujo actual
     8     1     0     0     0     0     0     0     0
     0     4     0     0     0     0     0     0     5
     0     0     9     0     0     0     0     0     0
     0     0     0     9     0     0     0     0     0
     0     0     0     0     9     0     0     0     0
     0     4     0     0     0     5     0     0     0
     0     0     0     0     0     0     9     0     0
     0     0     0     0     0     0     0     9     0
     1     0     0     0     0     4     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
    13

     0     0     0     0     0     0     0     0    13
```

```
ITERACIÓN 5
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     3   Inf     4   Inf   Inf   Inf     0
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     5   Inf
   Inf     0   Inf   Inf     0     0     1     1   Inf
   Inf   Inf   Inf     4   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf    -1   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     8     1
     2     4     1
     3     5     1
     4     5     1
     5     4     1
     2     4     1
     0     0     0
     0     0     0
     0    13     1

     9     1     4     9     6     6     0     0     2
     9     8     5     5     4     4     0     0     0
     1     1     1     1     1     1     0     0     0
ITERACIÓN 6
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     3   Inf     4   Inf   Inf   Inf    -1
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     4   Inf
   Inf     0   Inf   Inf     0     0     0     0   Inf
   Inf   Inf   Inf     5   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf    -1   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     8     1
     2     4     1
     3     5     1
     4     5     1
     5     4     1
     2     4     1
     7     4     1
     8     2     1
     0    13     1

     9     1     4     9     6     6     6     6     8
     9     8     5     5     4     4     4     2     2
     1     1     1     1     1     1     1     1     1
```

```
Flujo actual
     6     3     0     0     0     0     0     0     0
     0     4     0     0     0     0     0     0     5
     0     0     9     0     0     0     0     0     0
     0     0     0     9     0     0     0     0     0
     0     0     0     0     9     0     0     0     0
     0     2     0     0     0     5     0     2     0
     0     0     0     0     0     0     9     0     0
     0     0     0     0     0     0     0     7     2
     3     0     0     0     0     4     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
    11

     0     0     0     0     0     0     0     0    11


ITERACIÓN 7
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     3   Inf     4   Inf   Inf   Inf    -1
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     4   Inf
   Inf     0   Inf   Inf     0     0     0     0   Inf
   Inf   Inf   Inf     5   Inf   Inf     0     5   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf    -1   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     6     1
     2     4     1
     3     5     1
     4     5     1
     5     2     1
     2     2     1
     7     2     1
     0     0     0
     0    11     1

     9     1     4     9     6     6     6     6     0
     7     6     5     5     2     2     2     0     0
     1     1     1     1     1     1     1     0     0
```

```
ITERACIÓN 8
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     3   Inf     4   Inf   Inf   Inf    -5
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     0   Inf
   Inf     0   Inf   Inf     0     0     0    -4   Inf
   Inf   Inf   Inf     5   Inf   Inf     0     1   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf    -1   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     6     1
     2     4     1
     3     5     1
     4     5     1
     5     2     1
     2     2     1
     7     2     1
     8     2     1
     0    11     1

     9     1     4     9     6     6     6     5     8
     7     6     5     5     2     2     2     2     2
     1     1     1     1     1     1     1     1     1

Flujo actual
     4     5     0     0     0     0     0     0     0
     0     4     0     0     0     0     0     0     5
     0     0     9     0     0     0     0     0     0
     0     0     0     9     0     0     0     0     0
     0     0     0     0     7     0     0     2     0
     0     0     0     0     2     5     0     2     0
     0     0     0     0     0     0     9     0     0
     0     0     0     0     0     0     0     5     4
     5     0     0     0     0     4     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
     9

     0     0     0     0     0     0     0     0     9
```

```
ITERACIÓN 9
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     3   Inf     4   Inf   Inf   Inf    -5
   Inf   Inf     0   Inf   Inf     3   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     0   Inf
   Inf     0   Inf   Inf     0     0     0    -4   Inf
   Inf   Inf   Inf     5   Inf   Inf     0     1   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf    -1   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     4     1
     2     4     1
     3     5     1
     4     5     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     9     1

     9     1     4     9     0     0     0     0     0
     5     4     5     5     0     0     0     0     0
     1     1     1     1     0     0     0     0     0

ITERACIÓN 10
Costos relativos
     0     0   Inf     3   Inf   Inf   Inf   Inf   Inf
   Inf     0     3   Inf     1   Inf   Inf   Inf    -8
   Inf   Inf     0   Inf   Inf     0   Inf   Inf   Inf
   Inf   Inf     0     0   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf   Inf     0   Inf
   Inf     3   Inf   Inf     0     0     0    -4   Inf
   Inf   Inf   Inf     8   Inf   Inf     0     1   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf     0   Inf    -4   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     4     1
     2     4     1
     3     5     1
     4     5     1
     5     2     1
     6     2     1
     7     2     1
     8     2     1
     0     9     1

     9     1     4     9     6     3     6     5     8
     5     4     5     5     2     2     2     2     0
     1     1     1     1     1     1     1     1     0
```

Obteniendo finalmente el óptimo siguiente:

```
flujo =

     4     5     0     0     0     0     0     0     0
     0     4     0     0     0     0     0     0     5
     0     0     9     0     0     0     0     0     0
     0     0     0     9     0     0     0     0     0
     0     0     0     0     7     0     0     2     0
     0     0     0     0     2     5     0     2     0
     0     0     0     0     0     0     9     0     0
     0     0     0     0     0     0     0     5     4
     5     0     0     0     0     4     0     0     0
```

Para finalizar el algoritmo, se ejecuta la función “postproc”.

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

Se obtiene el óptimo de la red original, así como los niveles de oferta y demanda que realmente se satisfacen en la red y el valor de la función objetivo.

```
flujoreal =

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


Kreales =

     8    -7     1     0    -2     5     0    -5     0


z =

    42
```

MATLAB también devuelve el siguiente grafo, en el que se indican los flujos óptimos para la red del problema propuesto:

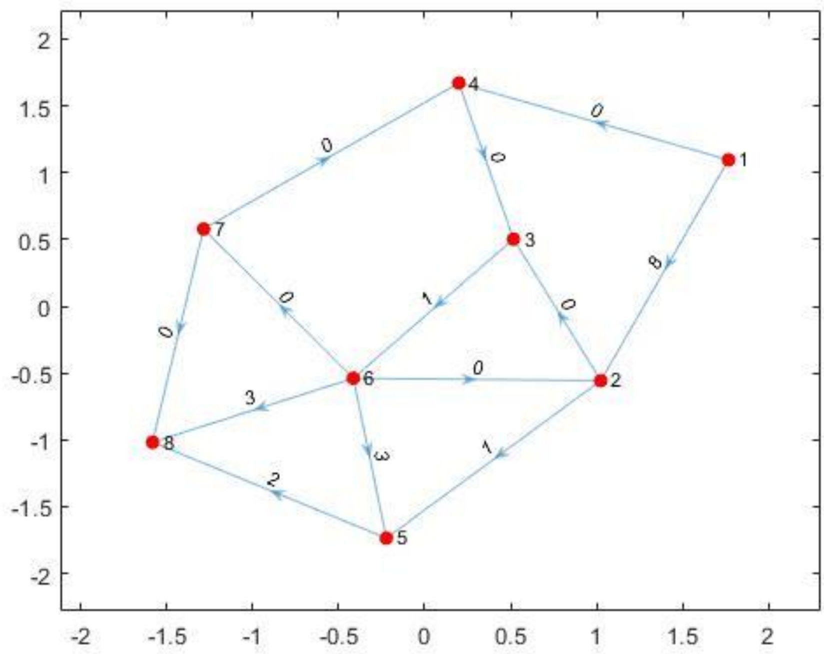


Figura 23.- Flujo óptimo en la red de la Figura 13 (método 3).

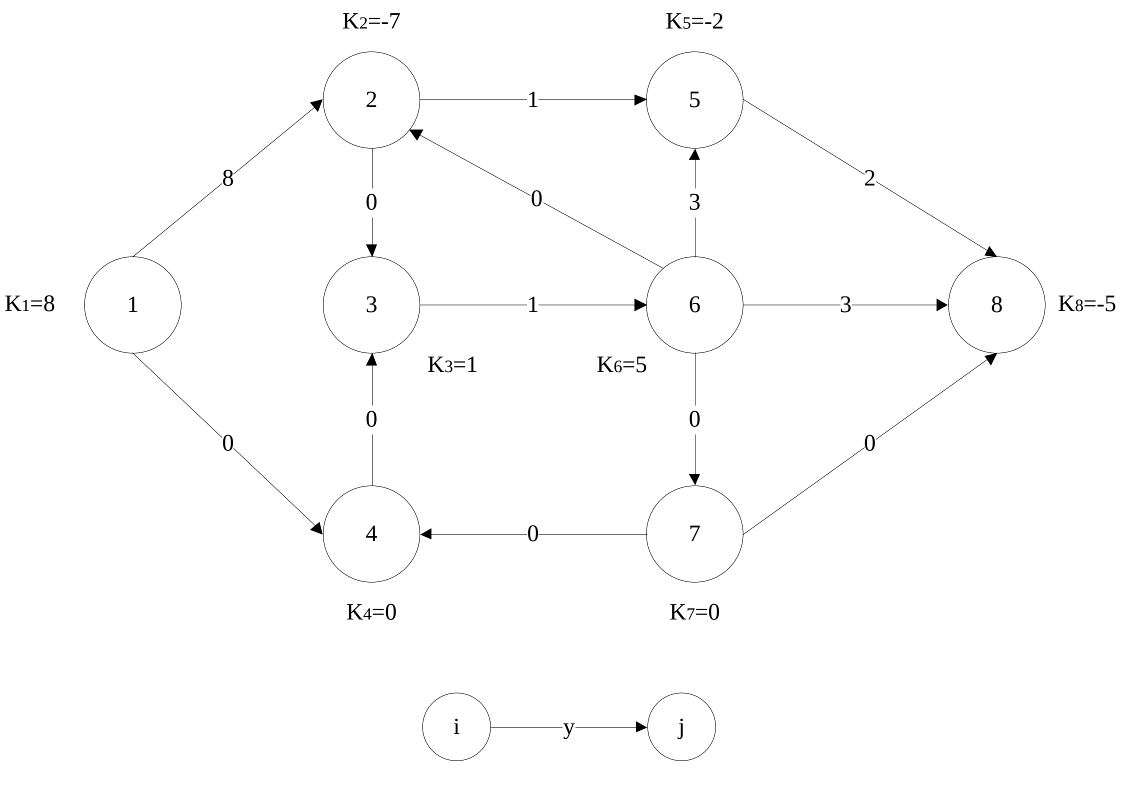


Figura 24.- Grafo realista equivalente al grafo de la Figura 23.

Como es de esperar, el óptimo obtenido en los tres métodos es exactamente el mismo.

3 ALGORITMO PRIMAL-DUAL CON COSTOS DEPENDIENTES DEL FLUJO

Una vez resuelta la implementación en MATLAB de la resolución del problema de flujo máximo a coste mínimo (con costes constantes), el siguiente paso es adaptar el algoritmo PDTCI para que se ejecute iterativamente y permita la resolución del problema de flujo máximo a coste mínimo con costes dependientes del flujo.

Este nuevo algoritmo, se denominará algoritmo Primal-Dual aplicado a la Tabla de transporte con Costes Dependientes del flujo (algoritmo PDTCD).

La formulación matermática del problema viene dada por la ecuación (2):

$$\begin{aligned} & Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}(y_{ij}) \cdot y_{ij} \\ & s.a. \sum_{k\in D(j)} y_{jk} - \sum_{i\in A(j)} y_{ij} = K_j \quad \forall j = 1,2,\dots,n \\ & L_{ij} \le y_{ij} \le U_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{2}$$

Con

$$c_{ij}(y_{ij}) = \begin{cases} f(y_{ij}) & L_{ij} \le y_{ij} \le U_{ij}^1 \\ g(y_{ij}) & U_{ij}^1 \le y_{ij} \le U_{ij}^2 \\ h(y_{ij}) & U_{ij}^2 \le y_{ij} \le U_{ij}^3 \end{cases} \tag{17}$$

Esta formulación deberá estandarizarse de manera análoga al problema con costos constantes, eliminando las cotas inferiores y equilibrando ofertas y demandas en la red (ver Apartado 3.3).

En este caso, para resolver el problema, se emplean los denominados costos marginales de cada arco en lugar de los costos absolutos. El coste marginal de un determinado arco (i,j), ∂z_{ij}, representa el incremento de coste en la función objetivo *z* al aumentar una unidad de flujo en el arco (i,j). Matemáticamente:

$$\partial z_{ij} = \frac{\partial z}{\partial x_{ij}} = \frac{\partial\left[c_{ij}(x_{ij}) \cdot x_{ij}\right]}{\partial x_{ij}} = c_{ij}(x_{ij}) + \frac{\partial c_{ij}(x_{ij})}{\partial x_{ij}} \cdot x_{ij} \tag{18}$$

Se exige que la función de costos absolutos c_{ij} para los arcos con costes dependientes del flujo, sea continua, estrictamente convexa y creciente. De manera general podrá expresarse como:

$$c_{ij}(x_{ij}) = \begin{cases} a & l \le x_{ij} \le u_1 \\ a + b(x_{ij} - u_1) + c(x_{ij} - u_1)^2 + d(x_{ij} - u_1)^3 + \cdots & u_1 \le x_{ij} \le u_2 \\ e + f(x_{ij} - u_2) + g(x_{ij} - u_2)^2 + h(x_{ij} - u_2)^3 + \cdots & u_2 \le x_{ij} \le u_3 \\ \cdots & \cdots \end{cases} \quad (19)$$

Destacar que, en esta expresión, el valor del coeficiente "*e*" no puede ser aleatorio, sino que debe cumplir la condición de continuidad. Para ello, tomando la notación de la expresión (17), debe cumplirse que:

$$g(u_2) = h(u_2) \rightarrow a + b(u_2 - u_1) + c(u_2 - u_1)^2 + d(u_2 - u_1)^3 + \cdots = e \quad (20)$$

Esta condición de continuidad debe cumplirse para todos los tramos del polinomio de costes absolutos.

Como es lógico, para los arcos (i,j) con costes constantes, el coste marginal coincide con el coste absoluto.

El algoritmo PDTCD resolverá sucesivos problemas en el ámbito de los costes constantes (algoritmo PDTCI), tomando para ello los costes marginales de todos los arcos. Los incrementos de flujo entre los sucesivos problemas serán de unidad en unidad.

A la vista de la ecuación (19) (expresión general de los costes absolutos no constantes de los arcos (i,j)), inicialmente el coste marginal de dicho arco tendrá un valor constante de "a" unidades mientras el flujo sea menor o igual a u_1 unidades. Dicho valor u_1, recibirá el nombre de capacidad modificada del arco (i,j). Con esos valores, el problema puede resolverse como si los costes fueran constantes.

Cuando el arco (i,j) alcance un flujo igual a u_1 unidades, es decir, se sature, se incrementará dicha capacidad modificada en una unidad, lo que permitirá el incremento de flujo en la red en una unidad. Se seguirá sucesivamente aumentando de manera unitaria esta capacidad modificada hasta alcanzar su capacidad superior. Lógicamente, para los arcos (i,j) con costes constantes, su capacidad modificada coincide con su capacidad superior.

De manera resumida, los pasos del algoritmo PDTCD son los siguientes:

1. Construir la tabla de transbordo asociada a la red que se va a resolver, y a continuación, estandarizar dicha tabla, realizando los cambios de variable pertinentes para convertir las capacidades inferiores, L_{ij}, en cero, y equilibrando y orlando ofertas y demandas para permitir la asignación de flujos.
2. Calcular la matriz de costes marginales (ver Apartado 3.2) y de capacidades modificadas
3. Calcular la matriz reducida de la matriz de costes marginales, para obtener las celdas básicas y asignar flujos en dichas celdas de la tabla, si es posible.
4. Realizar el proceso de marcaje del algoritmo Ford-Fulkerson y aumentar flujos o modificar la matriz reducida según se obtenga o no un nodo de entrada.
5. Cuando un arco con coste dependiente del flujo haya saturado su capacidad modificada, la matriz reducida no puede modificarse. En lugar de ello, se debe recalcular el coste marginal de dicho arco, y aumentar en 1 unidad su capacidad modificada (si no se ha alcanzado ya la capacidad superior). Volver al paso 3, partiendo de los flujos obtenidos al resolver el problema previo.
6. El óptimo se alcanza, bien cuando se hayan saturado todas las ofertas y demandas de todos los nodos de la red o bien cuando no pueda modificarse la matriz reducida.

Respecto a la implementación en MATLAB, las funciones que permiten resolver el problema de flujo máximo a coste mínimo dependiente del flujo según el algoritmo PDTCD son las siguientes:

```
[U,C,of,dem,arcos]=estandarizarcd(arcosci,nodoscd,CDA,K,metodo)
[CDM]=calculacostemarginal(CDA)
[flujo]=pdtcd(C,of,dem,U,nodoscd,CDM)
[flujoreal,Kreales,z]=postproccd(flujo,arcos,C,nodoscd,CDA)
```

Como puede observarse, estas líneas de código son análogas (aunque con ciertas diferencias) a las líneas de código correspondientes al algoritmo PDTCI expuestas en el capítulo anterior.

Introduciendo los datos de la red en las variables "arcosci"," nodoscd", "CDA" y "K", y copiando en la ventana de comandos de MATLAB las líneas de código anteriores, se puede obtener el flujo máximo a coste mínimo que transcurre por la red, así como los niveles reales de oferta y demanda de cada nodo de la red y el valor de la función objetivo global.

En esto, ya puede intuirse una ligera diferencia entre algoritmos a la hora de introducir los parámetros de la red en MATLAB. A lo largo del Capítulo, se explica el significado de las distintas variables de entrada que deben definirse, así como cada una de las funciones y subrutinas que componen el código MATLAB del algoritmo PDTCD implementado, y sus modificaciones y/o diferencias con respecto al algoritmo PDTCI.

3.1. Introducción de datos de la red.

Sea una red de transporte representada por un grafo G de *n* nodos y *l* arcos, con ofertas/demandas K_j para cada nodo j, y capacidades inferiores L_{ij}, capacidades superiores U_{ij} y costes c_{ij} para cada arco (i,j), la información de los arcos de la red se introducirá en MATLAB definiendo las variables "arcosci"," nodoscd", "CDA" y "K", cuyo contenido se describe a continuación:

- Matriz "arcosci"

  La información relativa a los arcos con costos constantes de la red se transformará en una matriz de 5 columnas en las que se guardará la información de manera análoga a la matriz "arcos" descrita en el Apartado 2.1, y cuya estructura es:

$$arcosci = \begin{bmatrix} nodo\ origen & nodo\ destino & capacidad\ inferior & capacidad\ superior & coste \\ i & j & L_{ij} & U_{ij} & c_{ij} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad (21)$$

  Puede demostrarse matemáticamente que, para estos arcos, el coste absoluto c_{ij} coincide con el coste marginal (ver Apartado 3.2). Esto será de utilidad a la hora de calcular la matriz de costes marginales.

- Matriz "nodoscd"

  Esta matriz recoge los nodos origen y destino de los arcos con costes dependientes del flujo. Por tanto, será una matriz de tantas filas como arcos con costes no constantes tenga la red y 2 columnas.

$$nodoscd = \begin{bmatrix} nodo\ origen & nodo\ destino \\ i & j \\ \vdots & \vdots \end{bmatrix} \qquad (22)$$

- Estructura "CDA"

  La función de costes absolutos para un arco con coste dependiente del flujo, se introduce en MATLAB como un polinomio a trozos, usando la función "mkpp" (Make Piecewise Polynomial). Para ello, la función de costes debe estar escrita como indica la expresión (19)

$$c_{ij}(x_{ij}) = \begin{cases} a & l \le x_{ij} \le u_1 \\ a + b(x_{ij} - u_1) + c(x_{ij} - u_1)^2 + d(x_{ij} - u_1)^3 + \cdots & u_1 \le x_{ij} \le u_2 \\ e + f(x_{ij} - u_2) + g(x_{ij} - u_2)^2 + h(x_{ij} - u_2)^3 + \cdots & u_2 \le x_{ij} \le u_3 \\ \cdots & \cdots \end{cases} \qquad (19)$$

  La sintaxis en MATLAB para la función mkpp será:

```
CIJ=mkpp(breaks,coefs);
```

  Siendo "breaks" el vector de capacidades de cada tramo, y "coefs" la matriz de coeficientes del polinomio, ordenados de mayor a menor grado, es decir:

$$breaks = [l \quad u_1 \quad u_2 \quad u_3 \quad ...] \qquad (23)$$

$$coefs = \begin{bmatrix} 0 & 0 & 0 & 0 & a \\ ... & d & c & b & a \\ ... & h & g & f & e \\ ... & ... & ... & ... & ... \end{bmatrix} \qquad (24)$$

  Finalmente, "CDA" será un vector que englobe las distintas estructuras "CIJ" para todos los arcos de la red con costes no constantes. Es de crucial importancia en la ejecución del programa, ordenar las estructuras que componen "CDA" en el mismo orden en el que se introducen los arcos en las filas de "nodoscd" para que la función "estandarizarcd" sea capaz de calcular la matriz de costes marginales de manera correcta.

- Vector "K"

  Nuevamente, los niveles de oferta y demanda K_j asociados a cada nodo se recogen en un vector de *n* elementos.

$$K = [K_1 \quad K_2 \quad \dots \quad K_n] \qquad (25)$$

Ejemplo.- *Sea la red de transporte representada con el siguiente grafo:*

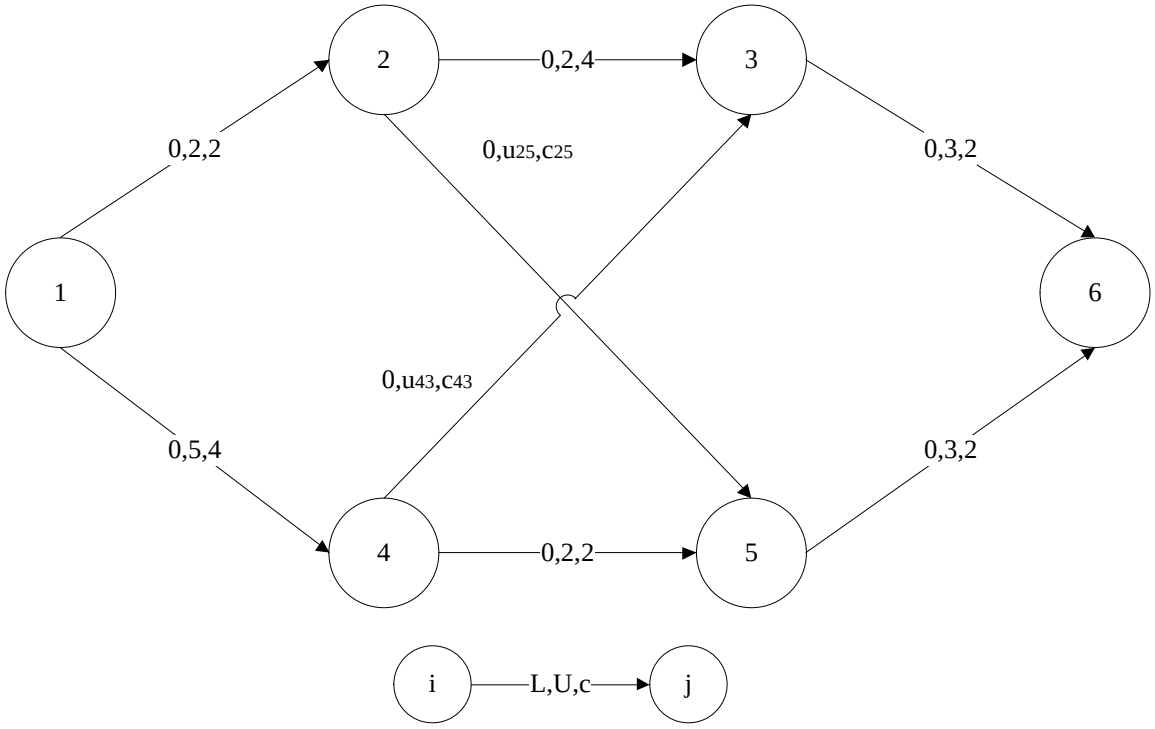


Figura 25.- Ejemplo de red de transporte con arcos con costes dependientes del flujo.

Los costos dependientes del flujo son los siguientes

$$c_{25}(x_{25}) = \begin{cases} 3 & 0 \le x_{25} \le 1 \\ 3 + 1.5(x_{25} - 1) & 1 \le x_{25} \le 3 \end{cases}$$

$$c_{43}(x_{43}) = \begin{cases} 1 & 0 \le x_{43} \le 2 \\ 1 + (x_{43} - 2)^2 & 2 \le x_{43} \le 4 \end{cases}$$

La matriz "arcosci" (21) recogerá la información de los arcos con costes constantes:

$$arcosci = \begin{bmatrix} 1 & 2 & 0 & 2 & 2 \\ 1 & 4 & 0 & 5 & 4 \\ 2 & 3 & 0 & 2 & 4 \\ 3 & 6 & 0 & 3 & 2 \\ 4 & 5 & 0 & 2 & 2 \\ 5 & 6 & 0 & 3 & 2 \end{bmatrix}$$

En la red de la Figura 25, el nodo 1 es el único nodo de salida, que se considerará como único nodo de oferta, pudiendo ofertar 7 unidades de flujo (suma de las capacidades de los arcos (1,2) y (1,4)). Análogamente, puede observarse que el nodo 6 es el único nodo de demanda, pudiendo demandar hasta 6 unidades de flujo. Al no indicar nada más, se entiende que el resto de nodos son de transbordo.

Con todo ello, el vector "K" (25) será:

$$K = [7 \quad 0 \quad 0 \quad 0 \quad 0 \quad -6]$$

Respecto a los costes dependientes del flujo, el primer paso es crear la matriz "nodoscd" (22)

$$nodoscd = \begin{bmatrix} 2 & 5 \\ 4 & 3 \end{bmatrix}$$

Y a continuación, crear las estructuras "C25" y "C43", que serán los polinomios a trozos anteriores. Así, por ejemplo, para el arco (4,3), "breaks" (23) y "coefs" (24) serán:

$$C43.breaks = [0 \quad 2 \quad 4]$$

$$C43.coefs = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Las estructuras "C25", "C43" y finalmente, "CDA", se introducirán en MATLAB mediante las siguientes líneas de código:

```
C25=mkpp([0 1 3],[0 3;1.5 3]);
C43=mkpp([0 2 4],[0 0 1;1 0 1]);
CDA=[C25,C43];
```

3.2. Cálculo de costes marginales.

La diferencia esencial entre los algoritmos PDTCI y PDTCD, es que este último emplea la matriz de costos marginales de todos los arcos en cada iteración, para el cálculo de la red básica. Por esta razón, y como etapa previa al cálculo algorítmico propiamente dicho, es necesario calcular esos costes marginales.

Sea $c_{ij}(x_{ij})$ el coste absoluto de un arco (i,j), este puede expresarse, de manera general, según la expresión (19):

$$c_{ij}(x_{ij}) = \begin{cases} a & l \le x_{ij} \le u_1 \\ a + b(x_{ij} - u_1) + c(x_{ij} - u_1)^2 + d(x_{ij} - u_1)^3 + \cdots & u_1 \le x_{ij} \le u_2 \\ e + f(x_{ij} - u_2) + g(x_{ij} - u_2)^2 + h(x_{ij} - u_2)^3 + \cdots & u_2 \le x_{ij} \le u_3 \\ \cdots & \cdots \end{cases} \qquad (19)$$

Su coste marginal, ∂z_{ij}, se calcula a partir de la expresión (18):

$$\partial z_{ij} = \frac{\partial z}{\partial x_{ij}} = \frac{\partial \left[c_{ij}(x_{ij}) \cdot x_{ij}\right]}{\partial x_{ij}} \qquad (18)$$

Para el primer tramo de $c_{ij}(x_{ij})$

$$\partial z_{ij,1} = \frac{\partial \left[c_{ij}(x_{ij})_1 \cdot x_{ij}\right]}{\partial x_{ij}} = \frac{\partial (a x_{ij})}{\partial x_{ij}} = a$$

Con lo que queda demostrado, que, para los arcos con costes constantes, su coste marginal coincide con su coste absoluto.

Para el segundo tramo de $c_{ij}(x_{ij})$ (y análogamente para el resto de tramos), tomando polinomios de grado 3, derivando, desarrollando los distintos términos del polinomio y simplificando:

$$\partial z_{ij,2} = \frac{\partial \left[c_{ij}(x_{ij})_2 \cdot x_{ij}\right]}{\partial x_{ij}} = \frac{\partial (a x_{ij} + b x_{ij}(x_{ij} - u_1) + c x_{ij}(x_{ij} - u_1)^2 + d x_{ij}(x_{ij} - u_1)^3)}{\partial x_{ij}} =$$
$$= a + b x_{ij} + b(x_{ij} - u_1) + 2c x_{ij}(x_{ij} - u_1) + c(x_{ij} - u_1)^2 + 3d x_{ij}(x_{ij} - u_1)^2 + d(x_{ij} - u_1)^3 =$$
$$= a + b x_{ij} + b x_{ij} - b u_1 + 2c x_{ij}^2 - 2c x_{ij} u_1 + c x_{ij}^2 - 2c x_{ij} u_1 + c u_1^2 + 3d x_{ij}^3 - 6d x_{ij}^2 u_1 +$$
$$+ 3d x_{ij} u_1^2 + d x_{ij}^3 - 3d x_{ij}^2 u_1 + 3d x_{ij} u_1^2 - d u_1^3 =$$
$$= (a - b u_1 + c u_1^2 + d u_1^3) + (2b - 4c u_1 + 6d u_1^2) x_{ij} + (3c - 9d u_1) x_{ij}^2 + 4d x_{ij}^3$$

Se obtiene,

$$\partial z_{ij,2} = (a - b u_1 + c u_1^2 + d u_1^3) + (2b - 4c u_1 + 6d u_1^2) x_{ij} + (3c - 9d u_1) x_{ij}^2 + 4d x_{ij}^3 \qquad (26)$$

Para permitir su implementación en MATLAB, el coste marginal debe estar expresado de manera análoga a la expresión anterior del coste absoluto (19), es decir:

$$\partial z_{ij} = \begin{cases} a & l \le x_{ij} \le u_1 \\ A + B(x_{ij} - u_1) + C(x_{ij} - u_1)^2 + D(x_{ij} - u_1)^3 + \cdots & u_1 < x_{ij} \le u_2 \\ E + F(x_{ij} - u_2) + G(x_{ij} - u_2)^2 + H(x_{ij} - u_2)^3 + \cdots & u_2 < x_{ij} \le u_3 \\ \cdots & \cdots \end{cases} \quad (27)$$

Destáquese que la función de costes marginales no tiene por qué ser continua, lo que se refleja en los signos de menor estricto que aparecen en la expresión (27).

En conclusión, debe haber una relación entre los coeficientes a, b, c, d… de la función de costes absolutos y los coeficientes A, B, C, D… de la función de costes marginales.

Por tanto, para determinar los coeficientes A, B, C y D, se desarrolla el polinomio $\partial z_{ij,2}$ en la expresión (27). Tomando nuevamente el polinomio de grado 3:

$$\begin{aligned} &\partial z_{ij,2} = A + B(x_{ij} - u_1) + C(x_{ij} - u_1)^2 + D(x_{ij} - u_1)^3 = \\ &= A + Bx_{ij} - Bu_1 + Cx_{ij}^2 - 2Cx_{ij}u_1 + Cu_1^2 + Dx_{ij}^3 - 3Dx_{ij}^2u_1 + 3Dx_{ij}u_1^2 - Du_1^3 = \\ &= (A - Bu_1 + Cu_1^2 + Du_1^3) + (B - 2cu_1 + 3Du_1^2)x_{ij} + (C - 3Du_1)x_{ij}^2 + Dx_{ij}^3 \end{aligned}$$

Es decir:

$$\partial z_{ij,2} = (A - Bu_1 + Cu_1^2 + Du_1^3) + (B - 2cu_1 + 3Du_1^2)x_{ij} + (C - 3Du_1)x_{ij}^2 + Dx_{ij}^3 \quad (28)$$

Como las expresiones (26) y (28) representan exactamente el mismo coste marginal, ambos polinomios deben ser coincidentes. En consecuencia, los coeficientes asociados a cada grado de x_{ij} son idénticos. De esta forma, se obtiene el siguiente sistema de ecuaciones:

$$\begin{cases} a - bu_1 + cu_1^2 - du_1^3 = A - Bu_1 + Cu_1^2 - Du_1^3 \\ 2b - 4cu_1 + 6du_1^2 = B - 2Cu_1 + 3Du_1^2 \\ 3c - 9du_1 = C - 3Du_1 \\ 4d = D \end{cases}$$

Dicho sistema de ecuaciones puede resolverse de manera escalonada como se indica a continuación:

$$D = 4d$$

$$\begin{aligned} &3c - 9u_1 = C - 3 \cdot 4d \cdot u_1 \\ &3c - 9u_1 = C - 12du_1 \\ &C = 3c - 9du_1 + 12du_1 = 3c + 3du_1 \\ &C = 3(c + du_1) \end{aligned}$$

$2b - 4cu_1 + 6du_1^2 = B - 2 \cdot 3(c + du_1)u_1 + 3 \cdot 4d \cdot u_1^2$
$2b - 4cu_1 + 6du_1^2 = B - 6cu_1 - 6du_1^2 + 12du_1^2$
$B = 2b - 4cu_1 + 6du_1 + 6cu_1 + 6du_1^2 - 12du_1^2 = 2b + 2cu_1$
$B = 2(b + cu_1)$

$a - bu_1 + cu_1^2 - du_1^3 = A - 2(b + cu_1)u_1 + 3(c + du_1)u_1^2 - 4du_1^3$
$a - bu_1 + cu_1^2 - du_1^3 = A - 2bu_1 - 2cu_1 + 3cu_1^2 + 3du_1^3 - 4du_1^3$
$A = a - bu_1 + cu_1^2 - du_1^3 + 2bu_1 + 2cu_1^2 - 3cu_1^2 - 3du_1^3 + 4du_1^3$
$A = a - bu_1$

Es decir, considerando polinomios de grado 3, los coeficientes de los polinomios de la función de coste marginal (27) pueden hallarse a partir de los coeficientes de los polinomios la función de coste absoluto (19) según las relaciones siguientes:

$$\begin{cases} A = a + bu_1 \\ B = 2(b + cu_1) \\ C = 3(c + du_1) \\ D = 4d \end{cases}$$

De donde, generalizando a polinomios de orden *n*, puede observarse que los coeficientes A, B, C, D siguen una sucesión cuyo término general es:

$$A_{kn} = (n+1)\left[a_{kn} + a_{k,n+1} \cdot u_{k-1}\right] \ \forall n = 0,1,2,3,\ldots \ \forall k = 1,2,3,\ldots \qquad (29)$$

Siendo A_{kn} el coeficiente del término de grado *n* en la expresión del tramo *k* del coste marginal, y a_{kn} el coeficiente del término de grado *n* en la expresión del tramo *k* del coste absoluto.

La expresión (29) es válida para todos los tramos de la función de coste absoluto de un arco (i,j), siempre que esta sea continua, creciente y estrictamente convexa, y todos los tramos sean funciones polinómicas.

De toda esta demostración matemática previa, puede deducirse que los elementos A_{ij} de la matriz de coeficientes de la función de coste marginal, pueden calcularse a partir de los elementos a_{ij} de la matriz de coeficientes de la función de coste absoluto, según la siguiente expresión.

$$A_{ij} = j \cdot (a_{ij} + a_{i,j+1} \cdot u_i) \qquad (30)$$

Nótese que, para cada fila del polinomio a trozos, se indican los coeficientes ordenados desde grado 0 hasta grado *n*. Para la implementación en MATLAB, la función "mkpp" exige que la matriz de coeficientes esté ordenada de manera inversa, desde el coeficiente del término de grado *n* hasta el de grado 0.

La función "calculacostemarginal" realizará el siguiente proceso de cálculo:

- Voltear la matriz de coeficientes de la función de costes absolutos, usando el comando "fliplr" para ordenar los coeficientes en orden creciente de grados.
- Incluir una columna de ceros a la derecha de esta matriz, que corresponderá a los coeficientes del grado *n+1*, lo que permitirá a MATLAB no tener errores de cálculo al emplear la ecuación (30) (ver Ejemplo).
- Calcular la matriz de coeficientes de la función de costes marginales a partir de la expresión (30).
- Voltear de nuevo la matriz recién calculada para ordenar los coeficientes en orden creciente de grados.
- Asignar esta matriz de coeficientes a la estructura "CDM", que será la que guarde los costes marginales de los arcos con costos dependientes del flujo.

Además, la función "calculacostemarginal" también efectúa el cambio de variable asociado a anular la capacidad inferior del arco.

El código es el siguiente:

```
function [CDM]=calculacostemarginal(CDA)
    n=length(CDA);
    for ii=1:n
        Lij=CDA(ii).breaks(1);
        CMbreaks=CDA(ii).breaks-Lij;
        a=fliplr(CDA(ii).coefs);
        [ntramos,ncoefs]=size(a);
        a(:,ncoefs+1)=0;
        A=zeros(ntramos,ncoefs);
        for jj=1:ntramos
           for kk=1:ncoefs
               A(jj,kk)=kk*(a(jj,kk)+a(jj,kk+1)*CMbreaks(jj));
           end
        end
        CMcoefs=fliplr(A);
        CDM(ii)=mkpp(CMbreaks,CMcoefs);
    end
```

Ejemplo.- *Siguiendo con el grafo de la Figura 25, la función "calculacostemarginal" determina la expresión del coste marginal de los arcos (2,5) y (4,3) a partir de la estructura de costes absolutos "CDA".*

Tomando, por ejemplo, el arco (4,3), cuyo coste absoluto es:

$$c_{43}(x_{43}) = \begin{cases} 1 & 0 \le x_{43} \le 2 \\ 1 + (x_{43} - 2)^2 & 2 \le x_{43} \le 4 \end{cases}$$

Siendo (23) (24)

$$C43.coefs = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

$$C43.breaks = [0 \quad 2 \quad 4]$$

En primer lugar, se realiza el cambio de variable para eliminar la cota inferior. Como en este caso, dicha cota inferior ya es cero, no es necesario realizar el cambio de variable.

Luego, se calcula la matriz "a", volteando las columnas de "C43.coefs", es decir:

$$a = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Aquí, si se aplicase la expresión (30) para determinar A_{23}, MATLAB daría un error pues el coeficiente a_{24} no existe.

$$A_{ij} = j \cdot \left(a_{ij} + a_{i,j+1} \cdot u_i\right) \qquad (30)$$

Para solucionar esto, se añade una columna de ceros a la derecha de "a". En este caso, representaría al coeficiente de grado 3 en la expresión del coste absoluto del arco (4,3), que obviamente es nulo.

Por tanto,

$$a = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

y

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 4 & 3 \end{bmatrix}$$

Nótese que, según el código programado, MATLAB creará una matriz "A" de las mismas dimensiones que la matriz "a", ignorando la columna adicional añadida para solventar el error de cálculo.

Por último, se debe voltear de nuevo esta matriz "A" para obtener "CMcoefs"

$$CMcoefs = \begin{bmatrix} 0 & 0 & 1 \\ 3 & 4 & 1 \end{bmatrix}$$

Esta información ("CMcoefs" y "CMbreaks") se emplea para crear el polinomio a trozos correspondiente a la función de costes marginales con la función "mkpp".

Finalmente, el coste marginal para el arco (4,3) será

$$\partial z_{43} = \begin{cases} 1 & 0 \le x_{43} \le 2 \\ 1 + 4(x_{43} - 2) + 3(x_{43} - 2)^2 & 2 < x_{43} \le 4 \end{cases}$$

3.3. Estandarización del problema.

La primera etapa a la hora de resolver el problema de flujo máximo a coste mínimo consiste en realizar los cambios de variable necesarios para estandarizar el problema, así como equilibrar ofertas y demandas mediante nodos adicionales o añadiendo un nodo único de salida y otro de entrada.

El procedimiento de cálculo que se realiza para obtener dichas matrices y la red estandarizada es completamente idéntico al ya explicado para el algoritmo PDTCI (función "estandarizar", ver Apartado 2.2).

```
function [U,C,of,dem,arcos]=estandarizarcd(arcosci,nodoscd,CDA,K,metodo)
```

Respecto a la estandarización de la función de costes absolutos dependientes del flujo (17),

$$c_{ij}(y_{ij}) = \begin{cases} f(y_{ij}) & L_{ij} \le y_{ij} \le U_{ij}^1 \\ g(y_{ij}) & U_{ij}^1 \le y_{ij} \le U_{ij}^2 \\ h(y_{ij}) & U_{ij}^2 \le y_{ij} \le U_{ij}^3 \end{cases} \qquad (17)$$

realizando el cambio de variable $x_{ij} = y_{ij} - L_{ij}$, resulta:

$$c_{ij}(y_{ij}) = c_{ij}(x_{ij} + L_{ij}) = \begin{cases} f(x_{ij} + L_{ij}) & 0 \le x_{ij} \le u_{ij}^1 \\ g(x_{ij} + L_{ij}) & u_{ij}^1 \le x_{ij} \le u_{ij}^2 \\ h(x_{ij} + L_{ij}) & u_{ij}^2 \le x_{ij} \le u_{ij}^3 \end{cases} \qquad (31)$$

Donde $u_{ij}^k \equiv U_{ij}^k - L_{ij}$, siendo *k* el superíndice correspondiente a cada función f, g o h.

La función "estandarizarcd" recibe como datos las variables "arcosci", "nodoscd", "CDA" y "K" explicadas previamente, así como la variable "metodo", que indica el método a emplear para crear la red de transbordo, y valdrá 1, 2 ó 3, según sea el método elegido.

Una vez se han introducido todos los datos de la red en MATLAB, la función "estandarizarcd" realiza esa transformación del problema a una red estándar, así como el cálculo de la matriz de costes marginales y de capacidades modificadas para la primera iteración del algoritmo PDTCD. También devolverá por pantalla el grafo de la red estandarizada.

La única diferencia entre ambas funciones está en la obtención de la matriz "arcos", que es la matriz que recoge toda la información de todos los arcos de la red, ya tengan costes constantes o dependientes del flujo. Esta matriz se empleará en la función de post-procesado de los resultados.

Recuérdese que en el caso del algoritmo PDTCI, la matriz "arcos" directamente era un dato de entrada al programa. En el caso del algoritmo PDTCD, la función "estandarizarcd" realiza una serie de cálculos sencillos para generar esa matriz a partir de los datos de entrada, mediante el siguiente fragmento de código:

```
[ncd,~]=size(nodoscd);
arcoscd=zeros(ncd,5);
for ii=1:ncd
    arcoscd(ii,:)=[nodoscd(ii,:) CDA(ii).breaks(1) CDA(ii).breaks(2)…
       …polyval(CDA(ii).coefs(1,:),CDA(ii).breaks(2))];
end
arcos=[arcosci;arcoscd];
```

Lo que hace este fragmento de código es calcular una matriz denominada "arcoscd", con información análoga a la matriz "arcosci" pero para los arcos con coste dependientes del flujo. Para ello, se emplea la información de las matrices "nodoscd" y la estructura de costes "CDA".

Ejemplo.- *Para la red de la Figura 25, cuyos parámetros de entrada son las matrices "arcosci" (21), "nodoscd" (22), "K" (25)*

$$arcosci = \begin{bmatrix} 1 & 2 & 0 & 2 & 2 \\ 1 & 4 & 0 & 5 & 4 \\ 2 & 3 & 0 & 2 & 4 \\ 3 & 6 & 0 & 3 & 2 \\ 4 & 5 & 0 & 2 & 2 \\ 5 & 6 & 0 & 3 & 2 \end{bmatrix}$$

$$K = [7 \quad 0 \quad 0 \quad 0 \quad 0 \quad -6] \qquad nodoscd = \begin{bmatrix} 2 & 5 \\ 4 & 3 \end{bmatrix}$$

y la estructura CDA, que contendrá la información de los costes absolutos de los arcos (2,5) y (4,3), arcos con costes no constantes.

$$C25.breaks = [0 \quad 1 \quad 3] \qquad C43.breaks = [0 \quad 2 \quad 4]$$

$$C25.coefs = \begin{bmatrix} 0 & 3 \\ 1.5 & 3 \end{bmatrix} \qquad C43.coefs = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

"arcoscd" será, en este caso, una matriz con 2 filas (pues hay dos arcos con costes dependientes del flujo) y 5 columnas. La información que recogerán estas columnas será, por orden: nodo origen, nodo destino, capacidad inferior, capacidad superior modificada y coste marginal para dicha capacidad modificada.

```
for ii=1:ncd
   arcoscd(ii,:)=[nodoscd(ii,:) CDA(ii).breaks(1) CDA(ii).breaks(2)…
      …polyval(CDA(ii).coefs(1,:),CDA(ii).breaks(2))];
end
```

Para ii=1 (arco (2,5)), se tiene:

Los nodos origen y destino se obtienen desde la matriz "nodoscd". En este caso, el nodo origen es el 2 y el nodo destino el 5.

La capacidad inferior corresponde al primer elemento del vector "C25.breaks". En este caso vale 0.

La capacidad superior modificada corresponde al segundo elemento de "C25.breaks". En este caso vale 1 (Esa capacidad modificada se aumentará de unidad en unidad a lo largo del proceso algorítmico hasta alcanzar como máximo el valor 3, correspondiente al último elemento de "C25.breaks").

El coste marginal del arco (2,5) para una capacidad modificada de 1 coincide con su coste absoluto para el primer tramo (que, en las condiciones impuestas para la expresión de los costes absolutos, siempre será constante), como se ha demostrado en el Apartado 3.2. En MATLAB, este valor se determina evaluando el polinomio "C25" (de costes absolutos) en el tramo 1, mediante la función "polyval". En este caso, dicho coste marginal vale 3.

Procediendo de manera análoga para ii=2 (arco 4,3)), se obtiene la matriz "arcoscd":

$$arcoscd = \begin{bmatrix} 2 & 5 & 0 & 1 & 3 \\ 4 & 3 & 0 & 2 & 1 \end{bmatrix}$$

La matriz "arcos" será la unión de las matrices "arcosci" y "arcoscd" en una única matriz. Destacar que, como se ha indicado previamente, para los arcos con costes constantes, su coste marginal es constante y coincide con su coste absoluto, y su capacidad modificada es directamente su capacidad superior. Por ello, la matriz "arcos" indica para todos los arcos de la red los costes marginales y las capacidades modificadas.

$$arcos = \begin{bmatrix} 1 & 2 & 0 & 2 & 2 \\ 1 & 4 & 0 & 5 & 4 \\ 2 & 3 & 0 & 2 & 4 \\ 3 & 6 & 0 & 3 & 2 \\ 4 & 5 & 0 & 2 & 2 \\ 5 & 6 & 0 & 3 & 2 \\ 2 & 5 & 0 & 1 & 3 \\ 4 & 3 & 0 & 2 & 1 \end{bmatrix}$$

El resto de cálculos que realiza la función "estandarizarcd" son completamente idénticos al caso del problema con costes constantes, y permite la obtención de la matriz de costes marginales y de capacidades modificadas.

Eligiendo, por ejemplo, el método 3 (nodos adicionales de entrada y salida) para el equilibrado de la red:

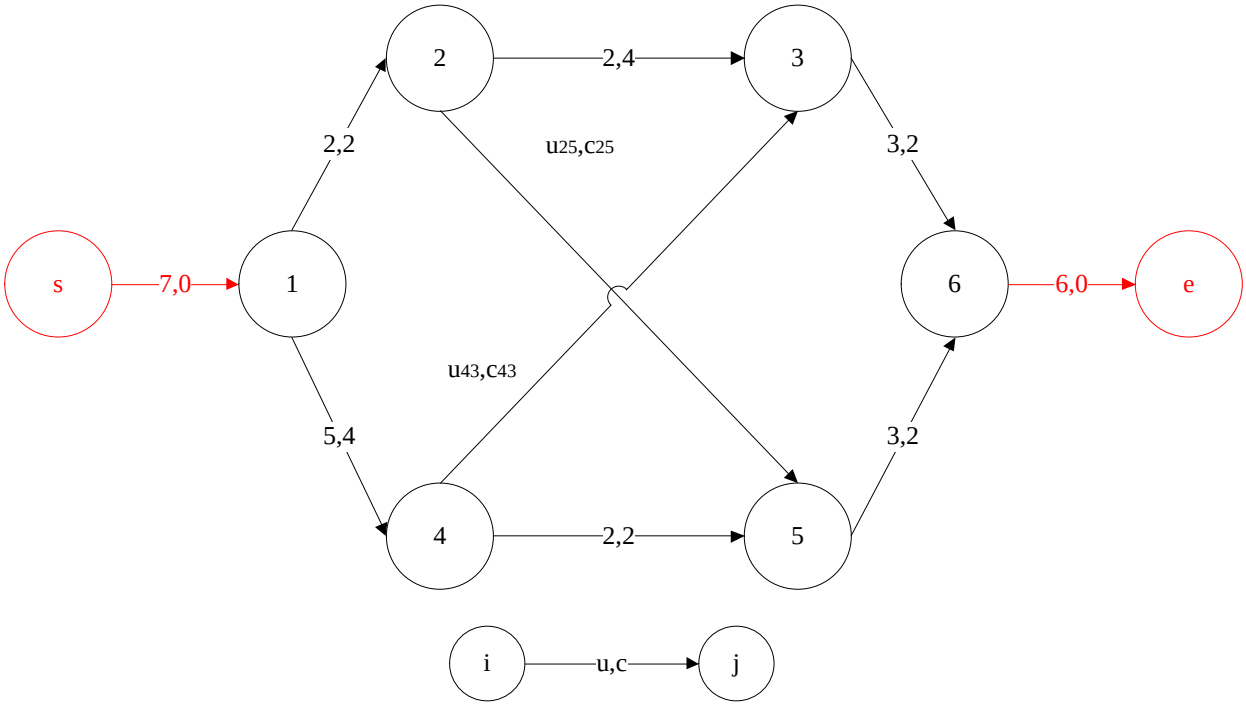


Figura 26.- Red de transporte de la Figura 25, estandarizada según el método 3.

Ejecutando:

```
[U,C,of,dem,arcos]=estandarizarcd(arcosci,nodoscd,CDA,K,3)
```

Se obtiene las matrices y vectores siguientes:

```
U =                                                C =

   Inf     2   Inf     5   Inf   Inf   Inf     0     2   Inf     4   Inf   Inf   Inf
   Inf   Inf     2   Inf     1   Inf   Inf   Inf     0     4   Inf     3   Inf   Inf
   Inf   Inf   Inf   Inf   Inf     3   Inf   Inf   Inf     0   Inf   Inf     2   Inf
   Inf   Inf     2   Inf     2   Inf   Inf   Inf   Inf     1     0     2   Inf   Inf
   Inf   Inf   Inf   Inf   Inf     3   Inf   Inf   Inf   Inf   Inf     0     2   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     6   Inf   Inf   Inf   Inf   Inf     0     0
     7   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf   Inf   Inf   Inf   Inf   Inf
                                                   arcos =
of =
                                                       1     2     0     2     2
     6     6     6     6     6     6    12            1     4     0     5     4
                                                       2     3     0     2     4
                                                       3     6     0     3     2
                                                       4     5     0     2     2
dem =                                                  5     6     0     3     2
                                                       2     5     0     1     3
     6     6     6     6     6     6    12            4     3     0     2     1
```

Así como el siguiente grafo orientado, en el que en los arcos se indican los costes marginales:

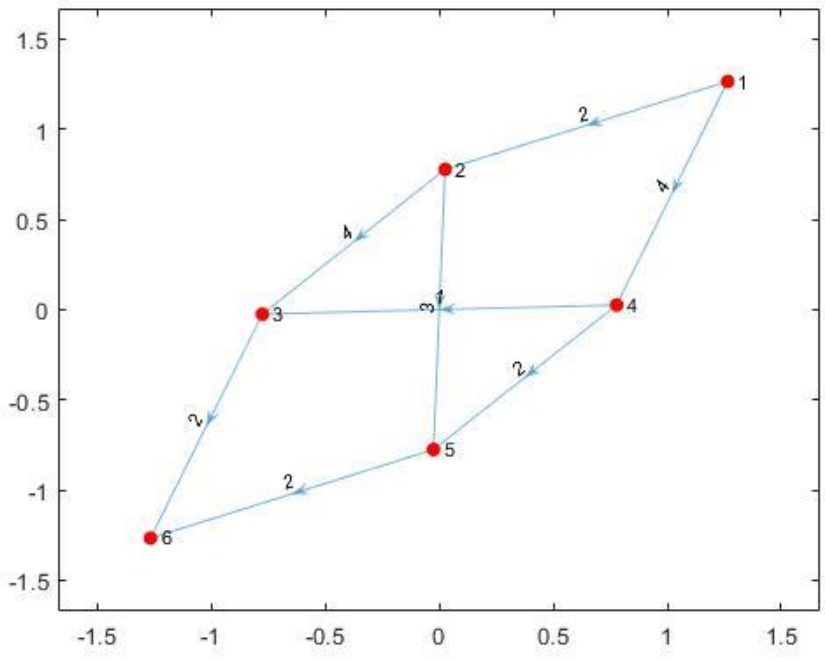


Figura 27.- Grafo de la Figura 25, representado en MATLAB.

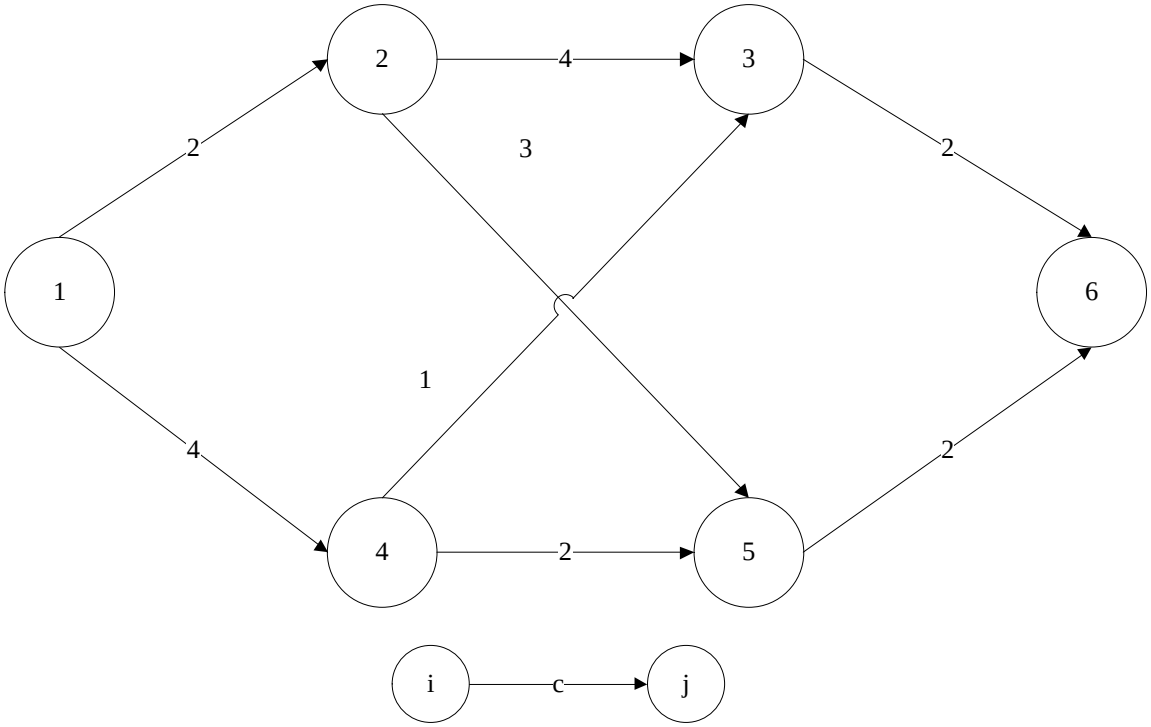


Figura 28.- Grafo realista equivalente al grafo de la Figura 27.

Nótese que el grafo que devuelve la función "estandarizarcd" presenta los costes marginales de cada arco con los que se inicia la primera iteración del algoritmo PDTCD. Es decir, para los arcos con coste no constante, indica el valor de coste marginal correspondiente al primer tramo de dicha función de costes.

3.4. Algoritmo Primal-Dual para costes no constantes.

Con los resultados que ofrece la función "estandarizarcd", la red original ya está transformada a una red estándar que puede resolverse aplicando el algortimo PDTCD. Para ello, en MATLAB, se ha programado la función "pdtcd", cuyo código, funcionamiento y cálculos se indican a continuación.

```
function [flujo]=pdtcd(C,of,dem,U,nodoscd,CDM)
[m,n]=size(C);
flujo=zeros(m,n);
while true
    mr=matred(C);
    while true
        [flujo,of,dem]=asignaflujo(mr,flujo,of,dem,U);
        if sum(of)==0, break; end
        while true
            [I,J,ent]=marcajeff(mr,flujo,of,dem,U);
            [modCU,ii]=compruebasaturacd(flujo,U,nodoscd,CDM);
            if ent~=0
                [flujo,of,dem]=aumentaflujo(flujo,of,dem,I,J,ent);
                if sum(of)==0, break; end
            else
                if modCU==0
                    [mr,modmr]=modificaMR(mr,I,J,flujo,U);
                    break
                else
                    [C,U]=nuevaCU(U,C,nodoscd,ii,CDM);
                    break
                end
            end
        end
        if modmr==0, break; end
        if modCU==1, break; end
        if sum(of)==0, break; end
    end
    if sum(of)==0, break; end
    if modmr==0, break; end
end
```

El programa se compone de tres bucles "while" anidados, en las que las variables de decisión limitan cuándo se sale de cada uno de ellos.

El bucle "while" más externo realiza el cálculo algorítmico propiamente dicho. El algoritmo comienza calculando la matriz reducida de la matriz de costos marginales (Apartado 2.3.1), con la que se establece una red básica desde la que empezar los cálculos.

A partir de aquí, los cálculos son totalmente análogos al algoritmo PDTCI. Se realiza la asignación directa de flujos (Apartado 2.3.2), si fuera posible.

A continuación, se procede con el marcaje del algoritmo Ford-Fulkerson (Apartado 2.3.3). Como primera diferencia entre ambos algoritmos, se implementa la función "compruebasaturacd", que devolverá una variable de decisión "modCU" igual a 1, en caso de que algún arco con coste dependiente del flujo haya alcanzado un flujo igual a su capacidad modificada actual, es decir, se haya saturado; e igual a 0, en caso contrario.

Por tanto, en función de los valores de las variables de decisión "ent" y modCU", se abren 3 posibilidades en el algoritmo.

- Si, tras el marcaje, se ha alcanzado un nodo de demanda, $\mathrm{ent} \neq 0$, entonces se puede aumentar el flujo en la red básica (Apartado 2.3.5). El algoritmo continúa realizando de nuevo el marcaje de la tabla.
- Si, no habiéndose alcanzado un nodo de demanda, $\mathrm{ent} = 0$, no hay arcos con costes dependientes del flujo saturados, $\mathrm{modCU} = 0$, entonces puede realizarse la modificación de la matriz reducida de costes marginales (Apartado 2.3.4), obteniendo nuevas celdas básicas sobre las que puede (o no) asignarse flujos de manera directa.
- Si, no habiéndose alcanzado un nodo de demanda, $\mathrm{ent} = 0$, hay un arco (con coste dependiente del flujo) saturado, $\mathrm{modCU} = 1$, la matriz reducida de los costes no puede modificarse más. Entonces, se aumenta la capacidad modificada de dicho arco en una unidad y se recalcula el coste marginal de dicho arco. Para ello, se implementa la función "nuevaCU". A partir de las nuevas matrices "C" y "U", se reinicia el algoritmo determinando una nueva red básica (cálculo de la matriz reducida), manteniendo los flujos anteriormente obtenidos.

El algoritmo alcanzará el óptimo si se da alguno de los siguientes casos:

- Si tras asignar flujos de manera directa o tras aumentar los flujos según el algoritmo Ford-Fulkerson, se han satisfecho todas las ofertas y demandas, $\mathrm{sum(of)} = 0$.
- Si tras intentar realizar la modificación de la matriz reducida, ésta no pueda modificarse, al no encontrar un r* real ($\mathrm{ent} = 0, \mathrm{modCU} = 0, \mathrm{modmr} = 0$).

En la siguiente tabla se indican, de manera descriptiva, cada una de las funciones o subrutinas que forman parte de la función "pdtcd", la mayoría de las cuales ya han sido desarrolladas en el Capítulo 2.

Tabla 17.- Subrutinas de la función "pdtcd".

FUNCIÓN "pdtcd"	
matred	Determinación de los arcos básicos de la red estandarizada
asignaflujo	Asignación directa de flujos sobre la tabla de transporte
marcajeff	Marcaje de los nodos según el algoritmo de Ford-Fulkerson
compruebasaturacd	Determinación de la presencia de algún arco con coste dependiente del flujo que esté saturado.
aumentaflujo	Aumento de flujos en la red básica según el algoritmo de Ford-Fulkerson
modificaMR	Modificación de la matriz reducida
nuevaCU	Recálculo de las matrices de costes marginales y de capacidades modificadas.

Puede observarse, que el algoritmo PDTCD emplea las mismas rutinas que el algoritmo PDTCI, con el añadido de las funciones "compruebasaturacd" y "nuevaCU", que serán explicadas más adelante este apartado.

Para facilitar la comprensión del funcionamiento de los bucles anidados de la función "pdtcd", así como de las variables de decisión, se indica el siguiente diagrama de flujo:

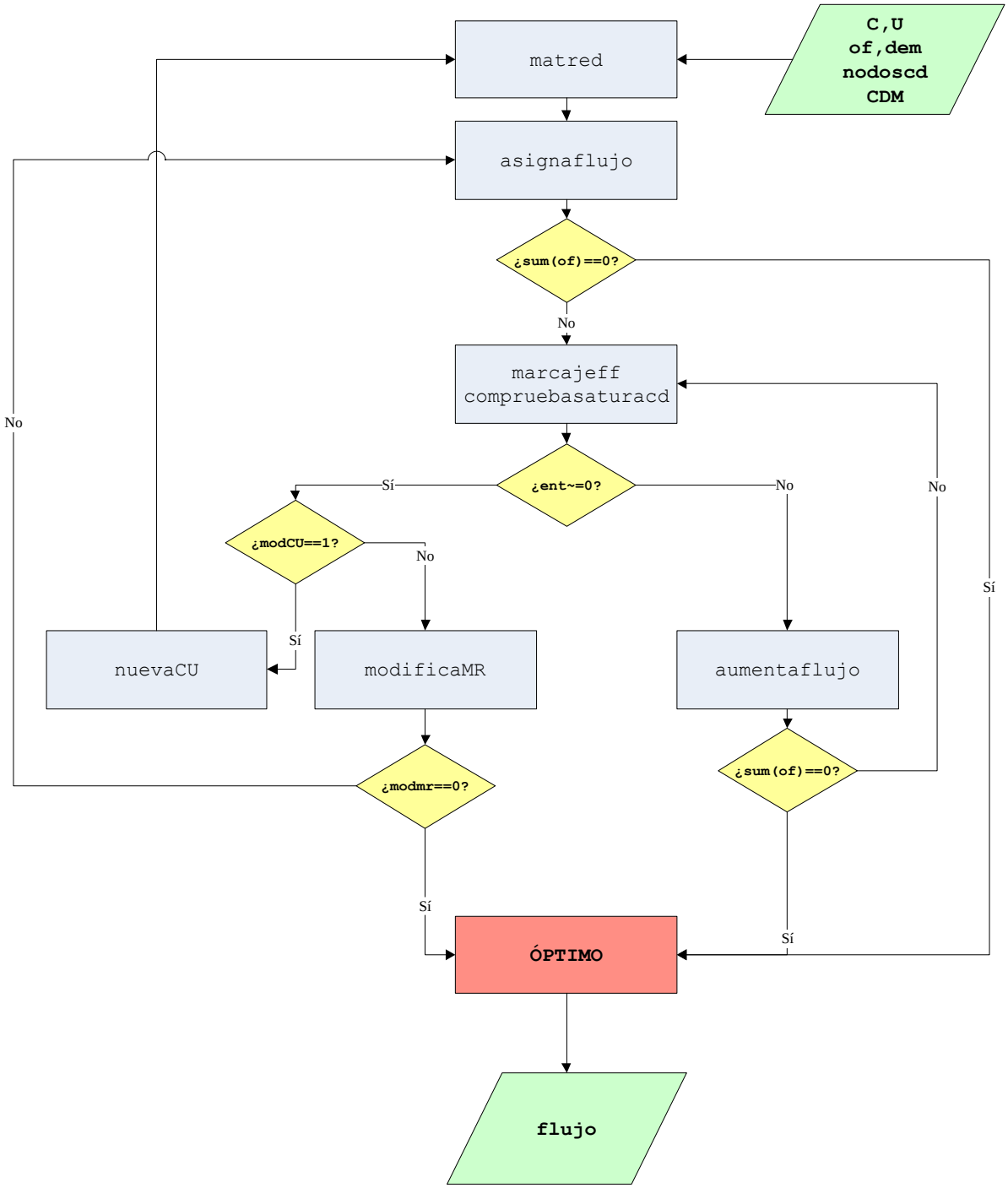


Figura 29.- Diagrama de flujo de la función "pdtcd".

3.4.1 Modificación de las matrices de costos marginales y capacidades superiores.

Como se ha indicado anteriormente, la diferencia esencial entre el algoritmo con costes constantes y costes no constantes está en el cálculo en cada iteración de la matriz de costes marginales y de capacidades modificadas. Esta modificación en ambas matrices se da una vez se ha comprobado que un arco con coste dependiente del flujo se ha saturado.

Para ello, se ha implementado dentro de la función "pdtcd" la subrutina "compruebasaturacd", cuyo código es:

```
function [modCU,ii]=compruebasaturacd(flujo,U,nodoscd,CDM)
[ncd,~]=size(nodoscd);
modCU=0;
for ii=1:ncd
    i=nodoscd(ii,1);
    j=nodoscd(ii,2);
    if flujo(i,j)==U(i,j) && U(i,j)<CDM(ii).breaks(end)
        modCU=1;
        break
    end
end
```

Sencillamente, esta función devuelve una variable de decisión, "modCU", que valdrá 1 cuando exista un arco con coste no constante cuyo flujo haya alcanzado su capacidad modificada, siempre que esta no sea igual a su capacidad máxima.

Además, guarda en la variable "ii", la fila de la matriz "nodoscd" que corresponde a la información del arco (i,j) con costes no constantes que se ha saturado.

Una vez saturado dicho arco (i,j), su capacidad se aumenta en una unidad y se recalcula su coste marginal, obteniéndose nuevas matrices "C" y "U", en la que solamente ha cambiado el valor del elemento (i,j). Para ese cálculo se implementa la función "nuevaCU".

```
function [C,U]=nuevaCU(U,C,nodoscd,ii,CDM)
    i=nodoscd(ii,1);
    j=nodoscd(ii,2);
    U(i,j)=U(i,j)+1;
    for kk=1:(length(CDM(ii).breaks)-1)
        if U(i,j)>CDM(ii).breaks(kk) && U(i,j)<=CDM(ii).breaks(kk+1)
            tramo=kk;
        end
    end
    C(i,j)=polyval(CDM(ii).coefs(tramo,:),U(i,j)-CDM(ii).breaks(kk));
end
```

Para evaluar el coste maginal del arco (i,j) para un flujo $x_{ij} = \mathrm{U}$, debe tenerse en cuenta una ligera diferencia existente entre la función a trozos que se define en MATLAB con la función "mkpp", y la función a trozos real que describe al coste marginal.

Al crear un polinomio a trozos con la función "mkpp", MATLAB interpreta por defecto que dicho polinomio tiene la siguiente expresión,

$$\begin{cases} a & l \le x_{ij} < u_1 \\ A + B(x_{ij} - u_1) + C(x_{ij} - u_1)^2 + D(x_{ij} - u_1)^3 + \cdots & u_1 \le x_{ij} < u_2 \\ E + F(x_{ij} - u_2) + G(x_{ij} - u_2)^2 + H(x_{ij} - u_2)^3 + \cdots & u_2 \le x_{ij} < u_3 \\ \cdots & \cdots \end{cases} \quad (32)$$

mientras que la expresión general del coste marginal es la ya indicada previamente:

$$\partial z_{ij} = \begin{cases} a & l \le x_{ij} \le u_1 \\ A + B(x_{ij} - u_1) + C(x_{ij} - u_1)^2 + D(x_{ij} - u_1)^3 + \cdots & u_1 < x_{ij} \le u_2 \\ E + F(x_{ij} - u_2) + G(x_{ij} - u_2)^2 + H(x_{ij} - u_2)^3 + \cdots & u_2 < x_{ij} \le u_3 \\ \cdots & \cdots \end{cases} \quad (27)$$

Obsérvese la diferencia entre las expresiones (27) y (32), que radica en el tratamiento de los intervalos. MATLAB interpreta dichos intervalos de una manera que no representa la realidad de la función de costes marginales.

La función "ppval" de MATLAB permite evaluar un determinado polinomio a trozos expresado como (32) en un cierto valor. Es fácil intuir un conflicto de programación a la hora de evaluar el coste marginal en los valores $u_1, u_2, u_3, \ldots$ de manera que, por ejemplo, para un flujo $x_{ij} = u_1$, se sabe que el coste marginal vale "a" (expresión (27)), mientras que la función "ppval" devuelve un valor de "A" para dicho coste marginal (como se ha demostrado en el Apartado 3.2, "A" y "a" no son valores coincidentes, en general).

Para solventar este conflicto a la hora de evaluar los costes marginales, pueden plantearse numerosas estrategias alternativas de programación. En esta Memoria, se opta por una estrategia en la que se determina en primer lugar el tramo del polinomio que se debe emplear para evaluar el nuevo coste marginal (para una mejor comprensión de esta estrategia de programación, se recomienda al lector consultar el ejemplo numérico de la siguiente página). Así, u_1 se evaluaría según el tramo 1 de la función a trozos que define el coste marginal del arco (i,j).

Luego, se usa la función "polyval" para calcular el coste marginal. Por defecto, "polyval" es una función que no admite polinomios a trozos, y que, además, entiende que el polinomio expresado por el vector de coeficientes [… D C B A] tiene la expresión siguiente

$$\ldots + Dx^3 + Cx^2 + Bx + A \quad (33)$$

Por lo que para obtener el coste marginal correcto debe evaluarse este polinomio en $x_{ij} = u_1 - l$, o en general en $x_{ij} = u_k - u_{k-1}$

De esta manera, se han obtenido las nuevas matrices "C" y "U" con las que iniciar la siguiente iteración del algoritmo PDTCD.

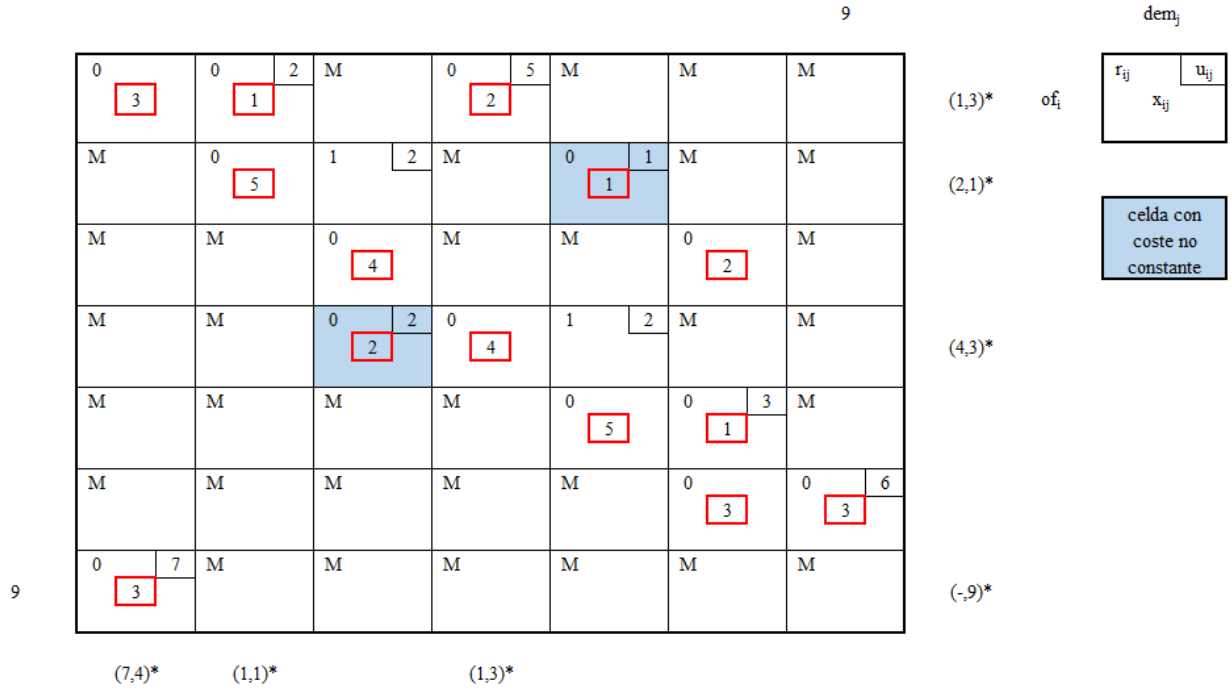
Ejemplo.- *Para la red de la Figura 25, las matrices de costes marginales y capacidad modificada obtenidas al estandarizar, y con las que se parte el algoritmo son:*

$$C = \begin{bmatrix} 0 & 2 & M & 4 & M & M & M \\ M & 0 & 4 & M & 3 & M & M \\ M & M & 0 & M & M & 2 & M \\ M & M & 1 & 0 & 2 & M & M \\ M & M & M & M & 0 & 2 & M \\ M & M & M & M & M & 0 & 0 \\ 0 & M & M & M & M & M & M \end{bmatrix} \qquad U = \begin{bmatrix} M & 2 & M & 5 & M & M & M \\ M & M & 2 & M & 1 & M & M \\ M & M & M & M & M & 3 & M \\ M & M & 2 & M & 2 & M & M \\ M & M & M & M & M & 3 & M \\ M & M & M & M & M & M & 6 \\ 7 & M & M & M & M & M & M \end{bmatrix}$$

En azul se indican los arcos con costes dependientes del flujo.

La aplicación del algoritmo PDTCD, en su primera iteración, alcanza en un cierto momento los siguientes flujos (por simplicidad, se omiten pasos previos):

Tabla 18.- Flujo en la primera iteración del algoritmo PDTCD para la red de la Figura 26.



Sobre esta tabla, puede observarse que, tras el marcaje de las filas y columnas, no se ha alcanzado el nodo de demanda, por lo que ent=0.

La ejecución de "compruebasaturacd" ofrecerá los siguientes resultados:

$$modCU = 1 \qquad ii = 1$$

pues el arco (2,5) (arco guardado en la fila "ii" de "nodoscd") está saturado y su capacidad puede aumentarse hasta 3 unidades.

Por tanto, "nuevaCU" aumentará la capacidad del arco (2,5) en una unidad, $u_{25} = 2$, y calculará el coste marginal.

$$\partial z_{25} = \begin{cases} 3 & 0 \leq x_{ij} \leq 1 \\ 3(x_{25} - 1) + 4{,}5 & 1 < x_{ij} \leq 3 \end{cases}$$

En el cálculo a mano, la operación es sencilla:

$$\partial z_{25}(u_{25}) = 3(u_{25} - 1) + 4{,}5 = 3(2 - 1) + 4{,}5 = 7{,}5$$

Sin embargo, para evitar los problemas a la hora de evaluar el polinomio en MATLAB, se determina primero en qué tramo debe evaluarse.

```
for kk=1:(length(CDM(ii).breaks)-1)
    if U(i,j)>CDM(ii).breaks(kk) && U(i,j)<=CDM(ii).breaks(kk+1)
        tramo=kk;
    end
end
```

siendo $CDM(1).breaks = [0\ 1\ 3]$

Para kk=1, no se cumple la condición del if, pues no se cumple $0 < u_{25} = 2 \leq 1$. *Es sencillo deducir, que para kk=2, sí se cumple la condición* $1 < u_{25} = 2 \leq 3$, *por lo que el coste marginal debe determinarse empleando el polinomio del tramo 2 del coste marginal del arco (2,5). Dicho polinomio vendrá indicado en MATLAB como el vector de sus coeficientes, es decir,*

$$CDM(1).coefs(2,:) = [3 \quad 4{,}5]$$

La función "polyval" interpreta que dicho polinomio es $3x + 4{,}5$, *por lo que el coste marginal se calculará evaluando este polinomio en* $x = u_{25} - 1$. *Luego finalmente:*

$$\partial z_{25}(u_{25}) = 3(u_{25} - 1) + 4{,}5 = 3(2 - 1) + 4{,}5 = 7{,}5$$

Las matrices "C" y "U" que devuelve la función "nuevaCU", y con las que continúa el algoritmo en la Iteración 2 son:

$$C = \begin{bmatrix} 0 & 2 & M & 4 & M & M & M \\ M & 0 & 4 & M & 7{,}5 & M & M \\ M & M & 0 & M & M & 2 & M \\ M & M & 1 & 0 & 2 & M & M \\ M & M & M & M & 0 & 2 & M \\ M & M & M & M & M & 0 & 0 \\ 0 & M & M & M & M & M & M \end{bmatrix} \quad U = \begin{bmatrix} M & 2 & M & 5 & M & M & M \\ M & M & 2 & M & 2 & M & M \\ M & M & M & M & M & 3 & M \\ M & M & 2 & M & 2 & M & M \\ M & M & M & M & M & 3 & M \\ M & M & M & M & M & M & 6 \\ 7 & M & M & M & M & M & M \end{bmatrix}$$

Se han indicado en color rojo los valores que han cambiado respecto a las matrices C y U originales.

Puede observarse que el arco (4,3) también está saturado, por lo que la segunda iteración esencialmente consistirá en aumentar la capacidad de dicho arco en una unidad y calcular su coste marginal asociado.

Obviando pasos intermedios, finalmente, la ejecución del algoritmo PDTCD

```
arcosci=[1 2 0 2 2;1 4 0 5 4;2 3 0 2 4;3 6 0 3 2;4 5 0 2 2;5 6 0 3 2];
K=[7 0 0 0 0 -6];
nodoscd=[2 5;4 3];
C25=mkpp([0 1 3],[0 3;1.5 3]);
C43=mkpp([0 2 4],[0 0 1;1 0 1]);
CDA=[C25,C43];
[U,C,of,dem,arcos]=estandarizarcd(arcosci,nodoscd,CDA,K,3);
CDM=calculacostemarginal(CDA);
flujo=pdtcd(C,of,dem,U,nodoscd,CDM)
```

da como resultado, en MATLAB, la siguiente matriz de flujo:

```
flujo =

     0     2     0     4     0     0     0
     0     4     1     0     1     0     0
     0     0     3     0     0     3     0
     0     0     2     2     2     0     0
     0     0     0     0     3     3     0
     0     0     0     0     0     0     6
     6     0     0     0     0     0     0
```

3.5. Post-procesado del resultado.

La ejecución de las distintas subrutinas en el algoritmo PDTCD (Figura 29), permite la obtención del óptimo del problema estandarizado, cuyo modelo responde a la ecuación:

$$\begin{aligned} & Min\ z' = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}(x_{ij}) \cdot x_{ij} \\ & s.a. \sum_{k\in D(j)} x_{jk} - \sum_{i\in A(j)} x_{ij} = k_j \quad \forall j = 1,2,\dots,n \\ & \qquad 0 \le x_{ij} \le u_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{34}$$

Sin embargo, la obtención del óptimo del problema original pasa por deshacer los cambios de variable, y obtener la solución para el problema modelado según la ecuación (2).

$$\begin{aligned} & Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}(y_{ij}) \cdot y_{ij} \\ & s.a. \sum_{k\in D(j)} y_{jk} - \sum_{i\in A(j)} y_{ij} = K_j \quad \forall j = 1,2,\dots,n \\ & L_{ij} \le y_{ij} \le U_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{2}$$

Es decir, que al igual que se indicó para el algortimo con costos constantes, se debe hacer un post-procesado de los resultados para obtener el óptimo del problema original. Para ello, en el programa MATLAB se ha implementado una función, "postproccd", cuyo código y cálculos son idénticos a los de la función "postproc" (Apartado 2.4) con la ligera salvedad a la hora de calcular los costes absolutos unitarios para los arcos con costes dependientes del flujo.

La sintaxis para dicha función será:

```
[flujoreal,Kreales,z]=postproccd(flujo,arcos,C,nodoscd,CDA)
```

Como ya se ha indicado previamente en el desarrollo de este capítulo, la matriz "C" obtenida en "estandarizarcd" (y que es la que se introduce como entrada en "postproccd") guarda los costes marginales con los que se inicia el algoritmo PDTCD.

Para los arcos con costes constantes, estos costes marginales coinciden con sus costes absolutos. Para los arcos con costes no constantes, "postproccd" debe calcular el coste absoluto unitario de dichos arcos según el flujo real que transcurra por ellos. En este caso, al exigir como condición previa que la función de costes absolutos sea una función continua, el conflicto de cálculo que aparecía al evaluar los costes marginales con "ppval" desaparece.

El siguiente fragmento de código de "postproccd" permite calcular los costes absolutos de los arcos con coste dependiente del flujo.

```
for ii=1:ncd
    i=nodoscd(ii,1);
    j=nodoscd(ii,2);
    C(i,j)=ppval(CDA(ii),flujoreal(i,j));
end
```

Ejemplo.- *Una vez resuelta la red estandarizada de la Figura 26, con la función "pdtcd" se obtiene el siguiente flujo óptimo:*

```
flujo =

     0     2     0     4     0     0     0
     0     4     1     0     1     0     0
     0     0     3     0     0     3     0
     0     0     2     2     2     0     0
     0     0     0     0     3     3     0
     0     0     0     0     0     0     6
     6     0     0     0     0     0     0
```

Al ejecutar en la ventana de comando de MATLAB:

```
[flujoreal,Kreales,z]=postproccd(flujo,arcos,C,nodoscd,CDA);
```

Se obtienen los siguientes resultados:

```
flujoreal =

     0     2     0     4     0     0     0
     0     4     1     0     1     0     0
     0     0     3     0     0     3     0
     0     0     2     2     2     0     0
     0     0     0     0     3     3     0
     0     0     0     0     0     0     6
     6     0     0     0     0     0     0

Kreales =

     6     0     0     0     0    -6     0


z =

    45
```

Además del siguiente grafo:

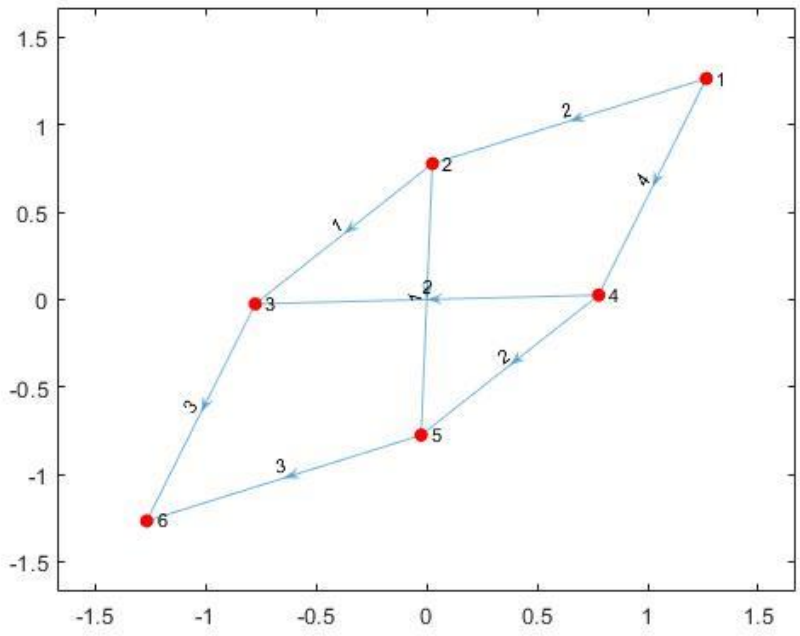


Figura 30.- Flujo óptimo de la red de la Figura 25.

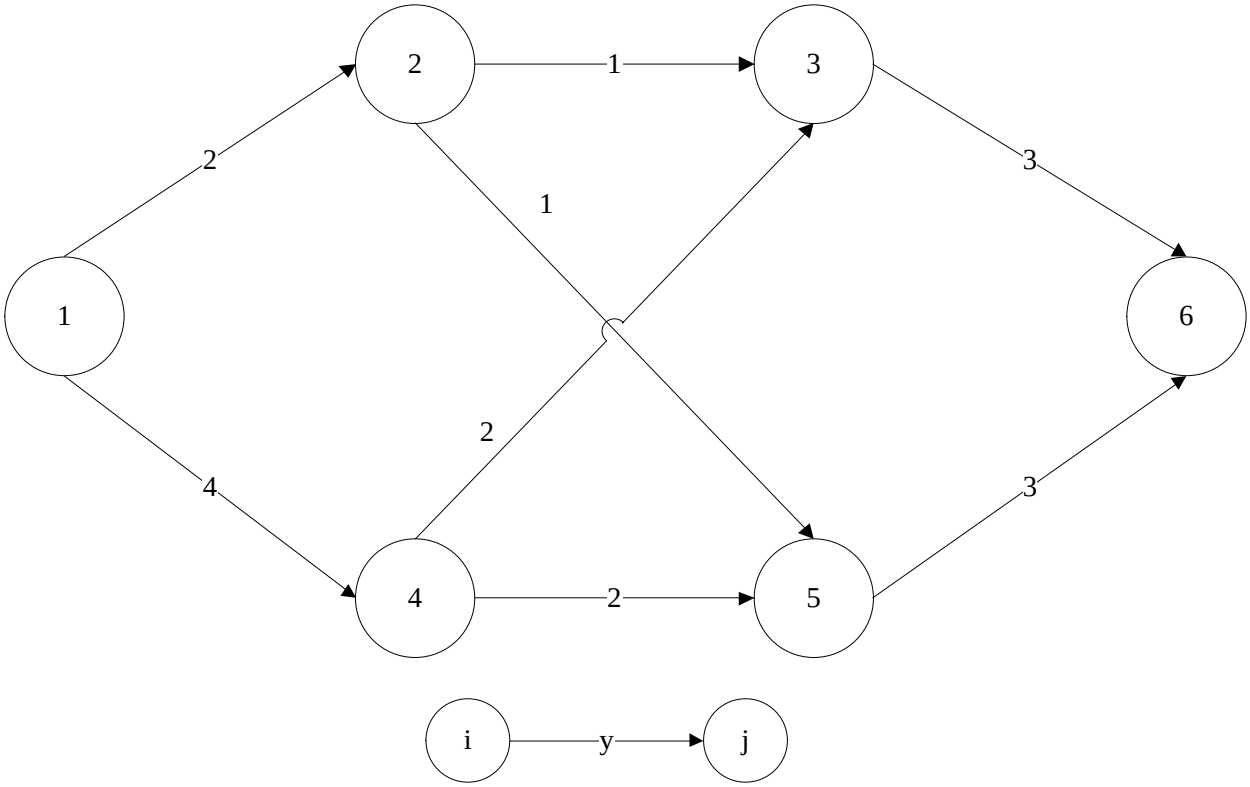


Figura 31.- Grafo realista equivalente al grafo de la Figura 30.

3.6. Resumen de la programación del algoritmo PDTCD.

Con todo lo expuesto hasta ahora, se ha conseguido dar solución al problema de flujo máximo a costo mínimo dependiente del flujo, aplicando el algoritmo Primal-Dual sobre la tabla de transporte. Al igual que se hizo para el algoritmo PDTCI, se muestra el diagrama de flujo completo para las distintas etapas del algoritmo PDTCD:

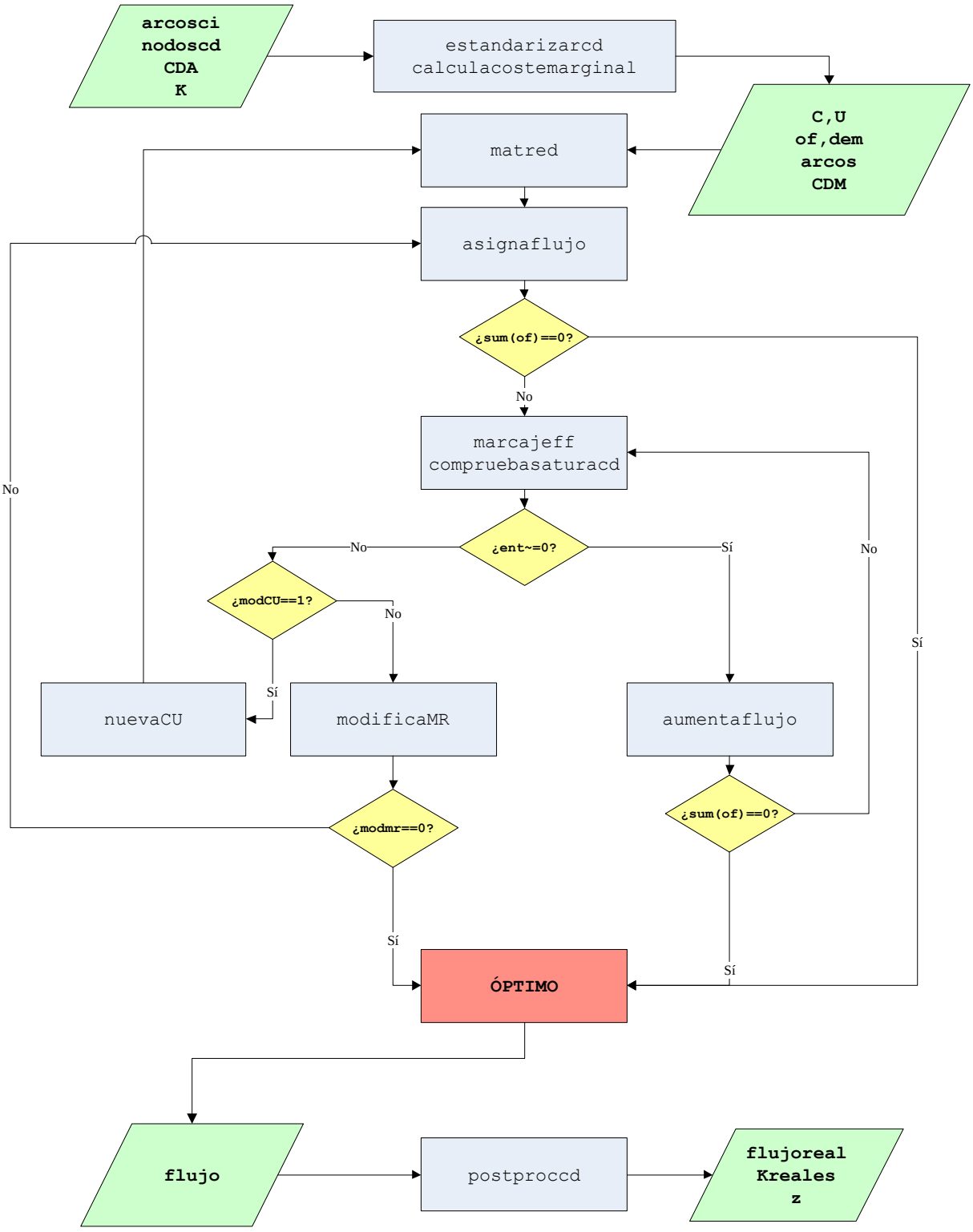


Figura 32.- Diagrama de flujo del algoritmo PDTCD.

Además, se recuerdan en la siguiente tabla, las variables de entrada y salida que intervienen en cada una de las funciones implementadas específicamente para el algoritmo PDTCD:

Tabla 19.- Variables de entrada y salida para las subrutinas del algoritmo PDTCD.

Función	Descripción	Variables de salida	Variables de entrada
estandarizarcd	Tranformación de la red original en una resoluble realizando los cambios de variable adecuados	`U,C,of, dem,arcos`	`arcosci,nodoscd, CDA,K,metodo`
calculacostemarginal	Determinación de la función de costos marginales para los arcos con coste absoluto dependiente del flujo	`CDM`	`CDA`
pdtcd	Resolver la red estandarizada aplicando el algoritmo PDTCD	`flujo`	`C,of,dem,U, nodoscd,CDM`
compruebasaturacd	Determinación de la presencia de algún arco con coste dependiente del flujo que esté saturado (función interna en "pdtcd")	`modCU,ii`	`flujo,U, nodoscd,CDM`
nuevaCU	Recálculo de las matrices de costes marginales y de capacidades modificadas. (función interna en "pdtcd")	`C,U`	`U,C,nodoscd, ii,CDM`
postproccd	Obtención del flujo óptimo en la red original	`flujoreal, Kreales,z`	`flujo,arcos,C, nodoscd,CDA`

3.7. Aplicación práctica del código implementado.

El objetivo de este apartado es comprobar el funcionamiento, paso a paso, del algoritmo PDCTD aplicándolo a la resolución de un ejercicio concreto. Sin embargo, como se indicará más adelante, la resolución completa del problema es ciertamente extensa como para presentarla en su totalidad en este Apartado. Se emplaza al lector al Apéndice C (ver página 191), en el que se indican todos los resultados intermedios de la resolución del problema empleando los tres métodos de estandarización expuestos en el Apartado 2.2.

Para conseguir que MATLAB muestre por pantalla los resultados de cada uno de los cálculos, se añaden unas líneas de código en la función "pdtcd", empleando las funciones de MATLAB "disp" y "pause" para conseguir ese resultado paso a paso.

Se resolverá la siguiente red:

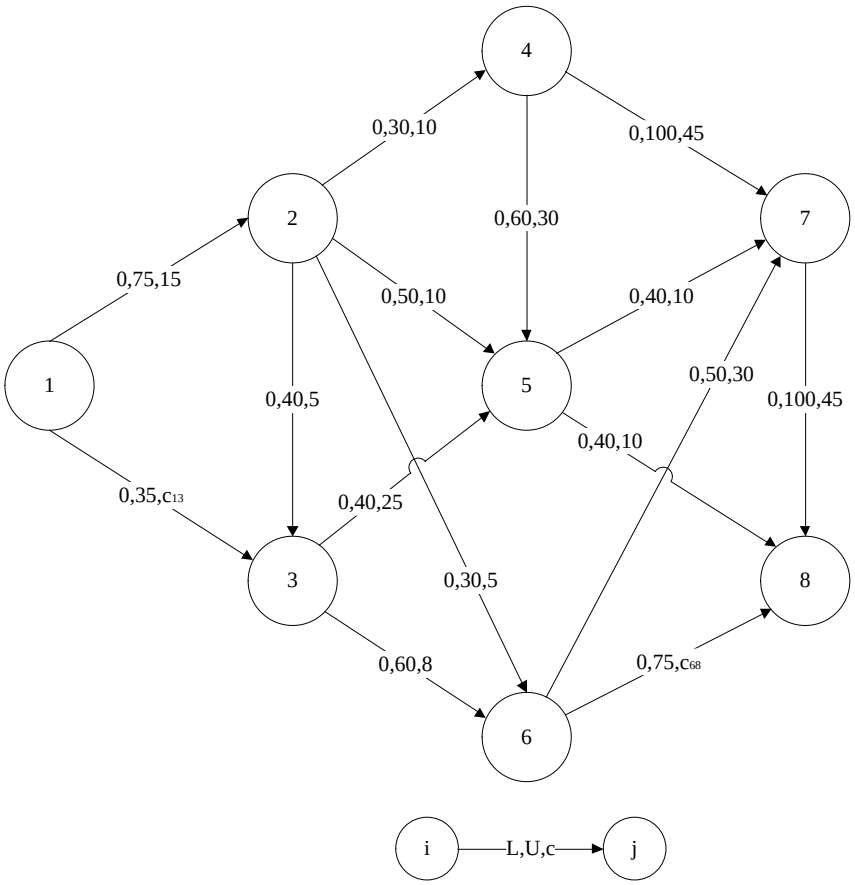


Figura 33.- Red de transporte a resolver en el Apartado 3.7.

En los arcos (1,3) y (6,8) el coste absoluto es dependiente del flujo según las expresiones:

$$c_{13}(x_{13}) = \begin{cases} 10 & 0 \le x_{13} \le 30 \\ 10 + 0{,}1(x_{13} - 30)^2 & 30 \le x_{43} \le 35 \end{cases}$$

$$c_{68}(x_{68}) = \begin{cases} 15 & 0 \le x_{68} \le 70 \\ 15 + 0{,}2(x_{68} - 70) & 70 \le x_{68} \le 75 \end{cases}$$

Para comenzar, se obtienen los parámetros de entrada al programa MATLAB, es decir:

$$arcosci = \begin{bmatrix} 1 & 2 & 0 & 75 & 15 \\ 2 & 3 & 0 & 40 & 5 \\ 2 & 4 & 0 & 30 & 10 \\ 2 & 5 & 0 & 50 & 10 \\ 2 & 6 & 0 & 30 & 5 \\ 3 & 5 & 0 & 40 & 25 \\ 3 & 6 & 0 & 60 & 8 \\ 4 & 5 & 0 & 60 & 30 \\ 4 & 7 & 0 & 100 & 45 \\ 5 & 7 & 0 & 40 & 10 \\ 5 & 8 & 0 & 40 & 10 \\ 6 & 7 & 0 & 50 & 30 \\ 7 & 8 & 0 & 100 & 45 \end{bmatrix}$$

$$nodoscd = \begin{bmatrix} 1 & 3 \\ 6 & 8 \end{bmatrix}$$

A la vista del grafo de la Figura 33, el nodo 1 es el único nodo de oferta, pudiendo ofertar hasta 110 unidades de flujo, y el nodo 8 es el único nodo de demanda, pudiendo demandar hasta 215 unidades de flujo. Por ello, el vector "K" es:

$$K = [110 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad -215]$$

En este caso, se decide estandarizar la red mediante el método 3, introduciendo un nodo artificial de salida unido al nodo de oferta 1, y un nodo artificial de entrada unido al nodo de demanda 8, mediante arcos ficticios.

Además, se introducen las estructuras de costes absolutos dependientes del flujo, para los arcos (1,3) y (6,8), así como el cálculo de los costes marginales.

$$C13.breaks = [0 \quad 30 \quad 35] \qquad C68.breaks = [0 \quad 70 \quad 75]$$

$$C13.coefs = \begin{bmatrix} 0 & 0 & 10 \\ 0{,}1 & 0 & 10 \end{bmatrix} \qquad C68.coefs = \begin{bmatrix} 0 & 15 \\ 0{,}2 & 15 \end{bmatrix}$$

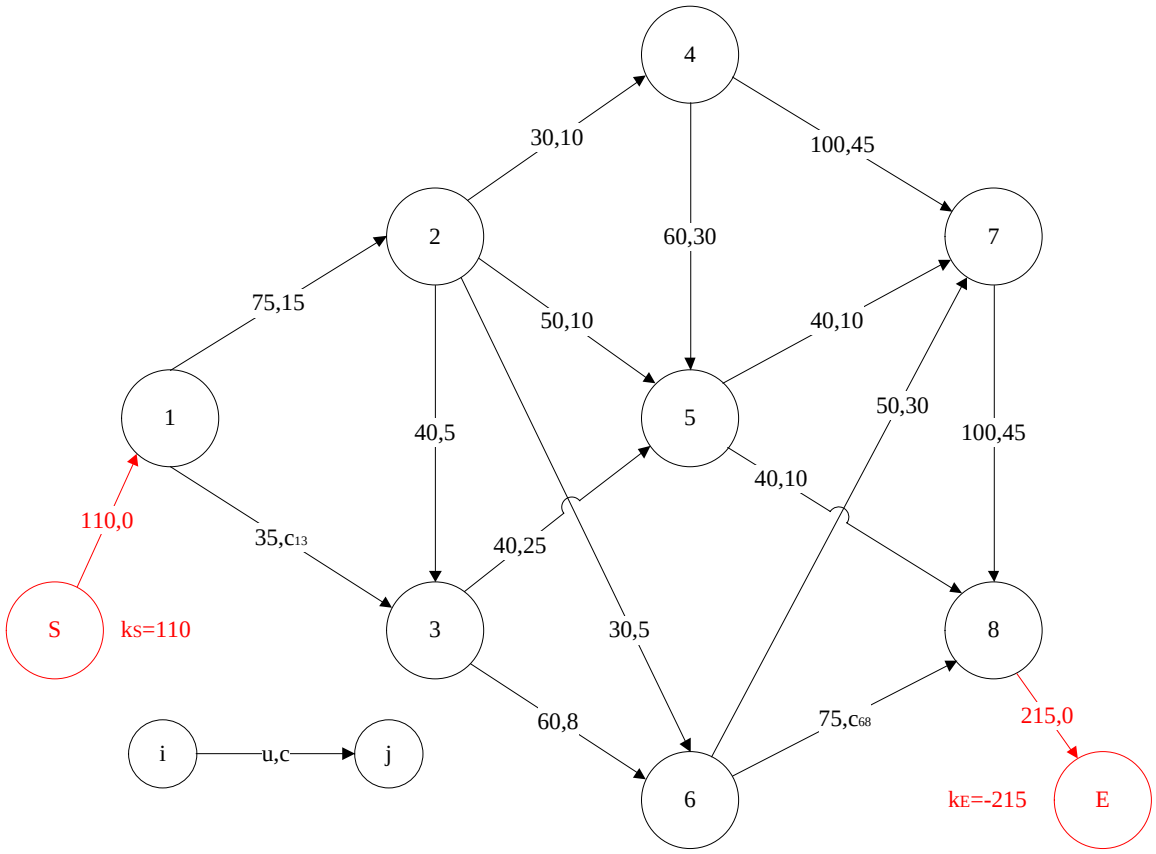


Figura 34.- Grafo de la Figura 33, estandarizado según el método 3.

Los datos que definen la red de la Figura 33 se introducen en la ventana de comandos de MATLAB mediante las siguientes líneas de código:

```
arcosci=[1 2 0 75 15;2 3 0 40 5;2 4 0 30 10;2 5 0 50 10;2 6 0 30 5;3 5 0 40
25;3 6 0 60 8;4 5 0 60 30;4 7 0 100 45;5 7 0 40 10;5 8 0 40 10;6 7 0 50
30;7 8 0 100 45];
nodoscd=[1 3;6 8];
K=[110 0 0 0 0 0 0 -215];
C13=mkpp([0 30 35],[0 0 10;0.1 0 10]);
C68=mkpp([0 70 75],[0 15; 0.2 15]);
CDA=[C13,C68];
[U,C,of,dem,arcos]=estandarizarcd(arcosci,nodoscd,CDA,K,3);
CDM=calculacostemarginal(CDA);
```

Obteniéndose las siguientes matrices:

```
U =

   Inf    75    30   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf    40    30    50    30   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    40    60   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    60   Inf   100   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    40    40   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    50    70   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   100   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   215
   110   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf


C =

     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf


of =

   110   110   110   110   110   110   110   110   220


dem =

   110   110   110   110   110   110   110   110   220
```

y el siguiente grafo, que representa a la red de la Figura 33.

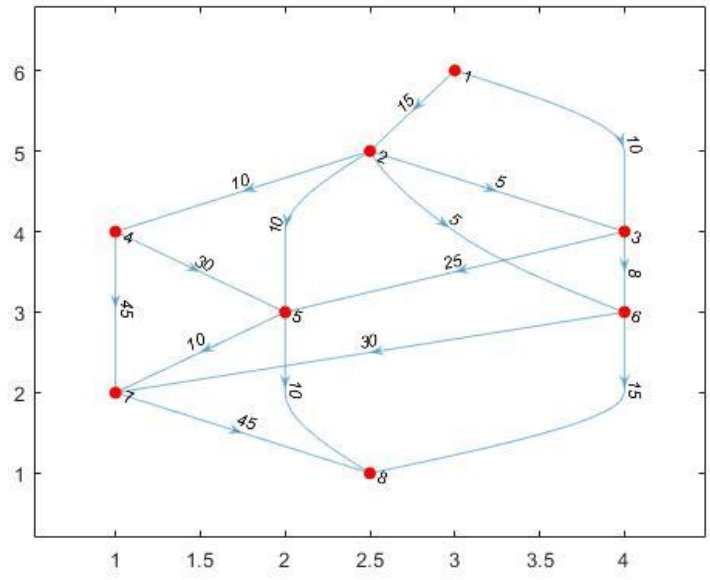


Figura 35.- Grafo de la Figura 33, representado en MATLAB.

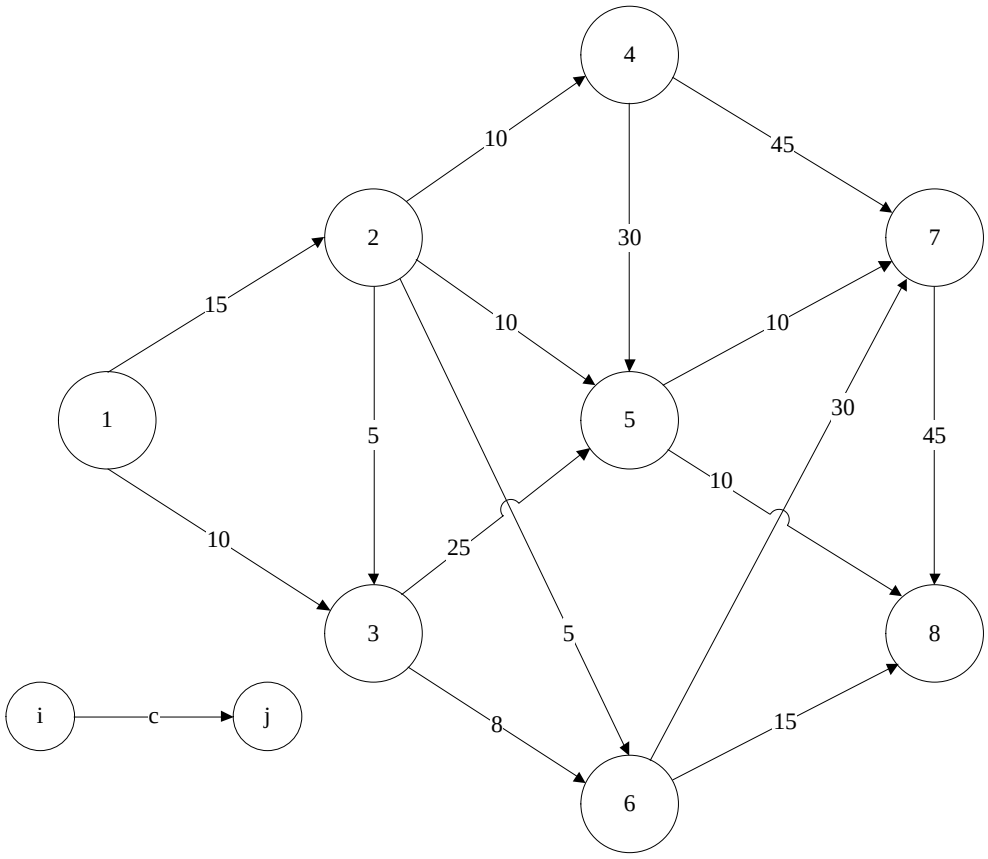


Figura 36.- Grafo realista equivalente al grafo de la Figura 35.

Con estas matrices, ya puede ejecutarse la función “pdtcd”.

```
[flujo]=pdtcd(C,of,dem,U,nodoscd,CDM)
```

En primer lugar, MATLAB muestra por pantalla las matrices de costes marginales y capacidades modificadas.

```
Costes marginales
     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf


Capacidades modificadas
   Inf    75    30   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf    40    30    50    30   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    40    60   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    60   Inf   100   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    40    40   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    50    70   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   100   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   215
   110   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

Luego, se inicia la primera iteración del algoritmo PDTCD, obteniendo las celdas básicas de la red, mediante la obtención de la matriz reducida de los costes marginales.

```
ITERACIÓN 1
Costos relativos
     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

El siguiente paso es realizar la asignación directa de flujos sobre la red básica que, en este caso, da como resultado la siguiente matriz de flujo y los siguientes vectores de oferta y demanda:

```
Flujo actual
   110     0     0     0     0     0     0     0     0
     0   110     0     0     0     0     0     0     0
     0     0   110     0     0     0     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0   110     0     0     0     0
     0     0     0     0     0   110     0     0     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0   110     0
     0     0     0     0     0     0     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   220

     0     0     0     0     0     0     0     0   220
```

Siguiendo el algoritmo (Figura 32), se realiza el marcaje de las filas y columnas, obteniéndose las matrices “I” y “J” siguientes

```
Marcaje del algoritmo FF
     1   110     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0   220     1

     9     0     0     0     0     0     0     0     0
   110     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

Como no se ha alcanzado el nodo de entrada en el marcaje, y además los arcos con costos dependientes no están saturados, se realiza la modificación de la matriz reducida, obteniéndose:

```
Nueva Mat Red
     0     5     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

El algoritmo prosigue ahora intentando asignar flujos de manera directa, aunque con las nuevas celdas básicas esta asignación no permite aumentar flujos en la red. Se obvian, de aquí en adelante las etapas en las que la asignación directa de flujo no ofrece resultados nuevos.

Los sucesivos procesos de marcaje y modificación de la matriz reducida ofrecen los siguientes resultados:

```
Marcaje del algoritmo FF
     1   110     1
     0     0     0
     3    30     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0   220     1

     9     0     1     0     0     0     0     0     0
   110     0    30     0     0     0     0     0     0
     1     0     1     0     0     0     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    20     3   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

```
Marcaje del algoritmo FF
     1   110     1
     2    75     1
     3    30     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0   220     1

     9     1     1     0     0     0     0     0     0
   110    75    30     0     0     0     0     0     0
     1     1     1     0     0     0     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     7     7     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    17     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf

Marcaje del algoritmo FF
     1   110     1
     2    75     1
     3    30     1
     0     0     0
     0     0     0
     6    30     1
     0     0     0
     0     0     0
     0   220     1

     9     1     1     0     0     3     0     0     0
   110    75    30     0     0    30     0     0     0
     1     1     1     0     0     1     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     0     0     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    10     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    23     8   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

```
Marcaje del algoritmo FF
     1   110     1
     2    75     1
     3    30     1
     4    30     1
     5    50     1
     6    30     1
     0     0     0
     0     0     0
     0   220     1

     9     1     1     2     2     3     0     0     0
   110    75    30    30    50    30     0     0     0
     1     1     1     1     1     1     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     0     0     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    10     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    37   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf     2     2   Inf
   Inf   Inf   Inf   Inf   Inf     0    15     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf

Marcaje del algoritmo FF
     1   110     1
     2    75     1
     3    30     1
     4    30     1
     5    50     1
     6    30     1
     0     0     0
     8    30     1
     0   220     1

     9     1     1     2     2     3     0     6     8
   110    75    30    30    50    30     0    30    30
     1     1     1     1     1     1     0     1     1
```

Aquí puede observarse que el marcaje alcanza el nodo de entrada, por lo que puede aumenarse el flujo en la red 30 unidades. El nuevo flujo es:

```
Flujo actual
    80     0    30     0     0     0     0     0     0
     0   110     0     0     0     0     0     0     0
     0     0    80     0     0    30     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0   110     0     0     0     0
     0     0     0     0     0    80     0    30     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0    80    30
    30     0     0     0     0     0     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   190

     0     0     0     0     0     0     0     0   190
```

Donde ya se ha saturado la capacidad modificada del arco (1,3), que era de 30 unidades.

El siguiente marcaje de filas y columnas resulta en las siguientes etiquetas:

```
Marcaje del algoritmo FF
     1    80     1
     2    75     1
     0     0     0
     4    30     1
     5    50     1
     0     0     0
     0     0     0
     0     0     0
     0   190     1

     9     1     1     2     2     0     0     0     0
    80    75     0    30    50     0     0     0     0
     1     1     0     1     1     0     0     0     0
```

Al no alcanzarse el nodo de entrada, y al ya haber saturado un arco con coste dependiente, se aumenta la capacidad modificada del arco (1,3) en una unidad (hasta las 31 unidades), y se calcula su nuevo coste marginal:

```
Nuevos costes marginales
     0    15.0    16.3     Inf     Inf     Inf     Inf     Inf     Inf
   Inf     0.0     5.0    10.0    10.0     5.0     Inf     Inf     Inf
   Inf     Inf     0.0     Inf    25.0     8.0     Inf     Inf     Inf
   Inf     Inf     Inf     0.0    30.0     Inf    45.0     Inf     Inf
   Inf     Inf     Inf     Inf     0.0     Inf    10.0    10.0     Inf
   Inf     Inf     Inf     Inf     Inf     0.0    30.0    15.0     Inf
   Inf     Inf     Inf     Inf     Inf     Inf     0.0    45.0     Inf
   Inf     Inf     Inf     Inf     Inf     Inf     Inf     0.0     0.0
     0     Inf     Inf     Inf     Inf     Inf     Inf     Inf     Inf

Nuevas capacidades
   Inf    75.0    31.0     Inf     Inf     Inf     Inf     Inf     Inf
   Inf     Inf    40.0    30.0    50.0    30.0     Inf     Inf     Inf
   Inf     Inf     Inf     Inf    40.0    60.0     Inf     Inf     Inf
   Inf     Inf     Inf     Inf    60.0     Inf   100.0     Inf     Inf
   Inf     Inf     Inf     Inf     Inf     Inf    40.0    40.0     Inf
   Inf     Inf     Inf     Inf     Inf     Inf    50.0    70.0     Inf
   Inf     Inf     Inf     Inf     Inf     Inf     Inf   100.0     Inf
   Inf     Inf     Inf     Inf     Inf     Inf     Inf     Inf   215.0
 110.0     Inf     Inf     Inf     Inf     Inf     Inf     Inf     Inf
```

Desde estas matrices arranca la segunda iteración del algoritmo PDTCD. Para no alargar en exceso la entensión de esta resolución, se omiten aquí el resto de resultados intermedios que devuelve MATLAB en la ventana de comandos (puede consultarse la resolución completa en el Apéndice C.3, ver página 255).

Para comprender la extensión de la resolución del problema, en la siguiente tabla se indica, a título informativo, el número de veces que se ejecuta cada subrutina (marcajes, modificaciones de la matriz reducida, aumentos de flujo, etc.) realizados en cada iteración del algoritmo PDTCD ejecutado hasta alcanzar el óptimo:

Tabla 20.- Subrutinas (PDTCD) ejecutadas en la resolución de la red de la Figura 33 (método 3).

	matred	**asignaflujo**	**marcajeff + compruebasaturacd**	**aumentaflujo**	**modificamr**	**nuevaCU**
Iteración 1	1	6	7	1	5	1
Iteración 2	1	7	10	3	6	1
Iteración 3	1	7	10	2	7	1
Iteración 4	1	4	5	1	3	1
Iteración 5	1	4	5	1	3	1
Iteración 6	1	4	5	1	3	1
Iteración 7	1	1	1	0	1	0

Finalmente, tras las 7 iteraciones:

```
flujo =

     0    75    35     0     0     0     0     0     0
     0    35     5     0    40    30     0     0     0
     0     0    70     0     0    40     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0    70     0     0    40     0
     0     0     0     0     0    40     0    70     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0     0   110
   110     0     0     0     0     0     0     0     0
```

Para finalizar el algoritmo, se realiza el post-procesado del resultado anterior, mediante:

```
[flujoreal,Kreales,z]=postproccd(flujo,arcos,C,nodoscd,CDA)
```

Con esta función, se obtiene el óptimo de la red original así como los niveles de oferta y demanda que realmente se satisfacen en la red y el valor de la función objetivo.

```
flujoreal =

     0    75    35     0     0     0     0     0     0
     0    35     5     0    40    30     0     0     0
     0     0    70     0     0    40     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0    70     0     0    40     0
     0     0     0     0     0    40     0    70     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0     0   110
   110     0     0     0     0     0     0     0     0

Kreales =

   110     0     0     0     0     0     0  -110     0


z =

       3907.5
```

Además, se muestra el flujo óptimo en la red mediante el siguiente grafo:

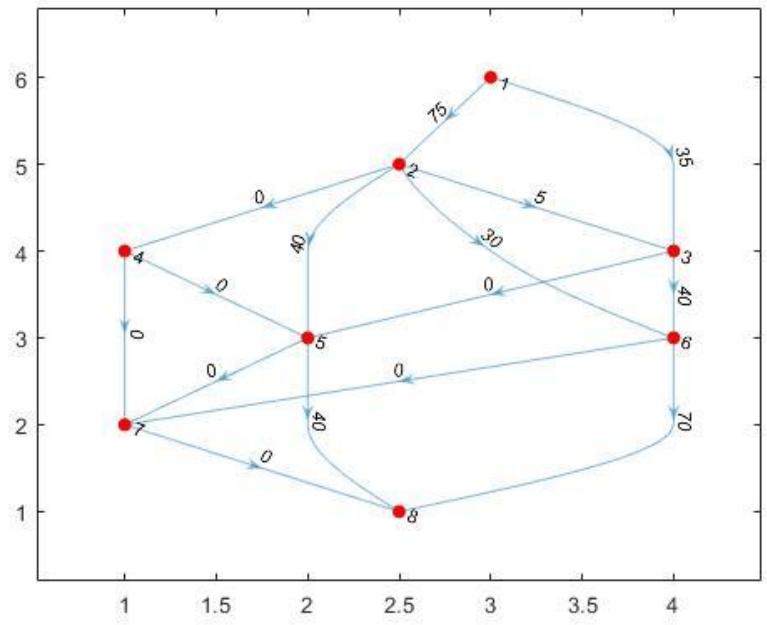


Figura 37.- Flujo óptimo, calculado con MATLAB, para la red de la Figura 33.

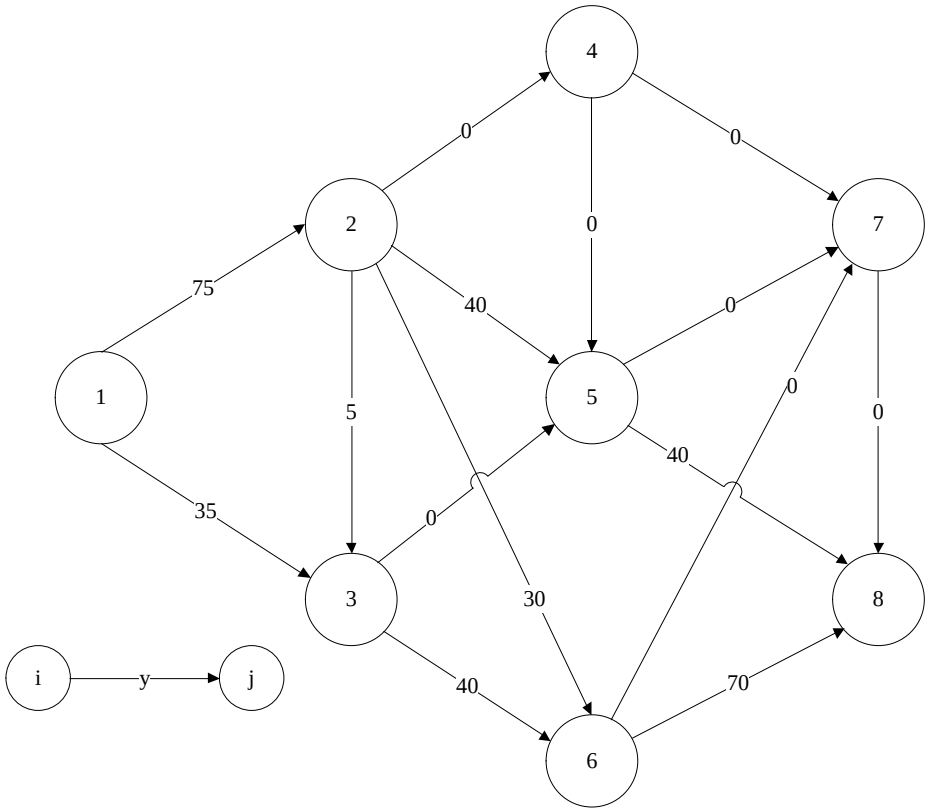


Figura 38.- Grafo realista equivalente al grafo de la Figura 37.

4 ALGORITMOS COMPETITIVOS Y REVISIÓN DEL ESTADO DEL ARTE

El algoritmo Primal-Dual no es el único algoritmo que permite resolver el problema de flujo máximo a coste mínimo. En la literatura específica sobre redes de transporte se recogen otros algoritmos, que serán expuestos de manera breve en este capítulo.

Además, se realizará una comparación de algoritmos en base a los tiempos de ejecución de los mismos.

La Teoría de la Complejidad Computacional estudia los recursos requeridos durante el cálculo para resolver un problema. Un cálculo resulta complejo si es difícil de realizar. En este contexto, es posible definir la complejidad de cálculo como la cantidad de recursos necesarios para efectuar un cálculo. Así, un cálculo difícil requerirá más recursos que uno de menor dificultad. Los recursos comúnmente estudiados son el tiempo (número de pasos de ejecución de un algoritmo para resolver un problema) y el espacio (cantidad de memoria utilizada para resolver un problema) [1]

Se introducen aquí algunas definiciones.

En computación, cuando el tiempo de ejecución de un algoritmo es menor que un cierto valor calculado a partir del número de variables implicadas (generalmente variables de entrada) usando una fórmula polinómica, se dice que dicho problema se puede resolver en un tiempo polinómico. Dentro de los tiempos polinómicos, se distinguen los tiempos logarítmicos O(log n), lineales O(n), cuadráticos $O(n^2)$, etc., siendo n el tamaño de la entrada.

Además, en el contexto de los números enteros, se dice que un algoritmo es fuertemente polinomial si el número de operaciones en el modelo aritmético está limitado por un polinomio en la dimensión del problema y el espacio que ocupa la computación del algoritmo está limitada por un polinomio en el tamaño de la entrada [2].

Por otra parte, un algoritmo se ejecuta en tiempo pseudo-polinómico si el tiempo de ejecución es polinómnico en el valor numérico de la entrada (es decir, el entero mayor presente en la entrada), pero no necesariamente en la longitud (número de bits) de la entrada.

El objetivo de esta revisión del estado del arte es describir algoritmos que permitan la resolución del problema de flujo máximo a coste mínimo dependiente del flujo. Sin embargo, y tal y como se ha realizado en el Capítulo 3, existe una estrategia que consiste en realizar una resolución sucesiva de distintos problemas en los que los costes se consideran constantes, simplificando así las no linealidades del problema.

Por ello, esta revisión del estado del arte sobre algoritmos competitivos tendrá la siguiente estructura.

- Descripción y comparación de otros algoritmos básicos estudiados en la bibliografía para la resolución del problema de flujo máximo a coste mínimo (constante).
- Descripción y comparación de algoritmos polinomiales que permiten la resolución del problema.
- Descripción de algoritmos para la resolución del problema no lineal.

4.1. Algoritmos básicos.

A continuación, se presentan algunos algoritmos básicos para la resolución del problema de flujo máximo a coste mínimo. Estos algoritmos se encuentran entre los algoritmos conocidos más eficientes en matemáticas aplicadas, informática e investigación de operaciones para resolver problemas de optimización a gran escala [3].

El problema que se pretende resolver debe tener las siguientes características:

- Todos los datos (costes, ofertas, demandas y capacidades) son valores enteros.
- El grafo de la red es orientado.
- La red está equilibrada, de manera que la suma de las ofertas coincide con la suma de las demandas. Así el problema tiene solución factible.
- Todos los costes son no negativos.

La mayoría de estos algoritmos resuelven una secuencia de problemas de ruta mínima para aumentar el flujo en una red residual. Dicha red residual G(x) correspondiente a un flujo x se define de la siguiente manera:

- Se reemplaza cada arco (i,j) de la red por dos arcos (i,j) y (j,i).
- El arco (i,j) tendrá coste c_{ij} y capacidad residual $u_{ij} - x_{ij}$
- El arco (j,i) tendrá coste $c_{ji} = -c_{ij}$ y capacidad residual x_{ij}

De esta manera, la red residual solo posee arcos con capacidad positiva.

El problema primal tiene la siguiente formulación matemática:

$$\begin{aligned} & Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}(y_{ij}) \cdot y_{ij} \\ & s.a. \sum_{k\in D(j)} y_{jk} - \sum_{i\in A(j)} y_{ij} = K_j \quad \forall j = 1,2,\dots,n \\ & L_{ij} \le y_{ij} \le U_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{2}$$

Expresión que, una vez realizado el cambio de variable para eliminar la cota inferior, se convierte en:

$$\begin{aligned} & Min\ z' = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}(x_{ij}) \cdot x_{ij} \\ & s.a. \sum_{k\in D(j)} x_{jk} - \sum_{i\in A(j)} x_{ij} = k_j \quad \forall j = 1,2,\dots,n \\ & 0 \le x_{ij} \le u_{ij} \qquad\qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{34}$$

Identificando las variables duales λ_j, correspondientes a las restricciones de conservación en cada nodo j, y las variables α_{ij}, correspondientes a las restricciones de capacidad superior en los arcos (i,j), la formulación dual del problema es:

$$\begin{aligned} &Max\ z' = \sum_{j=1}^{n} \lambda_j k_j + \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_{ij} u_{ij} \\ &s.a.\ \lambda_i - \lambda_j + \alpha_{ij} < c_{ij}(x_{ij}) \quad \forall j = 1,2,\dots,n \\ &\quad \alpha_{ij} \leq 0, \lambda_j \text{ libre} \qquad \forall i,j = 1,2,\dots,n \end{aligned} \tag{35}$$

Las condiciones de optimalidad, en general, proporcionan un método de validación simple para comprobar si un flujo x es óptimo para el problema. En este sentido, la literatura [3] recoge tres formas equivalentes de expresar estos criterios de optimalidad.

- Criterio de optimalidad de ciclos negativos.

  Una solución factible x* es óptima para un problema de flujo máximo a coste mínimo si y solo si la red residual G(x*) no contiene ciclos orientados con costes negativos.

- Criterio de optimalidad de costes relativos

  Una solución factible x* es óptima para un problema de flujo máximo a coste mínimo si y solo si para todos los arcos (i,j) de G(x*) su coste relativo r_{ij} es no negativo.

$$r_{ij} = c_{ij} - (\lambda_i - \lambda_j) \geq 0 \qquad \forall (i,j) \in G(x^*) \tag{36}$$

- Condiciones de holgura

  Una solución factible x* es óptima para el problema de flujo máximo a coste mínimo si y solo si se cumplen las condiciones obtenidas en los teoremas de holgura.

$$\begin{aligned} &\text{Si } r_{ij} > 0 \rightarrow x_{ij}^* = 0. \\ &\text{Si } 0 < x_{ij}^* < u_{ij} \rightarrow r_{ij} = 0 \\ &\text{Si } r_{ij} < 0 \rightarrow x_{ij}^* = u_{ij}. \end{aligned} \tag{37}$$

Las demostraciones correspondientes a la obtención del problema dual y a las distintas expresiones de los criterios de optimalidad, pueden encontrarse con detalle en la bibliografía.

Además, la variable dual λ_j, correspondiente a la restricción de conservación en cada nodo j, recibe también el nombre de potencial del nodo j, en analogía al concepto de potencial eléctrico. Una solución primal x debe cumplir los criterios de optimalidad indicados en (36) y (37). El aumento de flujo en un arco (i,j) que aprovecha el movimiento de flujo desde un nodo con potencial alto a un nodo con potencial bajo tiene la posibilidad de reducir el coste. Según la ecuación (36) si $(\lambda_i - \lambda_j) \geq c_{ij}$, entonces el coste relativo r_{ij} es negativo, es decir, si un arco tiene una diferencia de potenciales mayor que su coste absoluto, siempre se puede encontrar un ciclo en la red residual tal que, al aumentar el flujo a través de él, se obtiene una solución primal x' que mejora a la solución anterior x.

En adelante, a las variables λ_j se les nombrará de manera indistinta como variables duales o potenciales.

A continuación, se presentan de manera descriptiva algunos algoritmos básicos.

4.1.1 Algoritmo de cancelación de ciclo.

El criterio de optimalidad de ciclos negativos sugiere de manera inmediata un algoritmo sencillo para resolver el problema, que se denomina algoritmo de cancelación de ciclo. Este algoritmo parte de un flujo factible en la red (resolviendo un problema de flujo máximo). A partir de ahí, busca en cada iteración ciclos orientados con coste negativo, y aumenta el flujo en dichos ciclos. De esta manera, en cada iteración se mantienen flujos factibles que van mejorando progresivamente el valor de la función objetivo. Según el criterio de optimalidad, una vez se han cancelado todos los ciclos con coste negativo, se habrá alcanzado el óptimo del problema [3].

Cada iteración del algoritmo disminuye estrictamente el valor de la función objetivo. Sin embargo, de manera genérica, el algoritmo de cancelación de ciclo no especifica un orden a la hora de seleccionar los ciclos negativos en la red.

Según sea el método de selección del ciclo negativo, se pueden obtener distintas versiones polinomiales del algoritmo:

- Aumento del flujo en un ciclo negativo con máxima mejora de la función objetivo.

  Según la bibliografía [3], este método puede resolver el problema en $O(m\log(mCU))$ iteraciones (siendo "C" el coste del arco con mayor coste de la red y "U" la capacidad del arco con mayor capacidad de la red: $c_{ij} \leq C, x_{ij} \leq U,\ \forall(i,j) \in L$, con L el conjunto de arcos de la red).

- Aumento del flujo en un ciclo negativo con mínimo costo medio.

  Se define el costo medio de un ciclo como su coste total dividido entre el número de arcos que contiene. Distintas investigaciones han demostrado que si el algoritmo de cancelación de ciclo siempre aumenta el flujo a través del ciclo negativo con mínimo costo medio, el problema puede resolverse en $O(\min\{nm\log(nC), nm^2\log n\})$ iteraciones [3].

```
algorithm cycle-canceling;
begin
    establish a feasible flow x in the network;
    while G(x) contains a negative cycle do
    begin
        use some algorithm to identify a negative cycle W;
        δ := min{r_ij : (i, j) ∈ W};
        augment δ units of flow in the cycle W and update G(x);
    end;
end;
```

Figura 39.- Pseudocódigo del algoritmo de cancelación de ciclo [3].

Nótese que en la bibliografía [3] (de la que se extrae la Figura 39 anterior), la notación empleada es diferente, siendo en este caso: $r_{ij} = u_{ij} - x_{ij}$, es decir, la capacidad residual en lugar del costo relativo.

4.1.2 Algoritmo de ruta mínima sucesiva.

En contraposición al algoritmo de cancelación de ciclo, que mantiene la factibilidad de las soluciones del problema primal en cada iteración hasta alcanzar el óptimo, el algoritmo de ruta mínima sucesiva mantiene la optimalidad (36) de la solución en cada iteración hasta alcanzar una solución factible. Es decir, se mantienen unos flujos x, que satisfacen las restricciones de capacidades superiores e inferiores, pero que viola la ley de conservación de Kirchhoff en los nodos.

En cada iteración, el algoritmo selecciona un nodo fuente s con oferta no satisfecha y un nodo sumidero t con demanda no satisfecha y trata de envíar flujo desde s hasta t a través de la ruta mínima entre ellos (en la red residual).

Al flujo x, que no tiene por qué cumplir las restricciones de conservación, se le denomina pseudoflujo. El algoritmo finaliza cuando los flujos obtenidos cumplen todos los balances de masa en los nodos.

El algoritmo calcula, para cada nodo, su desequilibrio e_j:

$$e_j = k_j - \sum_{k \in D(j)} x_{jk} + \sum_{i \in A(j)} x_{ij} \qquad (38)$$

De esta manera, si $e_j > 0$, el nodo j es un nodo con exceso; si $e_j < 0$, el nodo j es un nodo con déficit, y si $e_j = 0$, el nodo j está equilibrado. Denotando como E y D, respectivamente, a los conjuntos de nodos con exceso y déficit, se observa que:

$$\sum_{j \in N} e_j = \sum_{j \in N} k_j = 0 \qquad (39)$$

y

$$\sum_{j \in E} e_j = -\sum_{j \in D} e_j \qquad (40)$$

Forzosamente, si la red contiene un nodo con exceso, también contendrá un nodo con déficit.

El algoritmo se basa en los siguientes teoremas (demostrados en la bibliografía [3]):

Teorema.- *Sea un flujo (o pseudoflujo) x que satisface las condiciones de optimalidad de costo relativo con respecto a los potenciales λ. El vector d representa las distancias de la ruta mínima desde el nodo s al resto de nodos de la red residual. Entonces, se cumplen las siguientes propiedades:*

a) El pseudoflujo x también satisface los criterios de optimalidad de coste relativo con respecto a los potenciales $\lambda' = \lambda - d$.

b) Los costes relativos r_{ij} *son cero para todos los arcos (i,j) pertenecientes a la ruta mínima desde el nodo s hacia el resto de nodos de la red.*

Teorema.- *Sea un pseudoflujo (o flujo) x, que satisface el criterio de optimalidad de costes relativos. Entonces el flujo x' obtenido a partir x, enviando flujo desde el nodo s hacia otro nodo j a través de la ruta mínima, también satisface el criterio de optimalidad de costes relativos.*

El algoritmo se inicia con el pseudoflujo $x = 0$. Además, puesto que existen ofertas y demandas k_j, los conjuntos E y D deben ser no vacíos. De esta manera, hasta que todos los nodos estén equilibrados, el algoritmo itera siempre identificando un nodo de exceso k y un nodo de déficit *l*.

Cada iteración del algoritmo resuelve un problema de ruta mínima (obteniendo las distancias entre el nodo k y el resto de nodos de la red residual), actualiza los valores de las variables duales ($\lambda' = \lambda - d$) y aumenta el flujo en la ruta que une el nodo k con el nodo *l*. Así, decrementa estrictamente el exceso del nodo k y el déficit del nodo *l*. Luego, actualiza la red residual, así como los valores e_j, pasando a la siguiente iteración.

Se suele emplear el algoritmo de Dikstra para obtener dicha ruta mínima. Este algoritmo obtiene la ruta mínima desde un nodo con exceso hacia todos los otros nodos de la red. Sin embargo, con encontrar la ruta mínima desde un nodo con exceso hacia un nodo de déficit, es suficiente.

El algoritmo de ruta mínima sucesiva requiere un tiempo de ejecución pseudopolinomial para resolver el problema. Este algoritmo puede generalizarse a un algoritmo polinomial para problemas con costes convexos (no constantes).

```
algorithm successive shortest path;
begin
    x := 0 and π := 0;
    e(i) := b(i) for all i ∈ N;
    initialize the sets E := {i : e(i) > 0} and D := {i : e(i) < 0};
    while E ≠ ∅ do
    begin
        select a node k ∈ E and a node l ∈ D;
        determine shortest path distances d(j) from node s to all
            other nodes in G(x) with respect to the reduced costs c^π_ij;
        let P denote a shortest path from node k to node l;
        update π := π − d;
        δ := min[e(k), − e(l), min{r_ij : (i, j) ∈ P}];
        augment δ units of flow along the path P;
        update x, G(x), E, D, and the reduced costs;
    end;
end;
```

Figura 40.- Pseudocódigo del algoritmo de ruta mínima sucesiva [3].

Nótese el cambio en la notación: π se corresponde con las variables duales λ; b(i) corresponde a los niveles de oferta demanda k_j; c_{ij}^{π} es el coste relativo, y nuevamente $r_{ij} = u_{ij} - x_{ij}$.

4.1.3 Algoritmo de relajación.

Los algoritmos anteriores, así como otros algoritmos clásicos (como el primal-dual o el Out-of-kilter), tienen varias características similares: aplican repetidamente algoritmos de ruta mínima, funcionan en un tiempo pseudopolinomial y sus tiempos de ejecución experimentales han sido probados inferiores al tiempo de ejecución del método simplex para la resolución del problema de flujo a costo mínimo.

El algoritmo de relajación es competivo o incluso mejor que el algoritmo simplex, para algunos tipos de redes. En la práctica, el algoritmo de relajación puede considerarse también como una variación del algoritmo de ruta mínima sucesiva.

El algoritmo se basa en la técnica de Relajación Lagrangiana, que permite resolver problemas de optimización en variables enteras. El procedimiento consiste en relajar las restricciones correspondientes a los balances de masa en los nodos, multiplicando cada restricción por una variable libre λ_j (también denominada "potencial" del nodo j). Sustrayendo esa cantidad en la función objetivo se llega al siguiente problema relajado:

$$Min\ w(\lambda) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\, x_{ij} + \sum_{j=1}^{n} \lambda_j \left[k_j - \sum_{k\in D(j)} x_{jk} + \sum_{i\in A(j)} x_{ij} \right] \qquad (41)$$
$$s.a.\ 0 \le x_{ij} \le u_{ij} \qquad \forall i,j = 1,2,\dots,n$$

Nótese que la solución del problema relajado (41) es un pseudoflujo para el problema de flujo a coste mínimo, pues no tiene por qué cumplir la restricción de conservación en los nodos. Teniendo en cuenta la definición de desequilibrio dada en la ecuación (38), la función objetivo del problema relajado puede reescribirse como:

$$w(\lambda) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\, x_{ij} + \sum_{j=1}^{n} \lambda_j\, e_j \qquad (42)$$

Desde otro punto de vista, la función objetivo también puede reescribirse en términos de los costes relativos:

$$w(\lambda) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\, x_{ij} + \sum_{j=1}^{n} \lambda_j \left[k_j - \sum_{k\in D(j)} x_{jk} + \sum_{i\in A(j)} x_{ij} \right] = \sum_{i=1}^{n}\sum_{j=1}^{n} \left(c_{ij} - \lambda_i + \lambda_j\right) x_{ij} + \sum_{j=1}^{n} \lambda_j k_j$$

$$w(\lambda) = \sum_{i=1}^{n}\sum_{j=1}^{n} r_{ij} x_{ij} + \sum_{j=1}^{n} \lambda_j k_j \qquad (43)$$

Para un vector λ dado, el óptimo del problema relajado se obtiene de manera sencilla:

$$\text{Si } r_{ij} > 0 \rightarrow x_{ij} = 0.$$
$$\text{Si } 0 < x_{ij} < u_{ij} \rightarrow r_{ij} = 0$$
$$\text{Si } r_{ij} < 0 \rightarrow x_{ij} = u_{ij}.$$

Es decir, el flujo óptimo del problema relajado es un pseudoflujo para el problema original, que satisface el criterio de optimalidad (37).

El algoritmo de relajación siempre mantiene un vector de variables duales (o potenciales) λ y un pseudoflujo x, que es el óptimo del problema relajado. Es decir, el par (x,λ) satisface el criterio de optimalidad de costes relativos. El algoritmo realiza repetidamente una de las siguientes operaciones:

- Manteniendo los potenciales λ constantes, modifica el flujo x a un flujo x' que también es un óptimo del problema relajado y que permite decrementar el exceso de al menos un nodo.
- Modifica los potenciales λ a λ' y el flujo x a x', siendo x' el óptimo del problema relajado para las variables λ', aumentando el valor de la función objetivo $w(\lambda') > w(\lambda)$.

El algoritmo prioriza la mejora de la función objetivo frente a la factibilidad de la solución. Finalmente, cuando todos los desequilibrios sean nulos (es decir, se cumplan las restricciones de conservación), el algoritmo ha obtenido el flujo óptimo en el problema de flujo a coste mínimo, puesto que en cada iteración el flujo cumple el criterio de optimalidad.

```
algorithm relaxation;
begin
    x := 0 and π := 0;
    while the network contains a node s with e(s) > 0 do
    begin
        S := {s};
        if e(S) > r(π, S) then adjust-potential;
        repeat
            select an arc (i, j) ∈ (S, S̄) in the residual network with c^π_ij = 0;
            if e(j) ≥ 0 then set pred(j) := i and add node j to S;
        until e(j) < 0 or e(S) > r(π, S);
        if e(S) > r(π, S) then adjust-potential
        else adjust-flow;
    end;
end;
```

Figura 41.- Pseudocódigo del algoritmo de relajación [3].

```
procedure adjust-potential;
begin
    for every arc (i, j) ∈ (S, S̄) with c^π_ij = 0 do send r_ij units of flow on the arc (i, j);
    compute α := min{c^π_ij : (i, j) ∈ (S, S̄) and r_ij > 0};
    for every node i ∈ S do π(i) := π(i) + α;
end;
```

```
procedure adjust-flow;
begin
    trace the predecessor indices to identify the directed path P from node s to node j;
    δ := min[e(s), − e(j), min{r_ij : (i, j) ∈ P}];
    augment δ units of flow along P, update imbalances and residual capacities;
end;
```

Figura 42.- Pseudocódigo de las subrutinas del algoritmo de relajación [3].

Nótense de nuevo, los mismos cambios respecto a la notación empleada en este Capítulo, así como que se indican variables intermedias que no se explican en esta revisión y que pueden consultarse con detalle en la bibliografía [3].

4.1.4 Comparación de algoritmos básicos.

En la siguiente tabla, se resumen las características de los algortitmos básicos anteriores. Se recoge también, la información correspondiente a los algoritmos Primal-Dual y Out-of-kilter, obviados en esta revisión.

Tabla 21.- Resumen de los algoritmos de tiempo pseudopolinomial [3].

Algoritmo	Número de iteraciones	Características
Cancelación de ciclo	O(mCU)	• Mantiene un flujo factible x en cada iteración y aumenta el flujo a través de los ciclos negativos de G(x). • En cada iteración, resuelve un problema de ruta mínima para identificar los ciclos negativos. • Algoritmo muy flexible: algunas reglas para obtener los ciclos negativos permiten obtener algoritmos polinomiales.
Ruta mínima sucesiva	O(nU)	• Mantiene un pseudoflujo x que satisface el criterio de optimalidad y aumenta el flujo a través de la ruta mínima entre un nodo de exceso y un nodo de déficit en la red residual. • En cada iteración, resuelve un problema de ruta mínima con costes no negativos. • Algoritmo muy flexible: seleccionando adecuadamente los aumentos, se pueden obtener varios algoritmos polinomiales.
Primal-Dual	O(min{nU,nC})	• Mantiene un pseudoflujo x, que satisface el criterio de optimalidad. Resuelve un problema de ruta mínima para actualizar las variables duales y reduce la no factibilidad del primal resolviendo un problema de flujo máximo. • En cada iteración, resuelve ambos problemas (ruta mínima y flujo máximo). • Algoritmo similar al de ruta mínima sucesiva.
Out-of-kilter	O(nU)	• Mantiene un flujo factible x en cada iteración e intenta satisfacer el criterio de optimalidad aumentando flujo a través de la ruta mínima. • En cada iteración, resuelve un problema de ruta mínima con costes no negativos. • Puede generalizarse para resolver situaciones en las que el flujo x no satisface las restricciones de capacidad.
Relajación	-	• Mantiene un pseudoflujo x, que satisface el criterio de optimalidad y modifica los flujos en los arcos y los potenciales de cada nodo, aunque la función objetivo puede empeorar ocasionalmente. • Incorporando algunas reglas heurísticas, el algoritmo es muy eficiente en la práctica, siendo el de mayor rendimiento para algunos tipos de problemas.

En general, las investigaciones realizadas por distintos autores [4] [5] [6] coinciden en que los métodos basados en la relajación de restricciones funcionan de manera más eficiente que el resto de algoritmos, siendo, en algunos casos, hasta un orden de magnitud más rápidos.

4.2. Algoritmos polinomiales.

Aunque los algoritmos vistos en el apartado anterior garantizan la convergencia en el óptimo para problemas con variables enteras, a nivel computacional los cálculos no están limitados por ningún polinomio en alguna de las especificaciones del problema. Esta situación no es completamente satisfactoria, por lo que resulta interesante desarrollar algoritmos fuertemente polinomiales, que permitan resolver problemas en los que no todas las variables sean enteras.

Estos algoritmos se basan, fundamentalmente, en la técnica del escalado de datos. Al aplicar esta técnica de resolución, se inician los cálculos partiendo de una solución que incumple uno o varios criterios de optimalidad de la red en Δ unidades. De esta manera, se ha generado un óptimo aproximado (o subóptimo). Eligiendo un valor Δ suficientemente grande, es sencillo encontrar una solución inicial que satisfaga el criterio de optimalidad aproximado. Luego, iterativamente, se reduce el parámetro Δ (generalmente a la mitad de una iteración a la siguiente) y se recalcula el subóptimo, hasta alcanzar un valor Δ=0 (o suficientemente cercano a 0). Esta estrategia de resolución es muy versátil y conduce a distintos algoritmos en función de qué criterio de optimalidad se relaje o de cómo se recalcule el subóptimo.

A continuación, se presenta de manera descriptiva, el fundamento de diversos algoritmos basados en esta técnica.

4.2.1 Algoritmo de escalado en capacidades.

El algoritmo de escalado en capacidades puede considerarse como una variante del algoritmo de ruta mínima sucesiva. El escalado en capacidades asegura que cada aumento de flujo en la ruta mínima es lo suficientemente grande como para reducir el número de iteraciones de $O(nU)$ a $O(m \log U)$.

Desde el punto de vista de los cálculos que se realizan, el algoritmo se inicia a partir de un pseudoflujo x y de los desequilibrios de los nodos, e_j. Gradualmente, el pseudoflujo se convierte a flujo identificando las rutas mìnimas desde los nodos de exceso hacia los nodos de déficit y aumentando el flujo a través de dichas rutas.

Partiendo de un valor $\Delta = 2^{\log U}$, en cada iteración, se realiza un aumento de flujo de Δ unidades, independientemente de que la ruta mínima tenga una capacidad residual mayor, es decir, independientemente de que admita un mayor aumento de flujo.

Para un valor Δ dado, se definen los conjuntos S(Δ) y T(Δ), que engloban respectivamente a los nodos con exceso y déficit superior a Δ.

$$\begin{aligned} S(\Delta) &= \{j: e_j \geq \Delta\} \\ T(\Delta) &= \{j: e_j \leq -\Delta\} \end{aligned} \qquad (44)$$

En cada fase de escalado, el aumento de flujo se inicia en un nodo perteneciente a S(Δ) y finaliza en un nodo perteneciente a T(Δ). Además, los arcos a través de los cuales se aumenta el flujo deben tener una capacidad residual de al menos Δ unidades. Si el aumento de flujo no fuera posible, bien porque alguno de los conjuntos S(Δ) o T(Δ) no contenga ningún nodo, o bien porque no sea posible aumentar el flujo en Δ unidades a través de la ruta mínima encontrada, el algoritmo reduce Δ a la mitad y repite los cálculos.

Finalmente, tras log U fases de escalado, Δ=1, donde todos lo desequilibrios son nulos, y por tanto el flujo obtenido es óptimo para el problema [7].

```
algorithm capacity scaling;
begin
    x := 0 and π := 0;
    Δ := 2^⌊log U⌋;
    while Δ ≥ 1
    begin {Δ-scaling phase}
        for every arc (i, j) in the residual network G(x) do
            if r_ij ≥ Δ and c^π_ij < 0 then send r_ij units of flow along arc (i, j),
              update x and the imbalances e(·);
        S(Δ) := {i ∈ N : e(i) ≥ Δ};
        T(Δ) := {i ∈ N : e(i) ≤ −Δ};
        while S(Δ) ≠ ∅ and T(Δ) ≠ ∅ do
        begin
            select a node k ∈ S(Δ) and a node l ∈ T(Δ);
            determine shortest path distances d(·) from node k to all other nodes in the
                Δ-residual network G(x, Δ) with respect to the reduced costs c^π_ij;
            let P denote shortest path from node k to node l in G(x, Δ);
            update π := π − d;
            augment Δ units of flow along the path P;
            update x, S(Δ), T(Δ), and G(x, Δ);
        end;
        Δ := Δ/2;
    end;
end;
```

Figura 43.- Pseudocódigo del algoritmo de escalado en capacidades [7].

4.2.2 Algoritmo de escalado en costes.

Este algoritmo se basa en el concepto de optimalidad aproximada. Un flujo o pseudoflujo x se dice que es ε-óptimo para un ε>0 si el par (x,λ) satisface los siguientes criterios de ε-optimalidad:

$$\begin{aligned} &\text{Si } r_{ij} > \varepsilon \rightarrow x_{ij} = 0. \\ &\text{Si } -\varepsilon < r_{ij} < \varepsilon \rightarrow 0 \leq x_{ij} \leq u_{ij} \\ &\text{Si } r_{ij} < -\varepsilon \rightarrow x_{ij} = u_{ij}. \end{aligned} \tag{45}$$

Estas condiciones corresponden a las condiciones relajadas del criterio de optimalidad. Cuando ε=0, las condiciones anteriores son exactamente las del criterio de optimalidad expresadas en términos del teorema de holgura (37).

Teorema.- *Para un problema de flujo máximo a coste mínimo con costes enteros, cualquier flujo factible es ε-óptimo cuando $\varepsilon \geq C$, siendo C el coste más alto en la red. Además, si ε<1/n, entonces cualquier flujo factible ε-óptimo es un flujo óptimo de la red [7].*

El algoritmo trata a ε como parámetro e iterativamente se obtienen flujos ε-óptimos para valores de ε sucesivamente más pequeños. Los cálculos se inician tomando $\varepsilon = C$ y un flujo factible que sea ε-óptimo.

Luego, el escalado de los costes se realiza ejecutando un proceso de aproximación y mejora que transforma el flujo ε-óptimo inicial en un flujo ½ε-óptimo. Este proceso se realiza en dos etapas:

- Conversión del flujo ε-óptimo en un pseudoflujo ½ ε-óptimo.
- Conversión del pseudoflujo en flujo, manteniendo la ½ ε-optimalidad de la solución.

Para ello, la aproximación y mejora de la solución emplea el algoritmo de empuje y etiquetado, que es un algoritmo de flujo máximo recogido en la bibliografía y cuyo fundamento y cálculos se obvian en esta revisión. En la Figura 45 se muestra el pseudocódigo de este proceso.

Tras 1+[log(nC)] fases de escalado, se alcanza un valor ε<1/n, por lo que, según el teorema anterior, se ha alcanzado el flujo óptimo del problema.

```
algorithm cost scaling;
begin
    π : = 0 and ϵ : = C;
    let x be any feasible flow;
    while ϵ ≥ 1/n do
    begin
        improve-approximation(ϵ, x, π);
        ϵ : = ϵ/2;
    end;
    x is an optimal flow for the minimum cost flow problem;
end;
```

Figura 44.- Pseudocódigo del algoritmo de escalado en costes [7].

```
procedure improve-approximation(ϵ, x, π);
begin
    for every arc (i, j) ∈ A do
        if c^π_ij > 0 then x_ij : = 0
        else if c^π_ij < 0 then x_ij : = u_ij;
    compute node imbalances;
    while the network contains an active node do
    begin
        select an active node i;
        push/relabel(i);
    end;
end;
```

```
procedure push/relabel(i);
begin
    if G(x) contains an admissible arc (i, j) then
        push δ : = min{e(i), r_ij} units of flow from node i to node j;
    else π(i) : = π(i) + ϵ/2;
end;
```

Figura 45.- Pseudocódigo del proceso de aproximación y mejora en el escalado en costes [7].

4.2.3 Algoritmo de doble escalado.

El algoritmo de doble escalado es análogo al algoritmo de escalado en costes presentado en el apartado anterior, con la salvedad de que emplea una versión mejorada del proceso de aproxmación y mejora. Realizar este proceso empleando la técnica de escalado en capacidades permite reducir el número de aumentos de flujo a O(m log U), pues se asegura que, en cada fase de escalado, se aumenta un flujo Δ suficientemente grande.

El algoritmo resultante ejecuta un escalado en costes en un bucle externo, obteniendo un flujo ε-óptimo para valores de ε sucesivamente más pequeños. En cada fase del escalado en costes, el algoritmo inicia los cálculos a partir de un pseudoflujo y realiza, en un bucle interno, un determinado número de fases de escalado en capacidad para valores de Δ sucesivamente más pequeños.

Este escalado en capacidades presenta ligeras diferencias respecto al algoritmo indicado en el Apartado 4.2.1:

- Los aumentos de flujo terminan en un nodo cuyo déficit no tiene por qué ser mayor que Δ.
- Las capacidades residuales son siempre múltiplos enteros de Δ, puesto que el flujo en cada arco es un múltiplo entero de Δ (suponiendo arcos sin capacidad superior)
- El algoritmo no modifica el flujo en los arcos al comienzo de un determinado escalado para garantizar el cumplimiento del criterio de optimalidad de la solución.

En definitiva, el algoritmo de doble escalado puede describirse con el mismo pseudocódigo del escalado en costes (Figura 44). Sin embargo, el proceso de aproximación y mejora es diferente. Se sustituye la identificación de rutas mediante el algoritmo de empuje y reetiquetado por el escalado en capacidades, lo que permite una mayor eficiencia computacional del algoritmo.

```
procedure improve-approximation(ε, x, π);
begin
    set x := 0 and compute node imbalances;
    π(j) := π(j) + ε, for all j ∈ N2;
    Δ := 2^⌊log U⌋;
    while the network contains an active node do
    begin
        S(Δ) := {i ∈ N1 ∪ N2 : e(i) ≥ Δ};
        while S(Δ) ≠ ∅ do
        begin {Δ-scaling phase}
            select a node k from S(Δ);
            determine an admissible path P from node k to some node l with e(l) < 0;
            augment Δ units of flow on path P and update x and S(Δ);
        end;
        Δ := Δ/2;
    end;
end;
```

Figura 46.- Pseudocódigo del proceso de aproximación y mejora en el doble escalado [7].

4.2.4 Otros algoritmos de escalado.

En este apartado, se describen brevemente otros dos algoritmos fuertemente polinomiales, que no son más que variaciones de los algoritmos anteriores.

- Algoritmo de escalado repetido en capacidades.

  A diferencia del resto de algoritmos vistos hasta ahora en esta revisión, el algoritmo de escalado repetido en capacidades resuelve el problema dual en lugar del problema primal. En este sentido, se obtiene un conjunto óptimo de potenciales, que luego se emplean para determinar el flujo óptimo.

  Sea un nodo j con $k_j = U$, siendo $U = \max\{k_j : j \in N, \forall k_j > 0\}$. El algoritmo comienza aplicando un escalado en capacidades. Si se ha alcanzado un flujo factible antes de que $\Delta \leq k_j/6n^2$, entonces se ha alcanzado el óptimo. En caso contrario, puede demostrarse que existe un arco (j,k) con un flujo suficientemente grande como para considerarlo constante. El algoritmo define un nuevo problema de flujo a coste mínimo con los nodos j y k contraídos en un único nodo p, y el coste de cada arco es el coste relativo del arco correspondiente antes de la contracción. Entonces, se redefine el valor de U y se vuelve a resolver el problema mediante el escalado en capacidades.

  El algoritmo finaliza cuando al aplicar el escalado en capacidades se alcanza un flujo (que cumpla las restricciones de conservación, o bien cuando la red contraída consiste en un único nodo (con solución trivial nula).

  Finalmente, deben expandirse los nodos en orden inverso al que fueron contraídos. Para ello, se asigna un valor de la variable dual para los nodos j y k idéntico al valor obtenido para el nodo p. Una vez expandidos todos los nodos, los potenciales obtenidos son óptimos para el problema de flujo a coste mínimo. Esos potenciales pueden emplearse para obtener el flujo óptimo resolviendo un problema de flujo máximo en la red.

- Algoritmo de escalado mejorado en capacidades.

  Este algoritmo tiene un enfoque similar al algoritmo anterior, pero con las siguientes diferencias:

  - Explícitamente, no se realiza una contracción de nodos.
  - No resuelve de nuevo el problema, pero parte de los resultados obtenidos en las iteraciones anteriores.

  Al evitar las contracciones de nodo, el algoritmo es más sencillo de implementar (puesto que las contracciones implican redefinir la estructura de la red) y mantiene un pseudoflujo que satisface el criterio de optimalidad del dual en cada iteración hasta el final, donde se convierte a un flujo óptimo.

Para terminar este apartado, la siguiente tabla indica los tiempos de ejecución de los algoritmos de escalado descritos.

Tabla 22.- Tiempos de ejecución de los algoritmos polinomiales [7].

Algoritmo	Tiempo de ejecución
Escalado en capacidades	$O((m \log U)(m + n \log n)$
Escalado en costes	$O(n^3 \log(nC))$
Doble escalado	$O(nm \log U \log nC)$
Escalado repetido en capacidades	$O((m^2 \log n)(m + n \log n)$
Escalado mejorado en capacidades	$O((m \log n)(m + n \log n)$

4.3. Algoritmos para costes no lineales.

La forma de la función de los costes absolutos de los arcos de la red es uno de los factores principales que afectan a la complejidad de la resolución de los problemas de redes, en general y del problema de flujo máximo a coste mínimo, en particular.

La no linealidad de la función de costes puede darse, por ejemplo, cuando el coste unitario de un determinado trayecto disminuye al aumentar el flujo o cuando hay un coste fijo al enviar flujo a través de un nuevo arco en la red.

Como ya se ha descrito previamente, el problema lineal puede resolverse en tiempo polinomial empleando algoritmos fuertemente polinomiales [8].

Sin embargo, en la mayoría de problemas reales, la aproximación lineal de los costes puede no representar la realidad adecuadamente. En esos casos, idealmente se emplean las funciones de coste no lineales (ya sean convexas o cóncavas).

La región factible definida por las restricciones del problema suele ser una región convexa, por lo que, si la función de costes es cóncava, es posible que la optimización converja en un óptimo local en lugar de en el óptimo global, aumentando la complejidad de la resolución.

Algunos métodos para la resolución de problemas con costes cóncavos son la programación dinámica o el método de ramificación y acotación. Sin embargo, estos algoritmos no son capaces de trabajar con problemas a gran escala de manera eficiente. De hecho, en [9] se resuelve una red de 19 nodos mediante programación dinámica en 5 horas, y en [10] consideran una red de 21 nodos, resuelta en 13 horas tomando la aproximación lineal de los costes.

Existen otros métodos metaheurísticos, entre los cuales los más usados son los algoritmos de la colonia de hormigas y los algoritmos genéticos.

4.3.1 Aproximación lineal de los costes.

Sea una red G en la que el coste de un arco (i,j) sea una función a trozos lineal convexa:

Tabla 23.- Función de costes absolutos lineal convexa [2].

Intervalo	**Pendiente de $c_{ij}(x_{ij})$**
$\boldsymbol{l_{ij} \le x_{ij} \le u_{ij,1}}$	$c_{ij,1}$
$\boldsymbol{u_{ij,1} \le x_{ij} \le u_{ij,2}}$	$c_{ij,2}$
...	...
$\boldsymbol{u_{ij,r-1} \le x_{ij} \le u_{ij}}$	$c_{ij,r}$

La forma de interpretar la función objetivo $c_{ij}(x_{ij})$ es la siguiente. Cada unidad de flujo en el arco (i,j) entre l_{ij} y $u_{ij,1}$ unidades tiene un costes unitario de $c_{ij,1}$. Una vez el flujo en dicho arco alcanza $u_{ij,1}$, cada unidad de flujo adicional sobre (i,j) tiene un coste de $c_{ij,2}$ hasta que se alcance un flujo de $u_{ij,2}$. Gráficamente, esta situación puede manejarse como si se estuviera enviando flujo a través de un arco paralelo de i hasta j con un coste unitario de $c_{ij,2}$ unidades. Y así sucesivamente. Así, para la función de costes indicada en la Tabla 23, el arco original (i,j) se reemplaza por r arcos paralelos como se indica en la Figura 47.

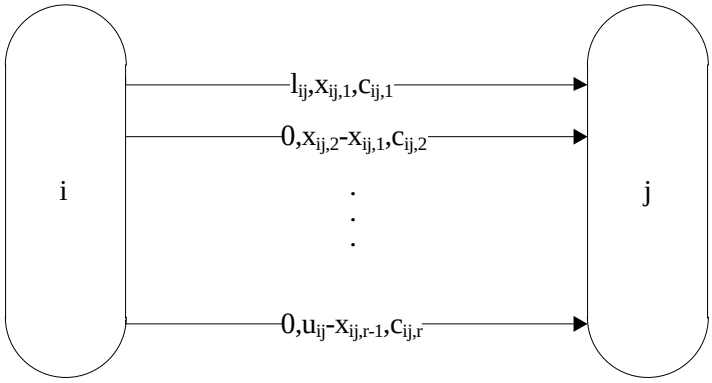


Figura 47.- Datos en los arcos paralelos correspondientes al arco (i,j) [2].

En cada arco se indica su capacidad inferior, capacidad superior y coste unitario, en ese orden.

Propiedad.- *El flujo x_{ij} correspondiente a la red original será la suma de los flujos que transcurren sobre los r arcos paralelos. Debido al sentido estrictamente creciente de los costes (al ser una función convexa), el flujo óptimo en el arco $(i,j)_t$ es nulo a menos que los arcos $(i,j)_1, \dots, (i,j)_{t-1}$ estén todos saturados [2].*

El problema de flujo a coste mínimo lineal puede resolverse con los algoritmos descritos previamente en esta revisión del estado del arte. Además, en caso de que las funciones de costes sean no lineales, pueden aproximarse como lineales siguiendo la misma técnica descrita, considerando intervalos de una unidad de flujo entre l_{ij} y u_{ij}, y aproximando de manera lineal la función correspondiente en cada intervalo. Esto es lo que se hace para resolver el problema presentado en el Capítulo 3, resolviendo la red mediante el método Primal-Dual.

Además de esta aproximación lineal para costes convexos, la bibliografía recoge numerosos desarrollos de algoritmos polinomiales basados en las técnicas ya descritas en este Capítulo, para resolver redes sin linealizar la función de costes.

Se han desarrollado algoritmos que generalizan el escalado en capacidades [11] [12], algoritmos de sobrerrelajación sucesiva [13], generalizaciones del método simplex [14] [15] o generalizaciones del Primal-Dual [16].

Para los arcos (i,j) de la red original en los que $c_{ij}(x_{ij})$ sea una función continua, pero no convexa, la propiedad anterior puede no cumplirse, debido a que los costes no son estrictamente crecientes, por lo que se debe recurrir a otros tipos de algoritmos, que se verán a continuación.

4.3.2 Otros tipos de algoritmos.

- Programación dinámica.

La programación dinámica se suele utilizar en problemas de optimización, donde una solución está formada por una serie de decisiones. El problema original se resuelve combinando las soluciones para subproblemas más pequeños. Se almacenan los resultados de los subproblemas, calculando primero las soluciones para los problemas pequeños, y llegando hasta el tamaño deseado con un proceso iterativo. Con esto se pretende evitar la repetición de cálculos para problemas más pequeños [17].

En la bibliografía se recogen numerosas versiones de algoritmos basados en programación dinámica para resolver problemas de flujo a costo mínimo. Dichos algoritmos resuelven redes con costes lineales y cóncavos [9], aproximando los costes cóncavos a una función lineal [10], algoritmos basados en el método send-and-split (enviar y dividir) [18], entre otros [19].

- Métodos de ramificación y poda.

Los métodos de ramificación y poda, también conocidos como métodos de ramificación y acotado, se interpretan como un árbol de soluciones, en la que en cada rama se alcanza una solución factible posterior a la actual. Los métodos son análogos al algoritmo del árbol que la bibliografía sobre algoritmos de transporte recoge. El algoritmo detecta cuál de las ramas del árbol ya no puede proporcionar una solución óptima al problema, "podando" esa rama y eliminándola de entre las posibles soluciones. Así no se malgastan recursos computacionales en operaciones que se alejan del óptimo.

De manera fundamental, los algoritmos de ramificación y poda se basan en encontrar el mínimo de una determinada función.

Suele ser habitual combinar los distintos métodos presentando métodos híbridos entre programación dinámica y métodos de ramificación y poda [20] [21] [22] [23].

- Algoritmo de la colonia de hormigas.

Los algoritmos de colonia de hormigas (ACO, por sus siglas en inglés Ant Colony Optimization) son algoritmos probabilísticos empleados para resolver problemas que pueden reducirse a buscar los caminos más cortos en grafos, basándose en el comportamiento de las hormigas cuando estas buscan un camino entre su colonia y una fuente de alimentos [24].

En la naturaleza, las hormigas vagan inicialmente de una manera aleatoria, y una vez encuentran el alimento, regresan a su colonia dejando un rastro de feromonas. Si otras hormigas encuentran ese rastro, es probable que dejen de caminar aleatoriamente y empiecen a seguir el rastro de feromonas. De esta manera, fortalecen el camino.

Sin embargo, pasado un cierto tiempo, las feromonas comienzan a evaporarse. Por tanto, cuánto más tiempo le tome a una hormiga viajar por el camino y regresar a la colonia, más tiempo tienen las feromonas para evaporarse. Es decir, los caminos más cortos son más atractivos para las hormigas, pues en ellos la densidad de feromonas es mayor que en los caminos largos. La evaporación de feromonas tiene la ventaja de evitar la convergencia a óptimos locales.

Cuando una hormiga encuentra un buen camino entre la colonia y la fuente de comida, hay más posibilidades de que las demás hormigas se vean atraídas por ese camino y con una retroalimentación positiva, finalmente acaba llevando a todas las hormigas a un solo camino [25].

La idea del algoritmo de colonia de hormigas es imitar este comportamiento con "hormigas simuladas" caminando a través de un grafo que representa el problema en cuestión. La idea original se ha diversificado obteniendo distintas clases de algoritmos para resolver problemas de flujo máximo a coste mínimo con costes cóncavos [26] [27] [28].

- Algoritmos genéticos.

Los algoritmos genéticos (GA, por sus siglas en inglés, Genetic Algorithms) se inspiran en la evolución biológica y la genética molecular. En esencia, en estos algoritmos, se considera una población de soluciones candidatas (que consituyen el fenotipo) con una serie de propiedades o características (cromosomas o genotipo) que pueden evolucionar o mutar iterativamente (generaciones).

En general, los GAs parten de una población inicial, constituida por un conjunto aleatorio de soluciones factibles (cromosomas). Cada cromosoma se somete a una evaluación para identificar cómo de buena es esa solución.

Cada generación (iteración) consta de las siguientes fases:

1. Selección de los cromosomas candidatos con mejor evaluación.
2. Aplicación de los operadores genéticos (recombinación y/o mutación) a los cromosomas seleccionados para generar descendientes (nuevas soluciones).

El algoritmo se detiene bien cuando se alcanza un determinado número de generaciones o bien cuando no haya cambios significativos en la población. Es decir, cuando se alcance un número máximo de iteraciones o cuando en dos iteraciones consecutivas, la población no varíe.

Los algoritmos genéticos son eficaces para optimizar funciones con expresiones complejas [8]

La bibliografía recoge numerosas versiones dentro de estos algoritmos genéticos: enfocados en redes a pequeña escala [29] [30], métodos híbridos GA con búsqueda local [31], o con representación en árbol [8], entre otros [32] [33] [34].

5 CONCLUSIONES

Como se ha indicado en capítulos previos, existen numerosos problemas reales que pueden modelarse matemáticamente como un problema de flujo máximo a coste mínimo. En general, las dimensiones de estos problemas reales suelen hacer inviable su resolución a mano, por lo que los algoritmos deben implementarse en algún tipo de software matemático.

Una vez comprobado el funcionamiento del código con numerosos ejemplos resueltos previamente a mano, es posible concluir que, con la realización de este Trabajo Fin de Máster, se ha conseguido implementar en MATLAB de manera correcta el algoritmo Primal-Dual aplicado a la Tabla de transporte para la resolución del problema de flujo máximo a coste mínimo tanto para costes constantes como para costes dependientes del flujo (algoritmos PDTCI y PDTCD).

El lenguaje utilizado en la programación de los algoritmos es muy sencillo, empleando funciones fácilmente comprensibles para cualquier usuario con nivel básico en MATLAB que pretenda emplear el código de este Trabajo Fin de Máster para resolver redes de transporte. A la hora de programar el algoritmo no se ha tenido en cuenta la eficiencia del mismo, sino que se ha buscado la efectividad, es decir, que resuelva correctamente el problema.

En lo que al código implementado respecta, cabe plantear algunos aspectos de mejora:

- Una vez resuelto el problema, el código no tiene en cuenta la presencia de posibles óptimos alternativos. Por ello, un punto de mejora podría ser la programación de una función que determine los óptimos alternativos a partir del flujo óptimo (sin post-procesado) y de los costes relativos de los arcos, empleando el algoritmo de Hitchcock. Incluso podría pensarse en una función que calcule subóptimos cercanos al óptimo (y que pueden ser de utilidad en algunas circunstancias concretas).
- Como se ha indicado en párrafos anteriores, la implementación del algoritmo se ha realizado con un lenguaje muy elemental y no se ha tenido en cuenta la eficiencia del código. Se deja abierta la posibilidad de optar por otras estrategias de programación empleando funciones de MATLAB más avanzadas que permitan reducir la extensión del código necesario para la resolución del problema. De esta manera, MATLAB necesitaría un menor número de operaciones para resolver el problema, mejorando la eficiencia del código.
- A la vista de los ejemplos expuestos en los Apartados 2.6, 3.7 y en los Apéndices B y C, la función "asignaflujo" normalmente no aporta resultados nuevos a la resolución del problema (excepto en la primera iteración del algoritmo). Tal como se indicó en el Apartado 2.3.2, la razón por la que en este Trabajo se ha incluido es para mantener un proceso de resolución que sea completamente análogo a la resolución a mano del problema, en la que en cada iteración se comprueba si puede asignarse algún flujo de manera directa sobre las celdas básicas. Sin embargo, esa asignación directa puede sustituirse fácilmente con el proceso de marcaje y aumento de flujo del algoritmo Ford-Fulkerson. En este sentido, el código puede adaptarse de manera sencilla para eliminar esta función implementada. Para redes pequeñas, esta eliminación no haría variar significativamente el tiempo de ejecución del algoritmo, pero es lógico pensar que, para redes de gran escala, esta modificación sí mejoraría el tiempo de ejecución.

Por otra parte, con relación a los distintos algoritmos presentados en el Capítulo 4:

- La elección de un determinado algoritmo a la hora de resolver el problema de flujo máximo a coste mínimo no debe ser aleatoria, pues existen algoritmos más eficientes que otros desde el punto de vista computacional. Los algoritmos fuertemente polinomiales son muy eficientes para resolver redes a gran escala y, por tanto, éstos deben priorizarse sobre otros algoritmos básicos.
- En general, los investigadores han encontrado multitud de versiones de algoritmos derivados de otros (ya sea generalizando algoritmos más básicos o combinando técnicas presentes en distintos algoritmos) que pueden ser más o menos eficientes para redes con costes no lineales. Esto ha supuesto y supone en la actualidad una interesante vía de investigación sobre nuevos algoritmos que resuelvan el problema de flujo máximo a coste mínimo con el menor numero de cálculos posibles. La tendencia en las investigaciones es la de encontrar algoritmos competitivos que resuelvan problemas complejos optimizando su coste computacional (tiempo de ejecución y memoria requerida).

En resumidas cuentas, este Trabajo Fin de Máster proporciona una herramienta informática (en forma de programa de MATLAB) para resolver el problema de flujo máximo a coste mínimo, tanto para costes constantes como para costes dependientes del flujo (algoritmos PDTCI y PDTCD). En este sentido, los Capítulos 2 y 3 de este Trabajo pueden entenderse como un "manual de usuario extendido" para entender cómo usar dicha herramienta informática y conocer su funcionamiento interno de manera detallada.

Este programa puede emplearse para la resolución de redes de transporte tanto a nivel académico como a niveles profesionales más avanzados.

APÉNDICE A.- SCRIPTS DE MATLAB

El script del programa, junto con un manual de usuario, puede proporcionarse solícitandolo al autor.

APÉNDICE B.- RESOLUCIÓN COMPLETA DE UN EJEMPLO (ALGORITMO PDTCI)

Con el objetivo de completar el Capítulo 2 de este Trabajo, se presenta en este Apéndice un segundo problema de flujo máximo a costo mínimo con costes constantes, que se resolverá empleando el algoritmo PDTCI implementado en MATLAB de manera análoga al ejemplo presentado en el Apartado 2.6.

Se resolverá la red representada por el siguiente grafo:

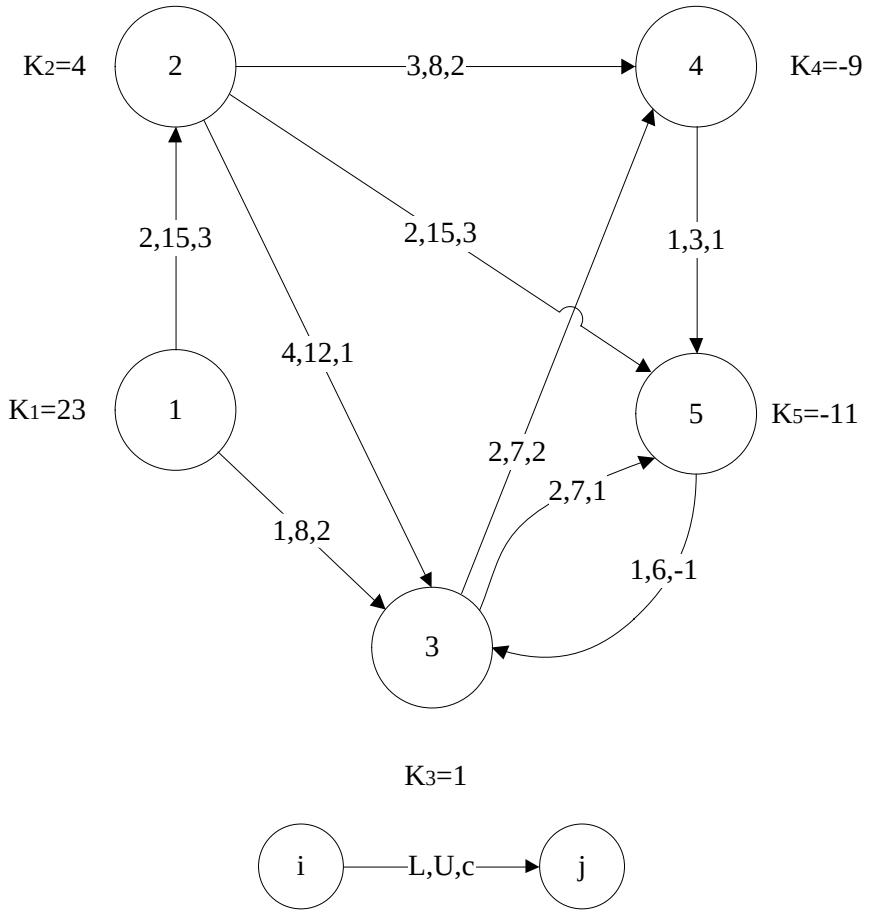


Figura 48.- Red de transporte a resolver en el Apéndice B.

Para esta red, la matriz "arcos" y el vector "K" correspondientes son:

$$arcos = \begin{bmatrix} 1 & 2 & 2 & 15 & 3 \\ 1 & 3 & 1 & 8 & 2 \\ 2 & 3 & 4 & 12 & 1 \\ 2 & 4 & 3 & 8 & 2 \\ 2 & 5 & 2 & 15 & 3 \\ 3 & 4 & 2 & 7 & 2 \\ 3 & 5 & 2 & 7 & 1 \\ 4 & 5 & 1 & 3 & 1 \\ 5 & 3 & 1 & 6 & -1 \end{bmatrix}$$

$$K = [23 \quad 4 \quad 1 \quad -9 \quad -11]$$

A continuación, se indican todos los pasos que el algoritmo PDTCI ejecuta hasta la resolución de este problema, empleando cada uno de los tres métodos de estandarización.

B.1. Método 1.

Al ejecutar

```
arcos=[1 2 2 15 3;1 3 1 8 2;2 3 4 12 1;2 4 3 8 2;2 5 2 15 3;3 4 2 7 2;3 5 2
7 1;4 5 1 3 1;5 3 1 6 -1];
K=[23 4 1 -9 -11];
[U,C,of,dem]=estandarizar(arcos,K,1)
```

Se obtienen las siguientes matrices y vectores:

```
U =

   Inf    13     7   Inf   Inf   Inf
   Inf   Inf     8     5    13   Inf
   Inf   Inf   Inf     5     5   Inf
   Inf   Inf   Inf   Inf     2   Inf
   Inf   Inf     5   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf

C =

     0     3     2   Inf   Inf     0
   Inf     0     1     2     3   Inf
   Inf   Inf     0     2     1     0
   Inf   Inf   Inf     0     1   Inf
   Inf   Inf    -1   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0

of =

    43    23    26    23    23    23

dem =

    23    26    23    28    30    31
```

De esta manera, MATLAB ha estandarizado la red de la Figura 48 según el método 1, lo que implica que la red se equilibre añadiendo un nodo ficticio de demanda que absorba el exceso de oferta de 8 unidades que existe en dicha red (además de realizar el cambio de variable asociado a eliminar la cota inferior y calcular los valores "k"). Es decir:

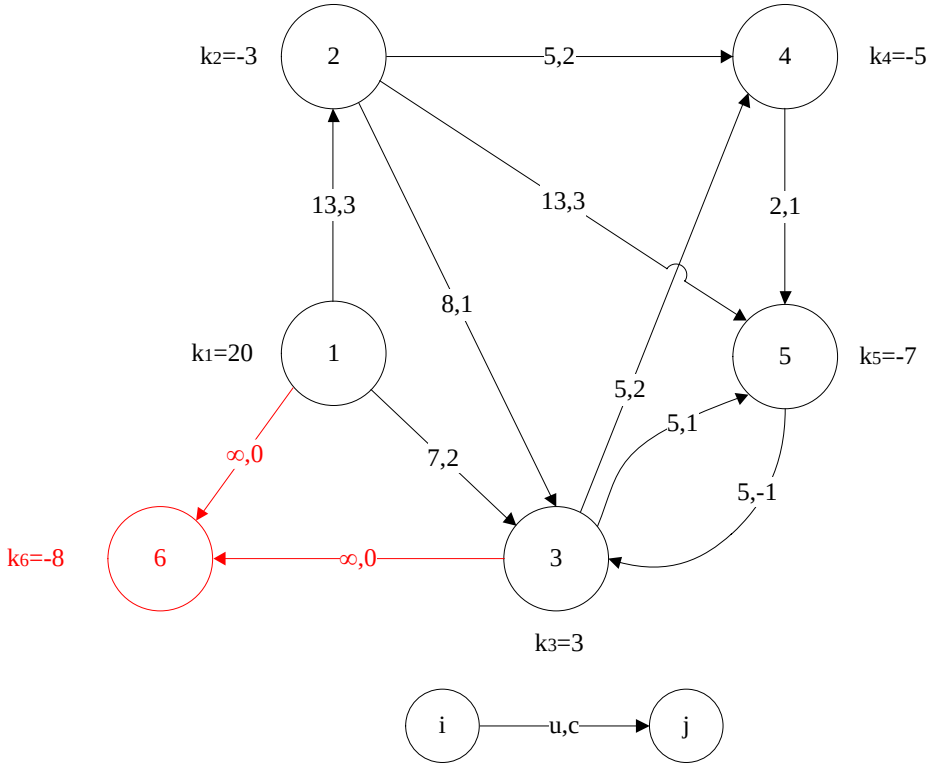


Figura 49.- Grafo de la Figura 48, estandarizado mediante el método 1.

Además, MATLAB representa el siguiente gráfico de la red, indicando los costes de cada arco.

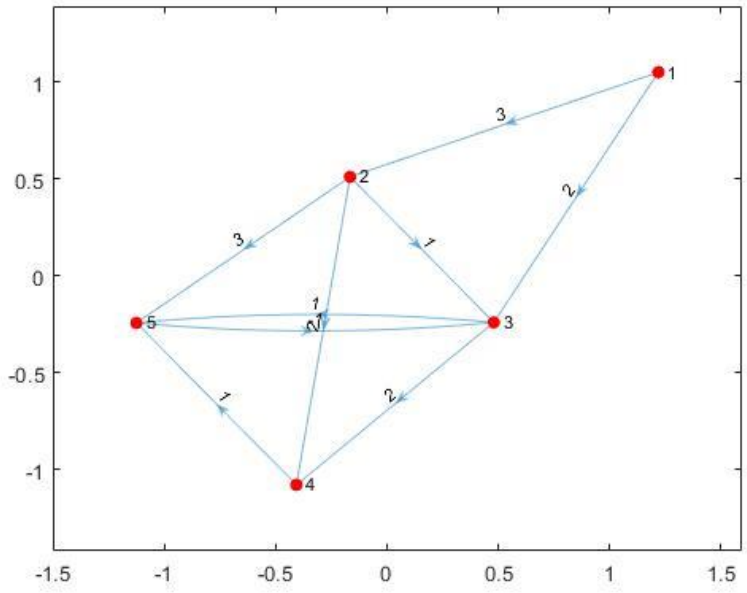


Figura 50.- Grafo de la Figura 48, representado en MATLAB.

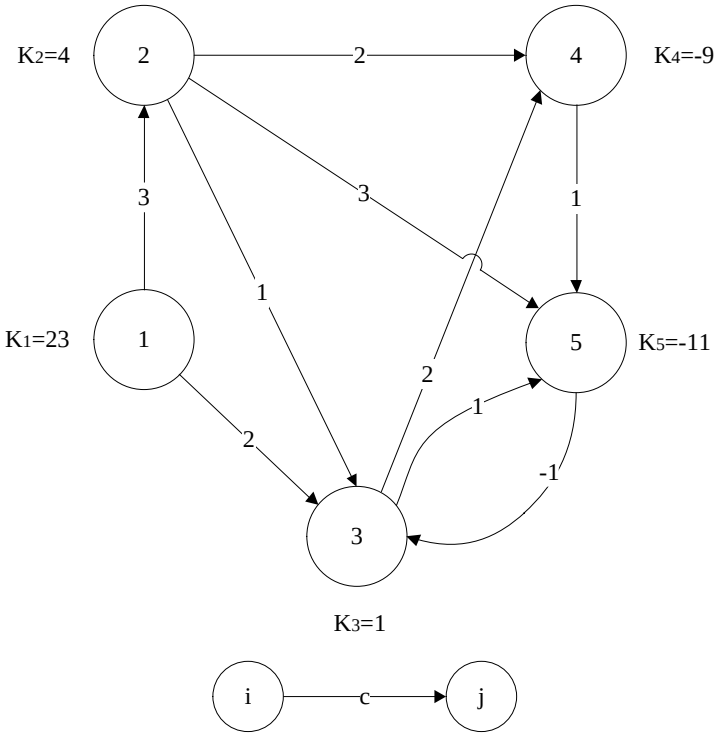


Figura 51.- Grafo realista equivalente al grafo de la Figura 50.

Este grafo que aparece en pantalla, corresponde al grafo de la Figura 48, indicando únicamente los costes de cada arco, por lo que es independiente del método de estandarización empleado. Por ello, se omitirá este grafo en el desarrollo de la resolución del problema según los otros dos métodos.

Una vez estandarizado el grafo, puede ejecutarse el algoritmo PDTCI, cuya resolución paso a paso ofrece los siguientes resultados intermedios (que se indican sin comentar)

```
[flujo,mr]=pdtci(C,of,dem,U)
```

```
ITERACIÓN 1
Costos relativos
     0     3     2   Inf   Inf     0
   Inf     0     1     2     2   Inf
   Inf   Inf     0     2     0     0
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     0   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0
```

```
Flujo actual
    23     0     0     0     0    20
     0    23     0     0     0     0
     0     0    23     0     3     0
     0     0     0    23     0     0
     0     0     0     0    23     0
     0     0     0     0     0    11

Oferta y demanda
     0
     0
     0
     0
     0
    12

     0     3     0     5     4     0
Marcaje del algoritmo FF
     6    12     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0    12     1

     1     0     0     0     0     6
    12     0     0     0     0    12
     1     0     0     0     0     1


ITERACIÓN 2
Costos relativos
     0     1     0   Inf   Inf     0
   Inf     0     1     2     2   Inf
   Inf   Inf     0     2     0     2
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     0   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6    12     1
     0     0     0
     3     7     1
     0     0     0
     5     2     1
     0    12     1

     1     0     1     0     3     6
    12     0     7     0     2    12
     1     0     1     0     1     1
```

```
Flujo actual
    23     0     2     0     0    18
     0    23     0     0     0     0
     0     0    21     0     5     0
     0     0     0    23     0     0
     0     0     0     0    23     0
     0     0     0     0     0    13

Oferta y demanda
     0
     0
     0
     0
     0
    10

     0     3     0     5     2     0


 ITERACIÓN 3
 Costos relativos
     0     1     0   Inf   Inf     0
   Inf     0     1     2     2   Inf
   Inf   Inf     0     2     0     2
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     0   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0

 Marcaje del algoritmo FF
     6    10     1
     0     0     0
     3     5     1
     0     0     0
     0     0     0
     0    10     1

     1     0     1     0     3     6
    10     0     5     0     0    10
     1     0     1     0     0     1


 ITERACIÓN 4
 Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     2     2   Inf
   Inf   Inf     0     1    -1     2
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     1   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0
```

```
Marcaje del algoritmo FF
     6     10      1
     2     10      1
     3      5      1
     0      0      0
     0      0      0
     0     10      1

     1      1      1      0      0      6
    10     10      5      0      0     10
     1      1      1      0      0      1

Flujo actual
    23      3      2      0      0     15
     0     23      0      0      0      0
     0      0     21      0      5      0
     0      0      0     23      0      0
     0      0      0      0     23      0
     0      0      0      0      0     16

Oferta y demanda
     0
     0
     0
     0
     0
     7

     0      0      0      5      2      0


ITERACIÓN 5
Costos relativos
     0      0      0    Inf    Inf      0
   Inf      0      2      2      2    Inf
   Inf    Inf      0      1     -1      2
   Inf    Inf    Inf      0      0    Inf
   Inf    Inf      1    Inf      0    Inf
   Inf    Inf    Inf    Inf    Inf      0

Marcaje del algoritmo FF
     6      7      1
     2      7      1
     3      5      1
     0      0      0
     0      0      0
     0      7      1

     1      1      1      0      0      6
     7      7      5      0      0      7
     1      1      1      0      0      1
```

```
ITERACIÓN 6
Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     1     1   Inf
   Inf   Inf     0     0    -2     2
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     2   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6     7     1
     2     7     1
     3     5     1
     4     5     1
     5     2     1
     0     7     1

     1     1     1     3     4     6
     7     7     5     5     2     7
     1     1     1     1     1     1

Flujo actual
    23     3     4     0     0    13
     0    23     0     0     0     0
     0     0    19     2     5     0
     0     0     0    21     2     0
     0     0     0     0    23     0
     0     0     0     0     0    18

Oferta y demanda
     0
     0
     0
     0
     0
     5

     0     0     0     5     0     0


ITERACIÓN 7
Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     1     1   Inf
   Inf   Inf     0     0    -2     2
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     2   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0
```

```
Marcaje del algoritmo FF
     6     5     1
     2     5     1
     3     3     1
     4     3     1
     0     0     0
     0     5     1

     1     1     1     3     4     6
     5     5     3     3     0     5
     1     1     1     1     0     1

Flujo actual
    23     3     7     0     0    10
     0    23     0     0     0     0
     0     0    16     5     5     0
     0     0     0    21     2     0
     0     0     0     0    23     0
     0     0     0     0     0    21

Oferta y demanda
     0
     0
     0
     0
     0
     2

     0     0     0     2     0     0
ITERACIÓN 8
Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     1     1   Inf
   Inf   Inf     0     0    -2     2
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     2   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6     2     1
     2     2     1
     0     0     0
     0     0     0
     0     0     0
     0     2     1

     1     1     1     0     0     6
     2     2     0     0     0     2
     1     1     0     0     0     1
```

```
ITERACIÓN 9
Costos relativos
     0     0    -1   Inf   Inf     0
   Inf     0     1     0     0   Inf
   Inf   Inf     0     0    -2     3
   Inf   Inf   Inf     0     0   Inf
   Inf   Inf     2   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6     2     1
     2     2     1
     4     2     1
     5     2     1
     5     2     1
     0     2     1

     1     1     3     2     2     6
     2     2     2     2     2     2
     1     1     1     1     1     1
```

Finalmente, se obtiene el óptimo:

```
flujo =

    23     5     7     0     0     8
     0    21     0     2     0     0
     0     0    16     5     5     0
     0     0     0    21     2     0
     0     0     0     0    23     0
     0     0     0     0     0    23
```

El post-procesado permite deshacer los cambios de variable, obteniendo el flujo óptimo sobre la red original.

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

```
flujoreal =

    23     7     8     0     0     8
     0    21     4     5     2     0
     0     0    16     7     7     0
     0     0     0    21     3     0
     0     0     1     0    23     0
     0     0     0     0     0    23

Kreales =

    15     4     1    -9   -11     0

z =

    80
```

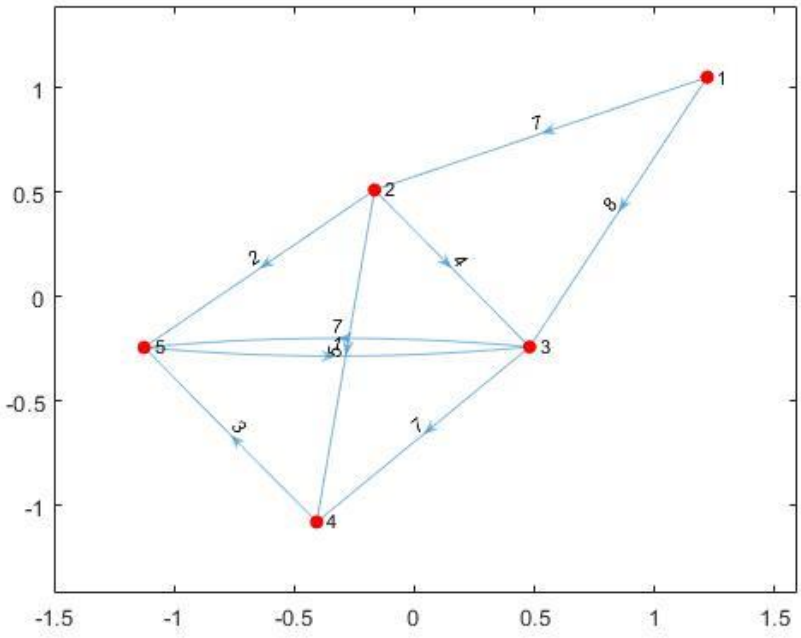


Figura 52.- Flujo óptimo en la red de la Figura 48 (método 1).

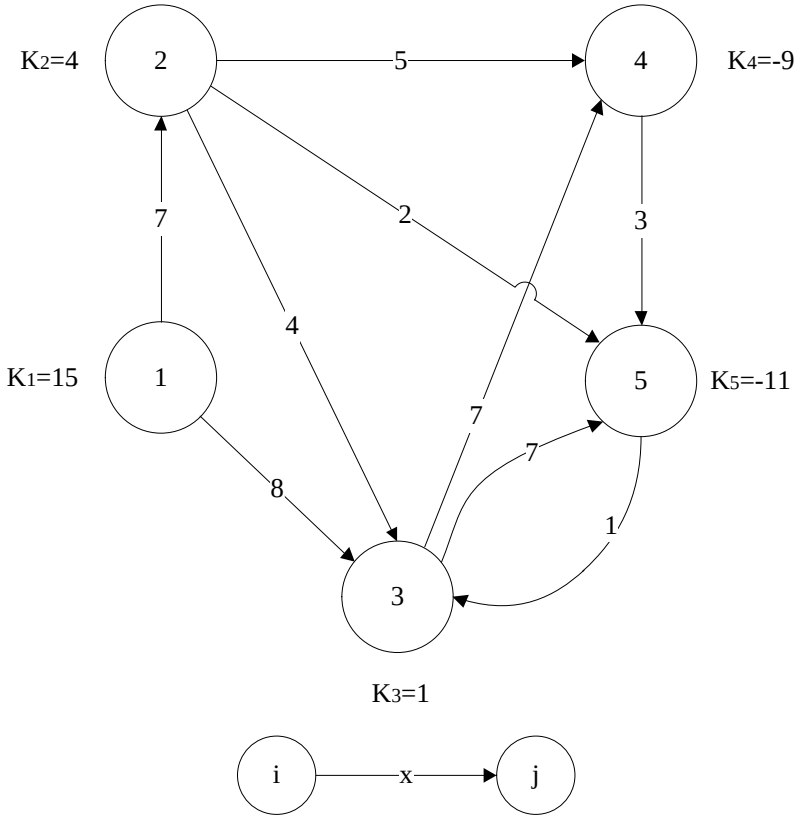


Figura 53.- Grafo realista equivalente al grafo de la Figura 52.

B.2. Método 2.

Al ejecutar

```
arcos=[1 2 2 15 3;1 3 1 8 2;2 3 4 12 1;2 4 3 8 2;2 5 2 15 3;3 4 2 7 2;3 5 2
7 1;4 5 1 3 1;5 3 1 6 -1];
K=[23 4 1 -9 -11];
[U,C,of,dem]=estandarizar(arcos,K,2)
```

MATLAB estandariza la red según el método 2, lo que implica la creación de un nodo ficticio adicional que absorba el exceso de oferta de 8 unidades, y que estará unido mediante arcos ficitios a todos los nodos de la red. Es decir:

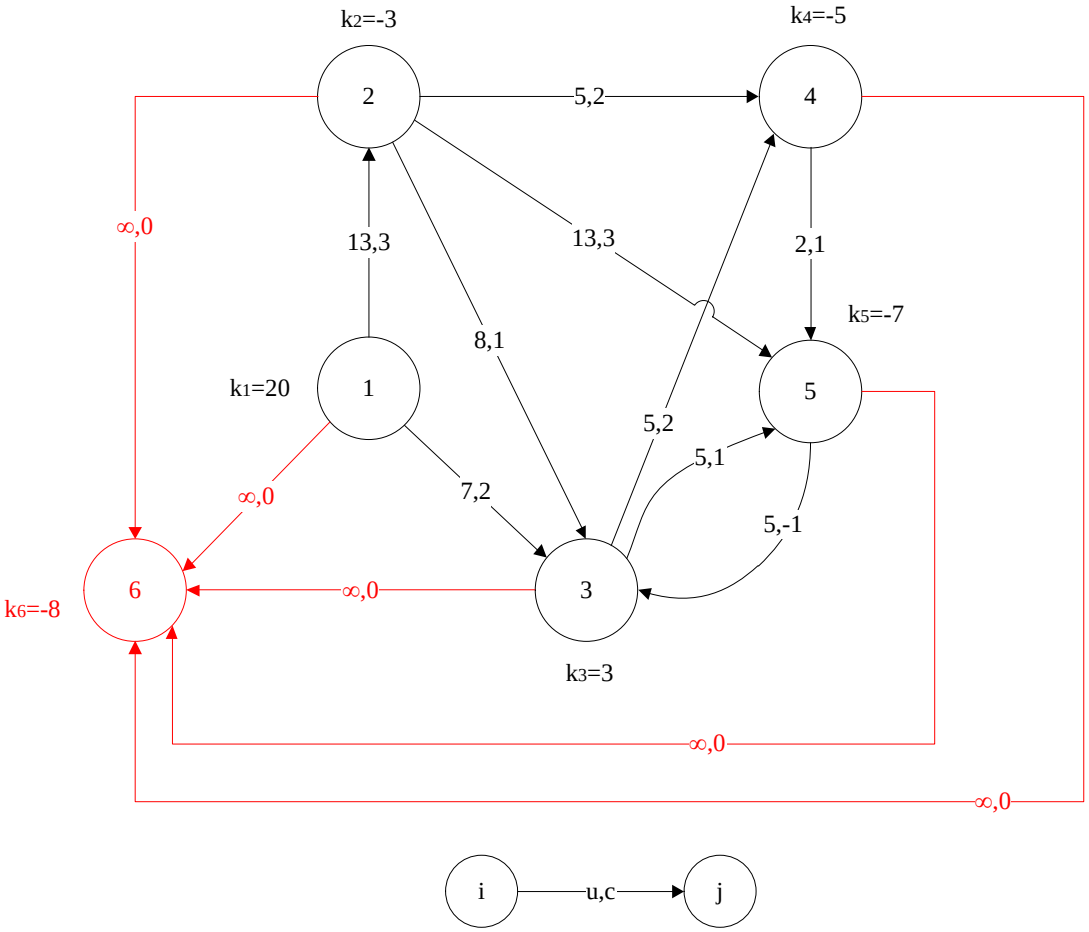


Figura 54.- Grafo de la Figura 48, estandarizado mediante el método 2.

Se obtienen las matrices U y C, y los vectores de oferta y demanda siguientes:

```
U =

   Inf    13     7   Inf   Inf   Inf
   Inf   Inf     8     5    13   Inf
   Inf   Inf   Inf     5     5   Inf
   Inf   Inf   Inf   Inf     2   Inf
   Inf   Inf     5   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf


C =

     0     3     2   Inf   Inf     0
   Inf     0     1     2     3     0
   Inf   Inf     0     2     1     0
   Inf   Inf   Inf     0     1     0
   Inf   Inf    -1   Inf     0     0
   Inf   Inf   Inf   Inf   Inf     0


of =

    43    23    26    23    23    23


dem =

    23    26    23    28    30    31
```

En este punto, puede ejecutarse la función “pdtci”, obteniendo las siguientes matrices intermedias:

```
[flujo,mr]=pdtci(C,of,dem,U)
```

```
ITERACIÓN 1
Costos relativos
     0     3     2   Inf   Inf     0
   Inf     0     1     2     2     0
   Inf   Inf     0     2     0     0
   Inf   Inf   Inf     0     0     0
   Inf   Inf     0   Inf     0     1
   Inf   Inf   Inf   Inf   Inf     0
```

```
Flujo actual
    23     0     0     0     0    20
     0    23     0     0     0     0
     0     0    23     0     3     0
     0     0     0    23     0     0
     0     0     0     0    23     0
     0     0     0     0     0    11

Oferta y demanda
     0
     0
     0
     0
     0
    12

     0     3     0     5     4     0

Marcaje del algoritmo FF
     6    12     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0    12     1

     1     0     0     0     0     6
    12     0     0     0     0    12
     1     0     0     0     0     1


ITERACIÓN 2
Costos relativos
     0     1     0   Inf   Inf     0
   Inf     0     1     2     2     2
   Inf   Inf     0     2     0     2
   Inf   Inf   Inf     0     0     2
   Inf   Inf     0   Inf     0     3
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6    12     1
     0     0     0
     3     7     1
     0     0     0
     5     2     1
     0    12     1

     1     0     1     0     3     6
    12     0     7     0     2    12
     1     0     1     0     1     1
```

```
Flujo actual
    23     0     2     0     0    18
     0    23     0     0     0     0
     0     0    21     0     5     0
     0     0     0    23     0     0
     0     0     0     0    23     0
     0     0     0     0     0    13

Oferta y demanda
     0
     0
     0
     0
     0
    10

     0     3     0     5     2     0


ITERACIÓN 3
Costos relativos
     0     1     0   Inf   Inf     0
   Inf     0     1     2     2     2
   Inf   Inf     0     2     0     2
   Inf   Inf   Inf     0     0     2
   Inf   Inf     0   Inf     0     3
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6    10     1
     0     0     0
     3     5     1
     0     0     0
     0     0     0
     0    10     1

     1     0     1     0     3     6
    10     0     5     0     0    10
     1     0     1     0     0     1


ITERACIÓN 4
Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     2     2     3
   Inf   Inf     0     1    -1     2
   Inf   Inf   Inf     0     0     3
   Inf   Inf     1   Inf     0     4
   Inf   Inf   Inf   Inf   Inf     0
```

```
Marcaje del algoritmo FF
     6    10     1
     2    10     1
     3     5     1
     0     0     0
     0     0     0
     0    10     1

     1     1     1     0     0     6
    10    10     5     0     0    10
     1     1     1     0     0     1
Flujo actual
    23     3     2     0     0    15
     0    23     0     0     0     0
     0     0    21     0     5     0
     0     0     0    23     0     0
     0     0     0     0    23     0
     0     0     0     0     0    16

Oferta y demanda
     0
     0
     0
     0
     0
     7

     0     0     0     5     2     0


  ITERACIÓN 5
  Costos relativos
       0     0     0   Inf   Inf     0
     Inf     0     2     2     2     3
     Inf   Inf     0     1    -1     2
     Inf   Inf   Inf     0     0     3
     Inf   Inf     1   Inf     0     4
     Inf   Inf   Inf   Inf   Inf     0

  Marcaje del algoritmo FF
       6     7     1
       2     7     1
       3     5     1
       0     0     0
       0     0     0
       0     7     1

       1     1     1     0     0     6
       7     7     5     0     0     7
       1     1     1     0     0     1
```

```
ITERACIÓN 6
Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     1     1     3
   Inf   Inf     0     0    -2     2
   Inf   Inf   Inf     0     0     4
   Inf   Inf     2   Inf     0     5
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6     7     1
     2     7     1
     3     5     1
     4     5     1
     5     2     1
     0     7     1

     1     1     1     3     4     6
     7     7     5     5     2     7
     1     1     1     1     1     1

Flujo actual
    23     3     4     0     0    13
     0    23     0     0     0     0
     0     0    19     2     5     0
     0     0     0    21     2     0
     0     0     0     0    23     0
     0     0     0     0     0    18

Oferta y demanda
     0
     0
     0
     0
     0
     5

     0     0     0     5     0     0


ITERACIÓN 7
Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     1     1     3
   Inf   Inf     0     0    -2     2
   Inf   Inf   Inf     0     0     4
   Inf   Inf     2   Inf     0     5
   Inf   Inf   Inf   Inf   Inf     0
```

```
Marcaje del algoritmo FF
     6     5     1
     2     5     1
     3     3     1
     4     3     1
     0     0     0
     0     5     1

     1     1     1     3     4     6
     5     5     3     3     0     5
     1     1     1     1     0     1

Flujo actual
    23     3     7     0     0    10
     0    23     0     0     0     0
     0     0    16     5     5     0
     0     0     0    21     2     0
     0     0     0     0    23     0
     0     0     0     0     0    21

Oferta y demanda
     0
     0
     0
     0
     0
     2

     0     0     0     2     0     0


ITERACIÓN 8
Costos relativos
     0     0     0   Inf   Inf     0
   Inf     0     2     1     1     3
   Inf   Inf     0     0    -2     2
   Inf   Inf   Inf     0     0     4
   Inf   Inf     2   Inf     0     5
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6     2     1
     2     2     1
     0     0     0
     0     0     0
     0     0     0
     0     2     1

     1     1     1     0     0     6
     2     2     0     0     0     2
     1     1     0     0     0     1
```

```
ITERACIÓN 9
Costos relativos
     0     0    -1   Inf   Inf     0
   Inf     0     1     0     0     3
   Inf   Inf     0     0    -2     3
   Inf   Inf   Inf     0     0     5
   Inf   Inf     2   Inf     0     6
   Inf   Inf   Inf   Inf   Inf     0

Marcaje del algoritmo FF
     6     2     1
     2     2     1
     4     2     1
     5     2     1
     5     2     1
     0     2     1

     1     1     3     2     2     6
     2     2     2     2     2     2
     1     1     1     1     1     1
```

Finalmente se alcanza el óptimo:

```
flujo =

    23     5     7     0     0     8
     0    21     0     2     0     0
     0     0    16     5     5     0
     0     0     0    21     2     0
     0     0     0     0    23     0
     0     0     0     0     0    23
```

Nuevamente, con el post-procesado, se deshace el cambio de variable y se obtiene el óptimo sobre la red original:

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

```
flujoreal =

    23     7     8     0     0     8
     0    21     4     5     2     0
     0     0    16     7     7     0
     0     0     0    21     3     0
     0     0     1     0    23     0
     0     0     0     0     0    23

Kreales =

    15     4     1    -9   -11     0

z =

    80
```

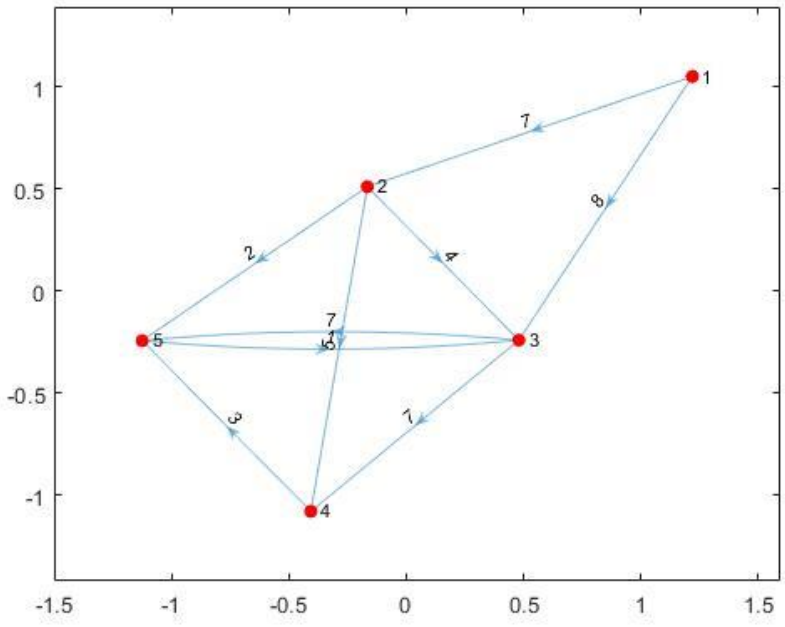


Figura 55.- Flujo óptimo en la red de la Figura 48 (método 2).

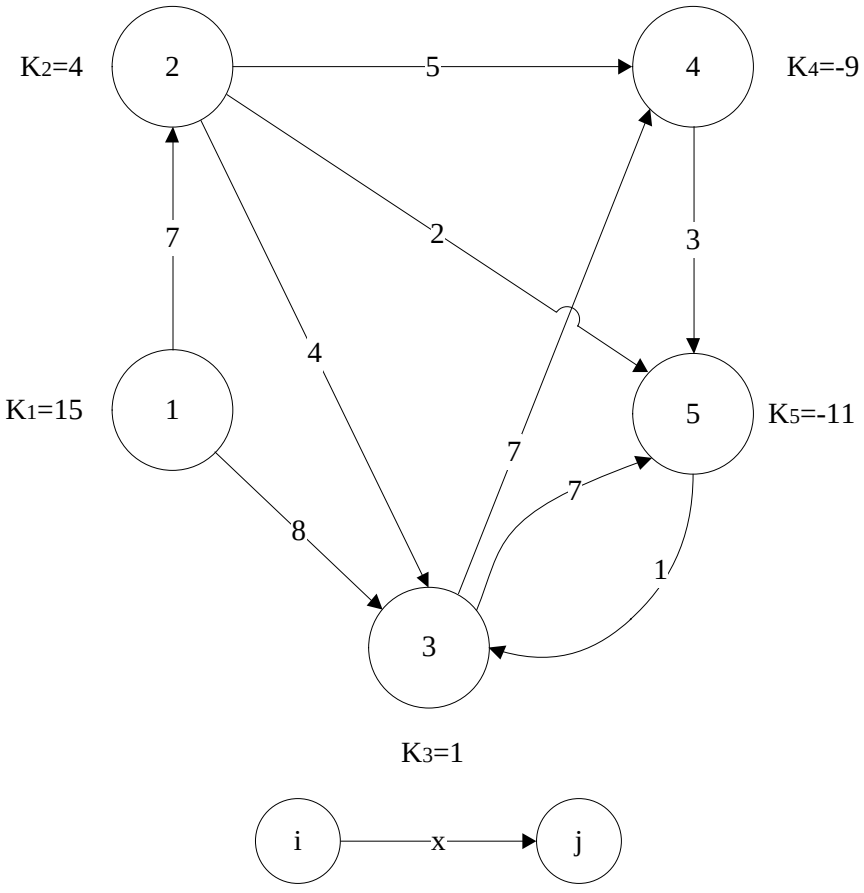


Figura 56.- Grafo realista equivalente al grafo de la Figura 55.

B.3. Método 3.

Al ejecutar

```
arcos=[1 2 2 15 3;1 3 1 8 2;2 3 4 12 1;2 4 3 8 2;2 5 2 15 3;3 4 2 7 2;3 5 2
7 1;4 5 1 3 1;5 3 1 6 -1];
K=[23 4 1 -9 -11];
[U,C,of,dem]=estandarizar(arcos,K,3)
```

MATLAB ha estandarizado la red de la Figura 48 empleando el método 3, lo que implica la creación de un nodo de salida unido mediante arcos ficitios a todos los nodos de oferta y un nodo de entrada unido mediante arcos ficticios a todos los nodos de demanda. Es decir:

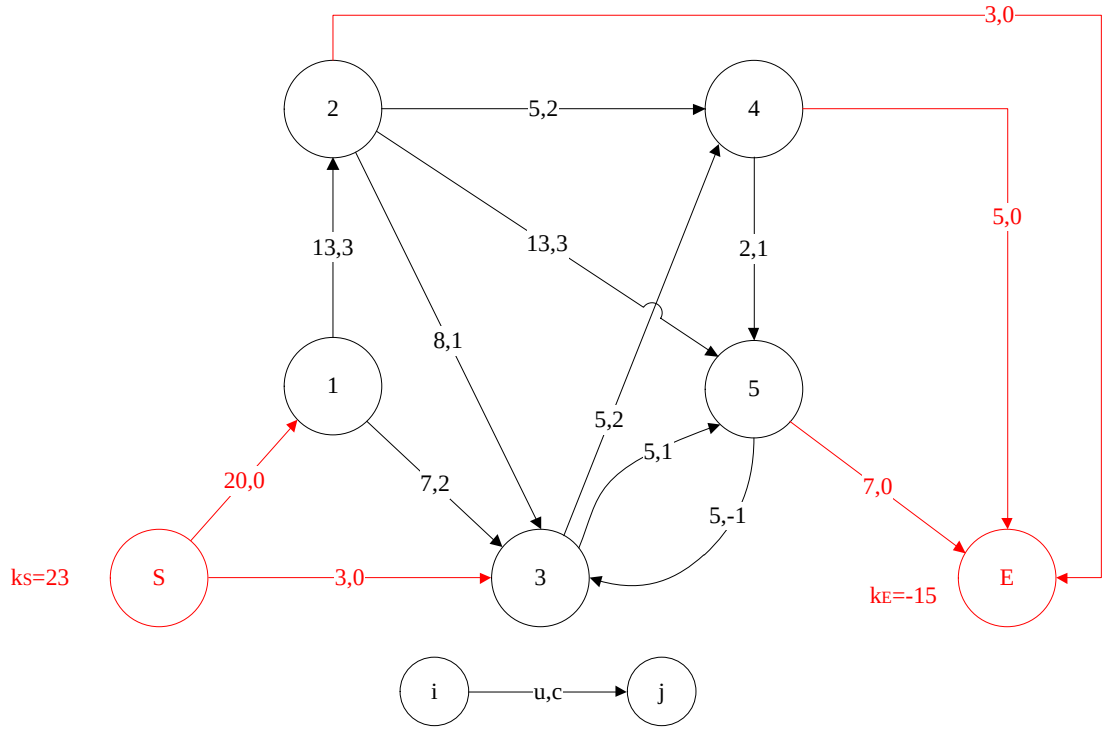


Figura 57.- Grafo de la Figura 48, estandarizado mediante el método 3.

Se obtienen las siguientes matrices y vectores:

```
U =                                          C =

   Inf    13     7   Inf   Inf   Inf          0     3     2   Inf   Inf   Inf
   Inf   Inf     8     5    13     3        Inf     0     1     2     3     0
   Inf   Inf   Inf     5     5   Inf        Inf   Inf     0     2     1   Inf
   Inf   Inf   Inf   Inf     2     5        Inf   Inf   Inf     0     1     0
   Inf   Inf     5   Inf   Inf     7        Inf   Inf    -1   Inf     0     0
    20   Inf     3   Inf   Inf   Inf          0   Inf     0   Inf   Inf   Inf
```

```
of =

    15    15    15    15    15    30


dem =

    15    15    15    15    15    30
```

Ahora se ejecuta la función “pdtci”, con los siguientes resultados intermedios:

```
[flujo,mr]=pdtci(C,of,dem,U)
```

```
ITERACIÓN 1
Costos relativos
     0     3     2   Inf   Inf   Inf
   Inf     0     1     2     2     0
   Inf   Inf     0     2     0   Inf
   Inf   Inf   Inf     0     0     0
   Inf   Inf     0   Inf     0     1
     0   Inf     0   Inf   Inf   Inf
Flujo actual
    15     0     0     0     0     0
     0    15     0     0     0     0
     0     0    15     0     0     0
     0     0     0    15     0     0
     0     0     0     0    15     0
     0     0     0     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
    30

     0     0     0     0     0    30
```

```
Marcaje del algoritmo FF
     1    15     1
     0     0     0
     3     3     1
     0     0     0
     5     3     1
     0    30     1

     6     0     6     0     3     0
    20     0     3     0     3     0
     1     0     1     0     1     0


ITERACIÓN 2
Costos relativos
     0     2     2   Inf   Inf   Inf
   Inf     0     2     2     3     0
   Inf   Inf     0     1     0   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     0   Inf     0     0
     0   Inf     0   Inf   Inf   Inf

Marcaje del algoritmo FF
     1    15     1
     0     0     0
     3     3     1
     0     0     0
     5     3     1
     0    30     1

     6     0     6     0     3     5
    20     0     3     0     3     3
     1     0     1     0     1     1

Flujo actual
    15     0     0     0     0     0
     0    15     0     0     0     0
     0     0    12     0     3     0
     0     0     0    15     0     0
     0     0     0     0    12     3
     0     0     3     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
    27

     0     0     0     0     0    27
```

```
ITERACIÓN 3
Costos relativos
     0     2     2   Inf   Inf   Inf
   Inf     0     2     2     3     0
   Inf   Inf     0     1     0   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     0   Inf     0     0
     0   Inf     0   Inf   Inf   Inf

Marcaje del algoritmo FF
     1    15     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0    27     1

     6     0     6     0     0     0
    20     0     0     0     0     0
     1     0     0     0     0     0


ITERACIÓN 4
Costos relativos
     0     0     0   Inf   Inf   Inf
   Inf     0     2     2     3     0
   Inf   Inf     0     1     0   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     0   Inf     0     0
     0   Inf    -2   Inf   Inf   Inf

Marcaje del algoritmo FF
     1    15     1
     2    13     1
     3     7     1
     0     0     0
     6     3     1
     0    27     1

     6     1     1     0     3     2
    20    13     7     0     2     3
     1     1     1     0     1     1
```

```
Flujo actual
    12     3     0     0     0     0
     0    12     0     0     0     3
     0     0    12     0     3     0
     0     0     0    15     0     0
     0     0     0     0    12     3
     3     0     3     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
    24

     0     0     0     0     0    24


ITERACIÓN 5
Costos relativos
     0     0     0   Inf   Inf   Inf
   Inf     0     2     2     3     0
   Inf   Inf     0     1     0   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     0   Inf     0     0
     0   Inf    -2   Inf   Inf   Inf

Marcaje del algoritmo FF
     1    12     1
     2    10     1
     3     7     1
     0     0     0
     5     2     1
     0    24     1

     6     1     1     0     3     5
    17    10     7     0     2     2
     1     1     1     0     1     1
```

```
Flujo actual
    10     3     2     0     0     0
     0    12     0     0     0     3
     0     0    10     0     5     0
     0     0     0    15     0     0
     0     0     0     0    10     5
     5     0     3     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
    22

     0     0     0     0     0    22


ITERACIÓN 6
Costos relativos
     0     0     0   Inf   Inf   Inf
   Inf     0     2     2     3     0
   Inf   Inf     0     1     0   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     0   Inf     0     0
     0   Inf    -2   Inf   Inf   Inf

Marcaje del algoritmo FF
     1    10     1
     2    10     1
     3     5     1
     0     0     0
     0     0     0
     0    22     1

     6     1     1     0     3     2
    15    10     5     0     0     0
     1     1     1     0     0     0


ITERACIÓN 7
Costos relativos
     0     0     0   Inf   Inf   Inf
   Inf     0     2     1     2    -1
   Inf   Inf     0     0    -1   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     1   Inf     0     0
     0   Inf    -2   Inf   Inf   Inf
```

```
Marcaje del algoritmo FF
     1    10     1
     2    10     1
     3     5     1
     4     5     1
     6     5     1
     0    22     1

     6     1     1     3     5     4
    15    10     5     5     5     5
     1     1     1     1     1     1

Flujo actual
     5     3     7     0     0     0
     0    12     0     0     0     3
     0     0     5     5     5     0
     0     0     0    10     0     5
     0     0     0     0    10     5
    10     0     3     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
    17

     0     0     0     0     0    17


ITERACIÓN 8
Costos relativos
     0     0     0   Inf   Inf   Inf
   Inf     0     2     1     2    -1
   Inf   Inf     0     0    -1   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     1   Inf     0     0
     0   Inf    -2   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     5     1
     2     5     1
     0     0     0
     0     0     0
     0     0     0
     0    17     1

     6     1     1     0     0     0
    10     5     0     0     0     0
     1     1     0     0     0     0
```

```
ITERACIÓN 9
Costos relativos
     0     0    -1   Inf   Inf   Inf
   Inf     0     1     0     1    -2
   Inf   Inf     0     0    -1   Inf
   Inf   Inf   Inf     0     1     0
   Inf   Inf     1   Inf     0     0
     0   Inf    -3   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     5     1
     2     5     1
     4     5     1
     4     5     1
     0     0     0
     0    17     1

     6     1     3     2     0     4
    10     5     5     5     0     0
     1     1     1     1     0     0


ITERACIÓN 10
Costos relativos
     0     0    -1   Inf   Inf   Inf
   Inf     0     1     0     0    -3
   Inf   Inf     0     0    -2   Inf
   Inf   Inf   Inf     0     0    -1
   Inf   Inf     2   Inf     0     0
     0   Inf    -3   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     5     1
     2     5     1
     4     5     1
     4     5     1
     5     5     1
     0    17     1

     6     1     3     2     2     5
    10     5     5     5     5     2
     1     1     1     1     1     1
```

```
Flujo actual
     3     5     7     0     0     0
     0    10     0     0     2     3
     0     0     5     5     5     0
     0     0     0    10     0     5
     0     0     0     0     8     7
    12     0     3     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
    15

     0     0     0     0     0    15


ITERACIÓN 11
Costos relativos
     0     0    -1   Inf   Inf   Inf
   Inf     0     1     0     0    -3
   Inf   Inf     0     0    -2   Inf
   Inf   Inf   Inf     0     0    -1
   Inf   Inf     2   Inf     0     0
     0   Inf    -3   Inf   Inf   Inf

Marcaje del algoritmo FF
     1     3     1
     2     3     1
     4     3     1
     4     3     1
     5     3     1
     0    15     1

     6     1     3     2     2     5
     8     3     3     3     3     0
     1     1     1     1     1     0
```

Alcanzando finalmente el óptimo:

```
flujo =

     3     5     7     0     0     0
     0    10     0     0     2     3
     0     0     5     5     5     0
     0     0     0    10     0     5
     0     0     0     0     8     7
    12     0     3     0     0     0
```

La función “postproc” da como resultado el flujo en la red original.

```
[flujoreal,Kreales,z]=postproc(flujo,C,arcos)
```

```
flujoreal =

     3     7     8     0     0     0
     0    10     4     3     4     3
     0     0     5     7     7     0
     0     0     0    10     1     5
     0     0     1     0     8     7
    12     0     3     0     0     0


Kreales =

    15     4     1    -9   -11     0


z =

    80
```

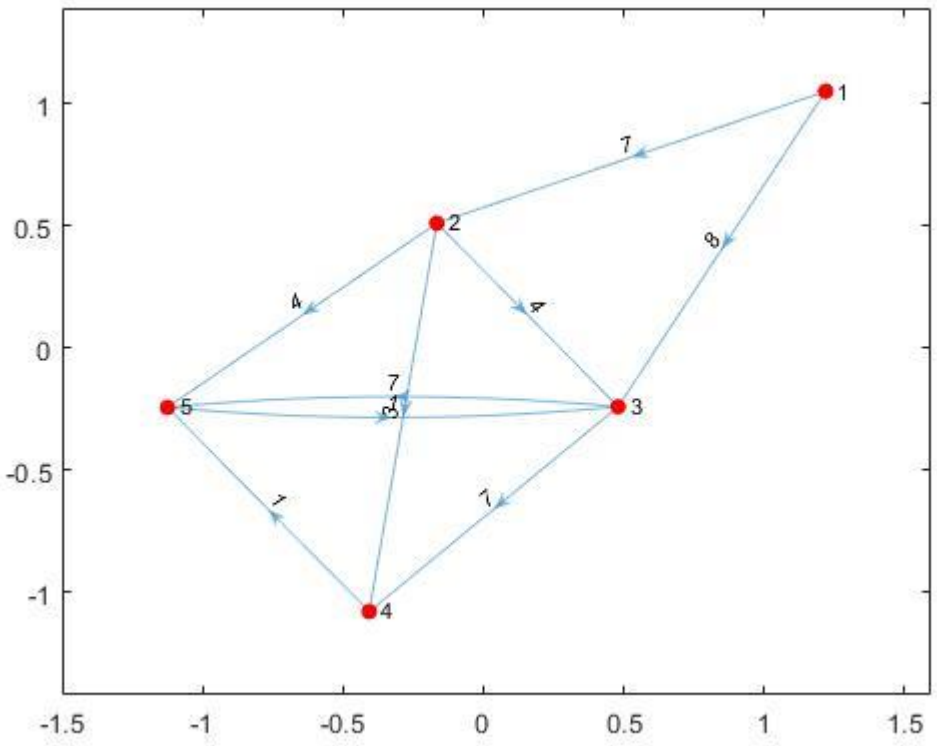


Figura 58.- Flujo óptimo en la red de la Figura 48 (método 3).

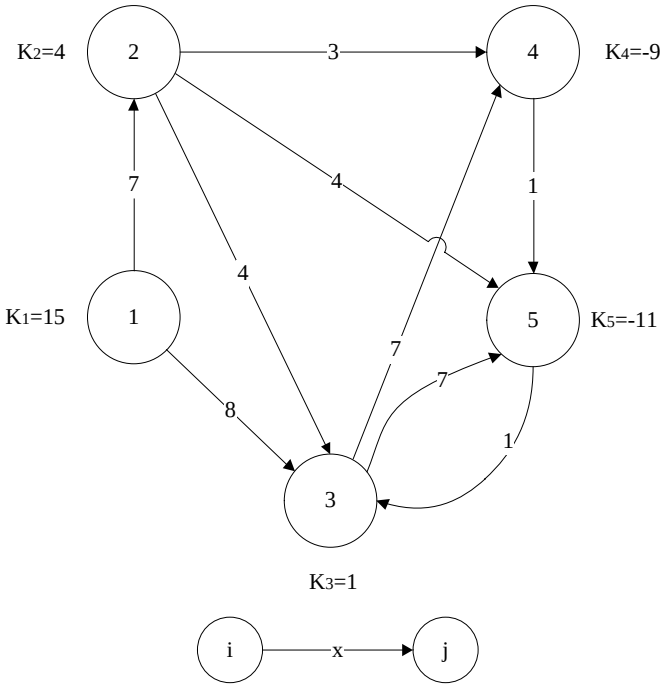


Figura 59.- Grafo realista equivalente al grafo de la Figura 58.

Para facilitar la comparación entre métodos, se indican de nuevo los óptimos alcanzados por el método 1 (Figura 53) y por el método 2 (Figura 56)

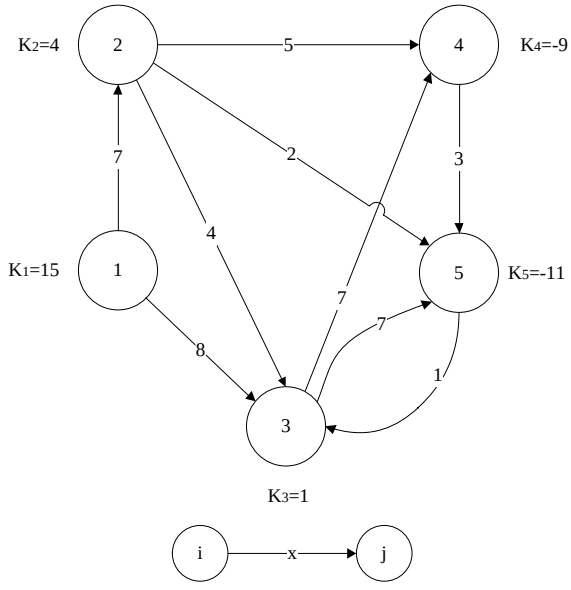


Figura 53

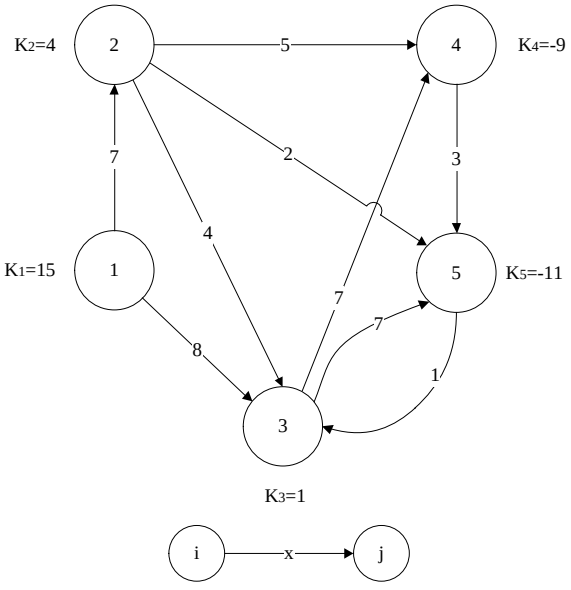


Figura 56

Los métodos 1 y 2, que tienen características similares, alcanzan el mismo óptimo para la red. Sin embargo, con el método 3, se alcanza un óptimo diferente, pero con el mismo valor de la función objetivo (80 unidades), por lo que es un óptimo alternativo. Esto es habitual en presencia de arcos con coste absoluto negativo (como es en este problema el arco (5,3)).

APÉNDICE C.- RESOLUCIÓN COMPLETA DE UN EJEMPLO (ALGORITMO PDTCD)

A la hora de resolver el problema de flujo máximo a coste mínimo con costes dependientes del flujo presentado en el Apartado 3.7, se indicó que su extensión era excesivamente compleja para mostrarla de manera total en dicho Apartado. El objetivo de este Apéndice es abordar la resolución completa de dicho ejemplo (incluyendo todos los resultados intermedios), y estandarizando la red con cada uno de los tres métodos expuestos en el Apartado 2.2.

Brevemente, se recuerda el enunciado del problema que se va a resolver:

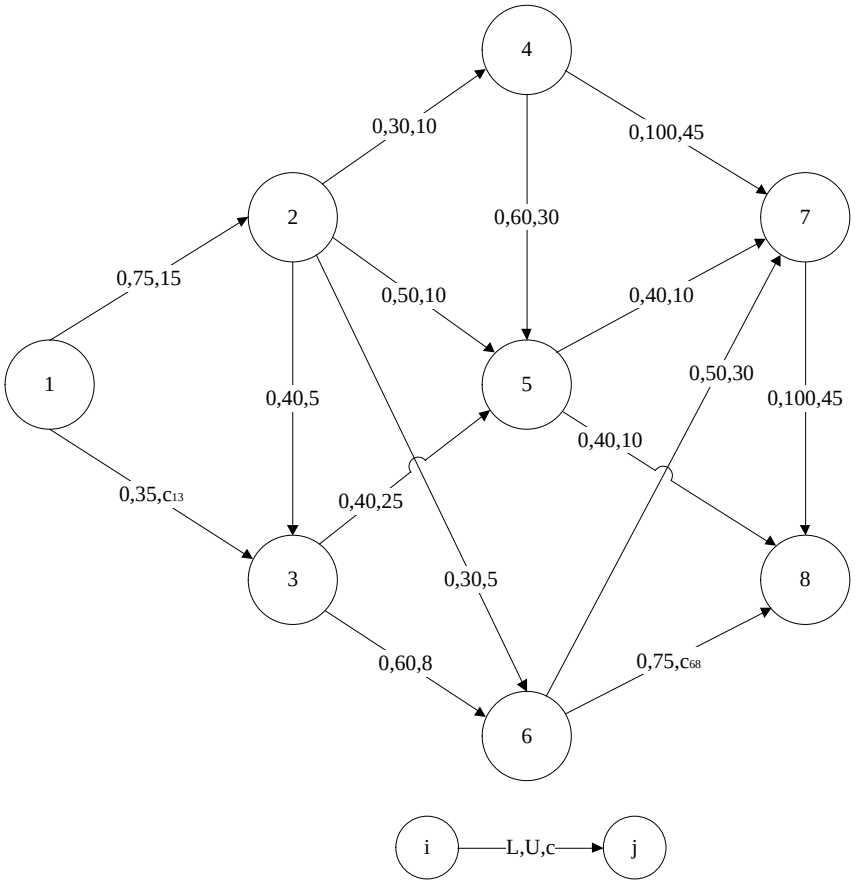


Figura 33

En los arcos (1,3) y (6,8) el coste absoluto es dependiente del flujo según las expresiones:

$$c_{13}(x_{13}) = \begin{cases} 10 & 0 \le x_{13} \le 30 \\ 10 + 0{,}1(\mathrm{x}_{13} - 30)^2 & 30 \le x_{43} \le 35 \end{cases}$$

$$c_{68}(x_{68}) = \begin{cases} 15 & 0 \le x_{68} \le 70 \\ 15 + 0{,}2(\mathrm{x}_{68} - 70) & 70 \le x_{68} \le 75 \end{cases}$$

Además, los parámetros de entrada al programa MATLAB, son:

$$arcosci = \begin{bmatrix} 1 & 2 & 0 & 75 & 15 \\ 2 & 3 & 0 & 40 & 5 \\ 2 & 4 & 0 & 30 & 10 \\ 2 & 5 & 0 & 50 & 10 \\ 2 & 6 & 0 & 30 & 5 \\ 3 & 5 & 0 & 40 & 25 \\ 3 & 6 & 0 & 60 & 8 \\ 4 & 5 & 0 & 60 & 30 \\ 4 & 7 & 0 & 100 & 45 \\ 5 & 7 & 0 & 40 & 10 \\ 5 & 8 & 0 & 40 & 10 \\ 6 & 7 & 0 & 50 & 30 \\ 7 & 8 & 0 & 100 & 45 \end{bmatrix}$$

$$nodoscd = \begin{bmatrix} 1 & 3 \\ 6 & 8 \end{bmatrix}$$

$$K = [110 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad -215]$$

$$C13.breaks = [0 \quad 30 \quad 35] \qquad C68.breaks = [0 \quad 70 \quad 75]$$

$$C13.coefs = \begin{bmatrix} 0 & 0 & 10 \\ 0{,}1 & 0 & 10 \end{bmatrix} \qquad C68.coefs = \begin{bmatrix} 0 & 15 \\ 0{,}2 & 15 \end{bmatrix}$$

que se introducirán en la ventana de comandos de MATLAB mediante las siguientes líneas de código:

```
arcosci=[1 2 0 75 15;2 3 0 40 5;2 4 0 30 10;2 5 0 50 10;2 6 0 30 5;3 5 0 40
25;3 6 0 60 8;4 5 0 60 30;4 7 0 100 45;5 7 0 40 10;5 8 0 40 10;6 7 0 50 30;7
8 0 100 45];
nodoscd=[1 3;6 8];
K=[110 0 0 0 0 0 0 -215];
C13=mkpp([0 30 35],[0 0 10;0.1 0 10]);
C68=mkpp([0 70 75],[0 15; 0.2 15]);
CDA=[C13,C68];
```

C.1. Método 1.

El equilibrado de la red se realiza incluyendo un nodo ficticio, 9, que aporte la oferta correspondiente al exceso de demanda que tiene la red, es decir, 105 unidades de flujo. De acuerdo al método 1, dicho nodo 9 estará unido al único nodo de demanda, que es el nodo 8. Es decir:

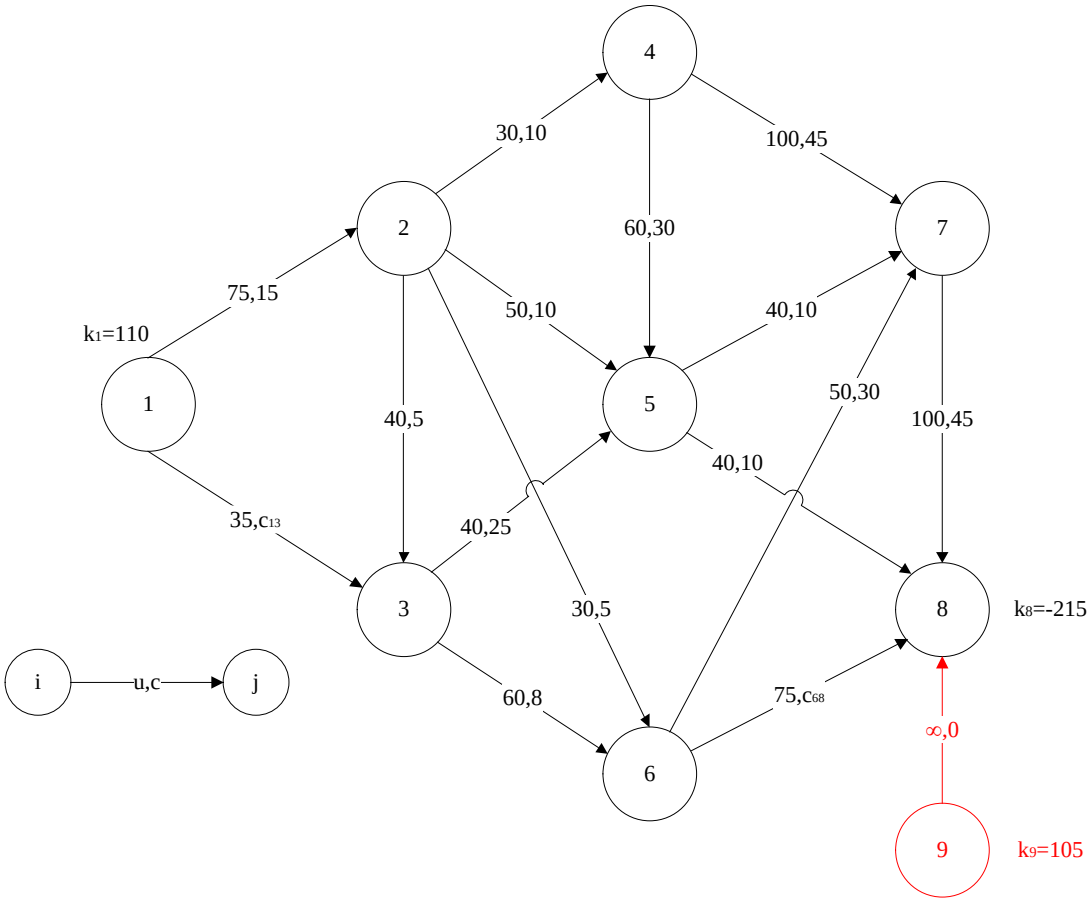


Figura 60.- Grafo de la Figura 33, estandarizado mediante el método 1.

Ejecutando la función “estandarizarcd”, se obtienen los siguientes resultados:

```
[U,C,of,dem,arcos]=estandarizarcd(arcosci,nodoscd,CDA,K,1);
```

```
U =

   Inf    75    30   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf    40    30    50    30   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    40    60   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    60   Inf   100   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    40    40   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    50    70   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   100   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

```
C =

     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0

of =

   325   215   215   215   215   215   215   215   320


dem =

   215   215   215   215   215   215   215   430   215
```

Así como una representación de la red, que ya se indicó en el Apartado 3.7., y que aquí se recupera a modo de recordatorio.

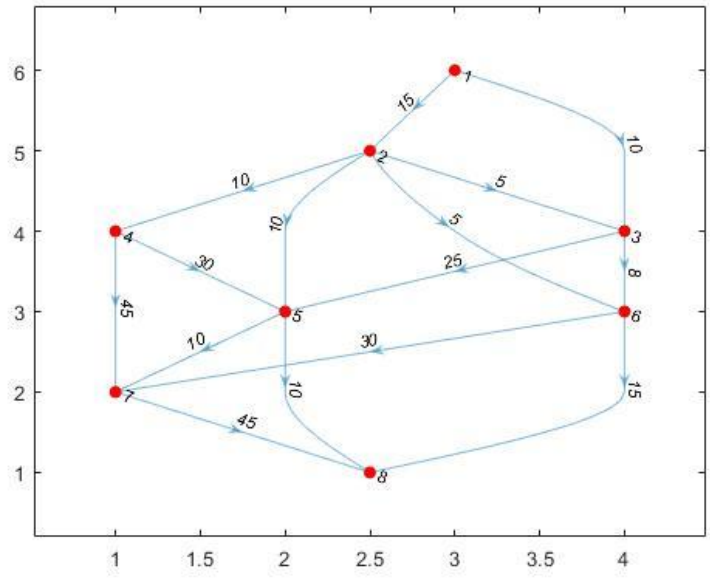


Figura 35.- Grafo de la Figura 33, representado en MATLAB.

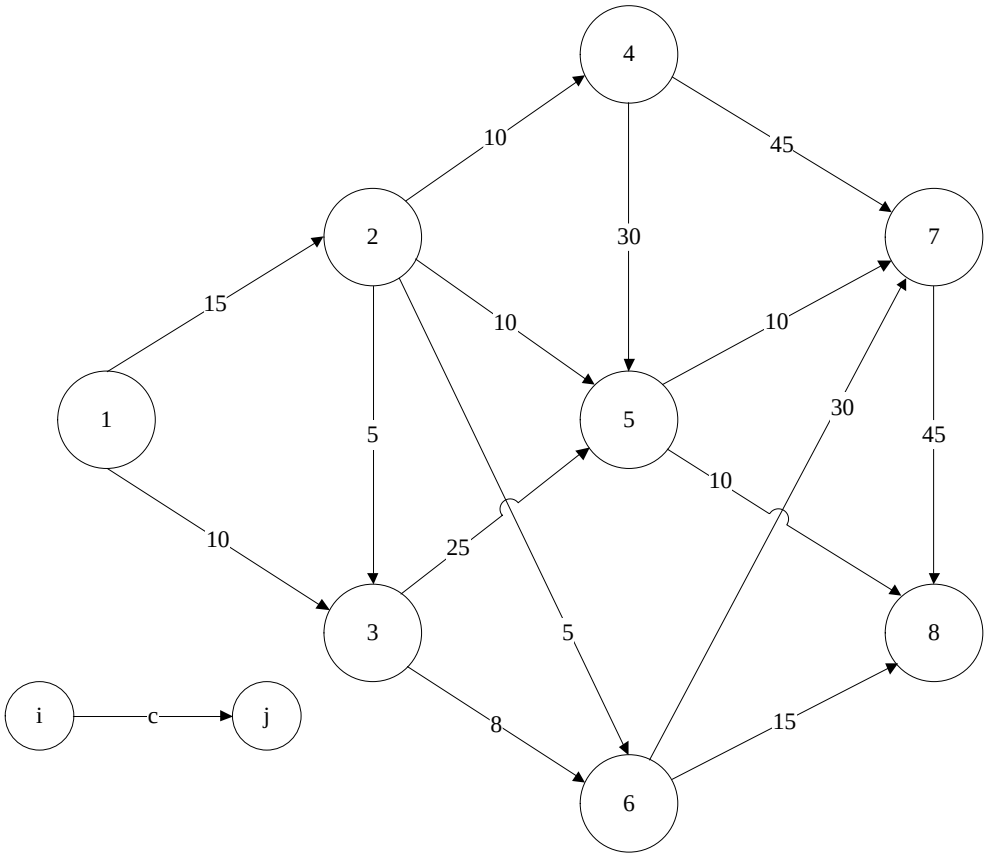


Figura 36.- Grafo realista equivalente al grafo de la Figura 35.

Una vez más, se recuerda que esta representación del grafo en MATLAB indica los costes de cada arco y, por tanto, es independiente del método de estandarización. Por tanto, se omitirá en la resolución del problema mediante los otros dos métodos.

A continuación, se indican (sin comentar) los resultados intermedios obtenidos al ejecutar la función "pdtcd" hasta alcanzar el óptimo de la red. Adicionalmente, se indica en una tabla análoga a la Tabla 20, el número de veces que se ha necesitado ejecutar cada subrutina para alcanzar el óptimo.

Tabla 24.- Subrutinas (PDTCD) ejecutadas en la resolución de la red de la Figura 33 (método 1).

	matred	**asignaflujo**	**marcajeff + compruebasaturacd**	**aumentaflujo**	**modificamr**	**nuevaCU**
Iteración 1	1	6	7	1	5	1
Iteración 2	1	7	10	3	6	1
Iteración 3	1	8	10	2	7	1
Iteración 4	1	4	5	1	3	1
Iteración 5	1	4	5	1	3	1
Iteración 6	1	4	4	1	3	0

```
CDM=calculacostemarginal(CDA);
[flujo]=pdtcd(C,of,dem,U,nodoscd,CDM)
```

```
Costes marginales
     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
Capacidades modificadas
   Inf    75    30   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf    40    30    50    30   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    40    60   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    60   Inf   100   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    40    40   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    50    70   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   100   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf

ITERACIÓN 1
Costos relativos
     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
```

```
Flujo actual
   215     0     0     0     0     0     0     0     0
     0   215     0     0     0     0     0     0     0
     0     0   215     0     0     0     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   215     0     0     0     0
     0     0     0     0     0   215     0     0     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   215   105

Oferta y demanda
   110
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0   110

 Marcaje del algoritmo FF
     0   110     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     0     0     0     0     0     0     0
   110     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0

Nueva Mat Red
     0     5     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
```

```
Marcaje del algoritmo FF
     0   110     1
     0     0     0
     3    30     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
   110     0    30     0     0     0     0     0     0
     1     0     1     0     0     0     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    20     3   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0


Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     1     1     0     0     0     0     0     0
   110    75    30     0     0     0     0     0     0
     1     1     1     0     0     0     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     7     7     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    17     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
```

```
Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     0     0     0
     0     0     0
     6    30     1
     0     0     0
     0     0     0
     0     0     0

     1     1     1     0     0     3     0     0     0
   110    75    30     0     0    30     0     0     0
     1     1     1     0     0     1     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     0     0     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    10     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    23     8   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0


Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     4    30     1
     5    50     1
     6    30     1
     0     0     0
     0     0     0
     0     0     0

     1     1     1     2     2     3     0     0     0
   110    75    30    30    50    30     0     0     0
     1     1     1     1     1     1     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     0     0     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    10     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    37   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf     2     2   Inf
   Inf   Inf   Inf   Inf   Inf     0    15     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
```

```
Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     4    30     1
     5    50     1
     6    30     1
     0     0     0
     8    30     1
     8    30     1

     1     1     1     2     2     3     0     6     9
   110    75    30    30    50    30     0    30    30
     1     1     1     1     1     1     0     1     1

Flujo actual
   215     0    30     0     0     0     0     0     0
     0   215     0     0     0     0     0     0     0
     0     0   185     0     0    30     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   215     0     0     0     0
     0     0     0     0     0   185     0    30     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   185   135

Oferta y demanda
    80
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0    80
```

Marcaje del algoritmo FF

0	80	1
2	75	1
0	0	0
4	30	1
5	50	1
0	0	0
0	0	0
0	0	0
0	0	0

1	1	1	2	2	0	0	0	0
80	75	0	30	50	0	0	0	0
1	1	0	1	1	0	0	0	0

Nuevos costes marginales

0	15	16.3	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Nuevas capacidades

Inf	75	31	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 2

Costos relativos

0	15	16.3	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	80	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	0	1.3	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	80	1
2	75	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	1	0	0	0	0	0	0	0
80	75	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	8.7	8.7	3.7	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     80      1
     2     75      1
     3      1      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0

     1      1      1      0      0      0      0      0      0
    80     75      1      0      0      0      0      0      0
     1      1      1      0      0      0      0      0      0
```

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	5	5	0	Inf	Inf	Inf
Inf	Inf	0	Inf	21.3	4.3	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     80      1
     2     75      1
     3      1      1
     0      0      0
     0      0      0
     6     30      1
     0      0      0
     0      0      0
     0      0      0

     1      1      1      0      0      2      0      0      0
    80     75      1      0      0     30      0      0      0
     1      1      1      0      0      1      0      0      0
```

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	0	0	0	Inf	Inf	Inf
Inf	Inf	0	Inf	16.3	4.3	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	25	10	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0    80     1
     2    75     1
     3     1     1
     4    30     1
     5    50     1
     6    30     1
     0     0     0
     0     0     0
     0     0     0

     1     1     1     2     2     2     0     0     0
    80    75     1    30    50    30     0     0     0
     1     1     1     1     1     1     0     0     0
```

Nueva Mat Red

0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	0	0	Inf	Inf	Inf
Inf	Inf	0	Inf	16.3	4.3	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	0	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0    80     1
     2    75     1
     3     1     1
     4    30     1
     5    50     1
     6    30     1
     7    40     1
     8    30     1
     8    30     1

     1     1     1     2     2     2     5     6     9
    80    75     1    30    50    30    40    30    30
     1     1     1     1     1     1     1     1     1
```

```
Flujo actual
   215    30    30     0     0     0     0     0     0
     0   185     0     0     0    30     0     0     0
     0     0   185     0     0    30     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   215     0     0     0     0
     0     0     0     0     0   155     0    60     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   155   165

Oferta y demanda
    50
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0    50


Marcaje del algoritmo FF
     0    50     1
     2    45     1
     3     1     1
     4    30     1
     5    45     1
     8    40     1
     7    40     1
     8    40     1
     8    40     1

     1     1     1     2     2     6     5     5     9
    50    45     1    30    45    40    40    40    40
     1     1     1     1     1     1     1     1     1
```

Flujo actual

215	70	30	0	0	0	0	0	0
0	145	0	0	40	30	0	0	0
0	0	185	0	0	30	0	0	0
0	0	0	215	0	0	0	0	0
0	0	0	0	175	0	0	40	0
0	0	0	0	0	155	0	60	0
0	0	0	0	0	0	215	0	0
0	0	0	0	0	0	0	215	0
0	0	0	0	0	0	0	115	205

Oferta y demanda

10								
0								
0								
0								
0								
0								
0								
0								
0								
0	0	0	0	0	0	0	0	10

Marcaje del algoritmo FF

0	10	1
2	5	1
3	1	1
4	5	1
5	5	1
0	0	0
7	5	1
0	0	0
0	0	0

1	1	1	2	2	2	5	5	0
10	5	1	5	5	0	5	0	0
1	1	1	1	1	0	1	0	0

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	0	0	-4.3	Inf	Inf	Inf
Inf	Inf	0	Inf	16.3	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	-4.3	Inf
Inf	Inf	Inf	Inf	Inf	0	19.3	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	40.7	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0    10     1
     2     5     1
     3     1     1
     4     5     1
     5     5     1
     6     1     1
     7     5     1
     8     1     1
     8     1     1

     1     1     1     2     2     3     5     6     9
    10     5     1     5     5     1     5     1     1
     1     1     1     1     1     1     1     1     1
Flujo actual
   215    70    31     0     0     0     0     0     0
     0   145     0     0    40    30     0     0     0
     0     0   184     0     0    31     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   154     0    61     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   114   206

Oferta y demanda
     9
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0     9
Marcaje del algoritmo FF
     0     9     1
     2     5     1
     0     0     0
     4     5     1
     5     5     1
     0     0     0
     7     5     1
     0     0     0
     0     0     0

     1     1     1     2     2     0     5     0     0
     9     5     0     5     5     0     5     0     0
     1     1     0     1     1     0     1     0     0
```

Nuevos costes marginales

0	15	23.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Nuevas capacidades

Inf	75	32	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 3

Costos relativos

0	15	23.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	9	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	0	8.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	9	1
2	5	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	1	0	0	0	0	0	0	0
9	5	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0

Nueva Mat Red

0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
0	0	5	5	0	Inf	Inf	Inf
Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	9	1
2	5	1
3	5	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	1	2	0	0	2	0	0	0
9	5	5	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0

Nueva Mat Red

0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
0	0	0	0	-5	Inf	Inf	Inf
Inf	0	Inf	20	3	Inf	Inf	Inf
Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	9	1
2	5	1
3	5	1
4	5	1
5	5	1
0	0	0
0	0	0
0	0	0
0	0	0

1	1	2	2	2	0	0	0	0
9	5	5	5	5	0	0	0	0
1	1	1	1	1	0	0	0	0

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-8	Inf	Inf	Inf
Inf	Inf	0	Inf	20	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	42	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	7	7	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	9	1
2	5	1
3	5	1
4	5	1
5	5	1
6	5	1
0	0	0
0	0	0
0	0	0

1	1	2	2	2	3	0	0	0
9	5	5	5	5	5	0	0	0
1	1	1	1	1	1	0	0	0

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-8	Inf	Inf	Inf
Inf	Inf	0	Inf	20	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	0	Inf
Inf	Inf	Inf	Inf	Inf	0	23	8	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     9     1
     2     5     1
     3     5     1
     4     5     1
     5     5     1
     6     5     1
     7     5     1
     0     0     0
     0     0     0

     1     1     2     2     2     3     5     5     0
     9     5     5     5     5     5     5     0     0
     1     1     1     1     1     1     1     0     0
```

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-8	Inf	Inf	Inf
Inf	Inf	0	Inf	20	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	-8	Inf
Inf	Inf	Inf	Inf	Inf	0	23	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	37	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     9     1
     2     5     1
     3     5     1
     4     5     1
     5     5     1
     6     5     1
     7     5     1
     8     5     1
     8     5     1

     1     1     2     2     2     3     5     6     9
     9     5     5     5     5     5     5     5     5
     1     1     1     1     1     1     1     1     1
```

Flujo actual

215	75	31	0	0	0	0	0	0
0	140	5	0	40	30	0	0	0
0	0	179	0	0	36	0	0	0
0	0	0	215	0	0	0	0	0
0	0	0	0	175	0	0	40	0
0	0	0	0	0	149	0	66	0
0	0	0	0	0	0	215	0	0
0	0	0	0	0	0	0	215	0
0	0	0	0	0	0	0	109	211

Oferta y demanda

4								
0								
0								
0								
0								
0								
0								
0								
0								
0	0	0	0	0	0	0	0	4

Marcaje del algoritmo FF

0	4	1						
0	0	0						
0	0	0						
0	0	0						
0	0	0						
0	0	0						
0	0	0						
0	0	0						
0	0	0						
1	1	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-3.2	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-8	Inf	Inf	Inf
Inf	Inf	0	Inf	20	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	-8	Inf
Inf	Inf	Inf	Inf	Inf	0	23	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	37	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0      4      1
     3      1      1
     3      1      1
     4      1      1
     5      1      1
     6      1      1
     7      1      1
     8      1      1
     8      1      1

     1      2      1      2      2      3      5      6      9
     4      1      1      1      1      1      1      1      1
     1      1      1      1      1      1      1      1      1

Flujo actual
   215     75     32      0      0      0      0      0      0
     0    140      5      0     40     30      0      0      0
     0      0    178      0      0     37      0      0      0
     0      0      0    215      0      0      0      0      0
     0      0      0      0    175      0      0     40      0
     0      0      0      0      0    148      0     67      0
     0      0      0      0      0      0    215      0      0
     0      0      0      0      0      0      0    215      0
     0      0      0      0      0      0      0    108    212

Oferta y demanda
     3
     0
     0
     0
     0
     0
     0
     0
     0

     0      0      0      0      0      0      0      0      3

Marcaje del algoritmo FF
     0      3      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0

     1      0      1      0      0      0      0      0      0
     3      0      0      0      0      0      0      0      0
     1      0      0      0      0      0      0      0      0
```

Nuevos costes marginales

0	15	30.7	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Nuevas capacidades

Inf	75	33	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 4

Costos relativos

0	15	30.7	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	3	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-15.7	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     3     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
     3     0     1     0     0     0     0     0     0
     1     0     1     0     0     0     0     0     0

Nueva Mat Red
     0   -23.7     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    13    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    17     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
```

```
Marcaje del algoritmo FF
     0     3     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     6     1     1
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     3     0     0     0
     3     0     1     0     0     1     0     0     0
     1     0     1     0     0     1     0     0     0

Nueva Mat Red
     0   -38.7     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    28    10    10    20   Inf   Inf   Inf
   Inf   Inf     0   Inf     2     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    15     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
```

```
Marcaje del algoritmo FF
     0     3     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     6     1     1
     0     0     0
     8     1     1
     8     1     1

     1     0     1     0     0     3     0     6     9
     3     0     1     0     0     1     0     1     1
     1     0     1     0     0     1     0     1     1

Flujo actual
   215    75    33     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   177     0     0    38     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   147     0    68     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   107   213

Oferta y demanda
     2
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0     2
Marcaje del algoritmo FF
     0     2     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
     2     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

Nuevos costes marginales

0	15	38.8	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Nuevas capacidades

Inf	75	34	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 5

Costos relativos

0	15	38.8	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	2	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-23.8	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	2	1
0	0	0
3	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	1	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0

Nueva Mat Red

0	-31.8	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	13	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	17	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	2	1
0	0	0
3	1	1
0	0	0
0	0	0
6	1	1
0	0	0
0	0	0
0	0	0

1	0	1	0	0	3	0	0	0
2	0	1	0	0	1	0	0	0
1	0	1	0	0	1	0	0	0

Nueva Mat Red

0	-46.8	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	28	10	10	20	Inf	Inf	Inf
Inf	Inf	0	Inf	2	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     2     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     6     1     1
     0     0     0
     8     1     1
     8     1     1

     1     0     1     0     0     3     0     6     9
     2     0     1     0     0     1     0     1     1
     1     0     1     0     0     1     0     1     1

Flujo actual
   215    75    34     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   176     0     0    39     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   146     0    69     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   106   214

Oferta y demanda
     1
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0     1

Marcaje del algoritmo FF
     0     1     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

Nuevos costes marginales

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Nuevas capacidades

Inf	75	35	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 6

Costos relativos

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

Marcaje del algoritmo FF

0	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-32.5	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     1     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
     1     0     1     0     0     0     0     0     0
     1     0     1     0     0     0     0     0     0
```

Nueva Mat Red

0	-40.5	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	13	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	17	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     1     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     6     1     1
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     3     0     0     0
     1     0     1     0     0     1     0     0     0
     1     0     1     0     0     1     0     0     0
```

Nueva Mat Red

0	-55.5	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	28	10	10	20	Inf	Inf	Inf
Inf	Inf	0	Inf	2	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0

```
Marcaje del algoritmo FF
     0     1     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     6     1     1
     0     0     0
     8     1     1
     8     1     1

     1     0     1     0     0     3     0     6     9
     1     0     1     0     0     1     0     1     1
     1     0     1     0     0     1     0     1     1
```

Y finalmente, se alcanza el óptimo:

```
flujo =

   215    75    35     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   175     0     0    40     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   145     0    70     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   105   215
```

Deshaciendo los cambios de variable para obtener el óptimo sobre la red original:

```
[flujoreal,Kreales,z]=postproccd(flujo,arcos,C,nodoscd,CDA)
```

Se obtiene:

```
flujoreal =

   215    75    35     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   175     0     0    40     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   145     0    70     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   105   215
```

```
Kreales =

   110     0     0     0     0     0     0  -110     0

z =

    3907.5
```

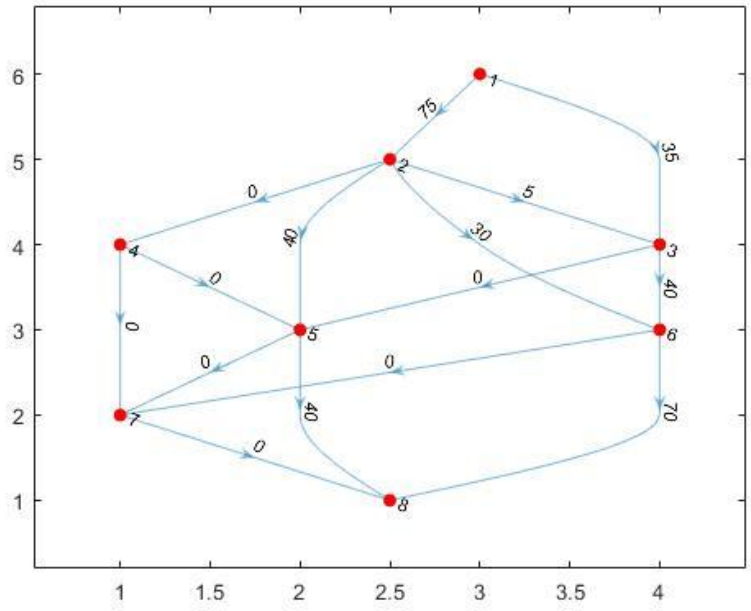


Figura 61.- Flujo óptimo en la red de la Figura 33 (método 1).

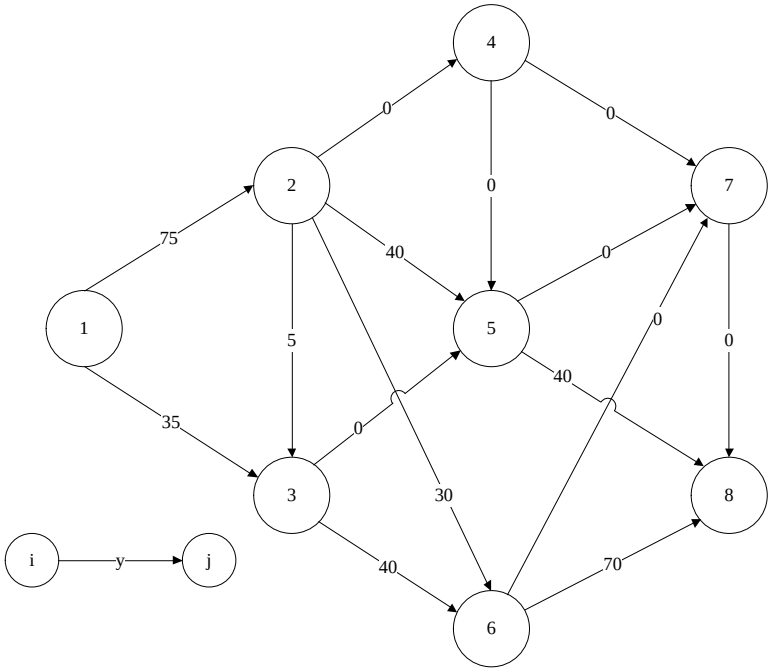


Figura 62.- Grafo realista equivalente al grafo de la Figura 61.

C.2. Método 2.

De acuerdo al método 2, el nodo ficticio 9, se une mediante arcos ficticios a todo el resto de nodos de la red. Es decir:

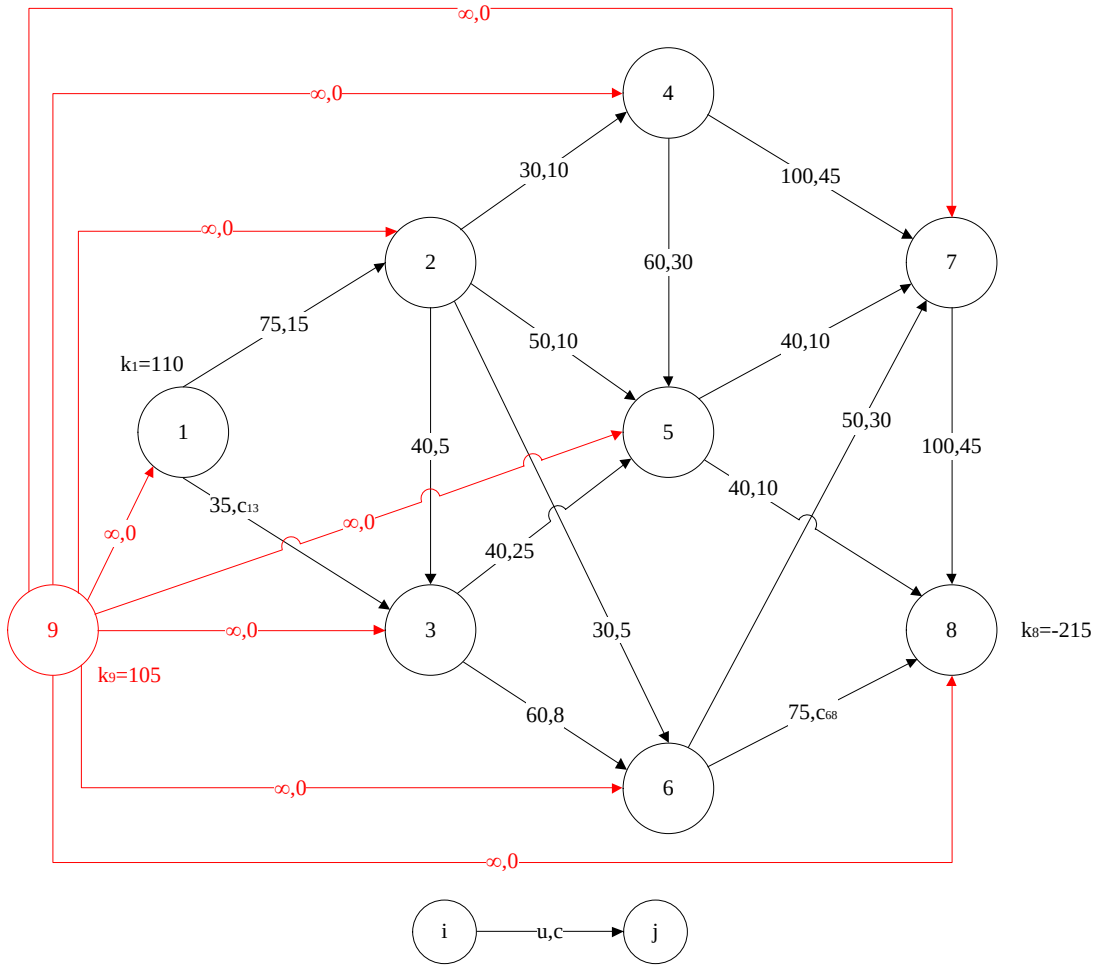


Figura 63.- Grafo de la Figura 33, estandarizado mediante el método 2.

Los resultados de la estandarización en MATLAB son las matrices “U” y “C” y los vectores “of” y “dem”.

```
[U,C,of,dem,arcos]=estandarizarcd(arcosci,nodoscd,CDA,K,2);
```

```
U =

   Inf    75    30   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf    40    30    50    30   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    40    60   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    60   Inf   100   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    40    40   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    50    70   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   100   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

```
C =

     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
     0     0     0     0     0     0     0     0     0

of =

   325   215   215   215   215   215   215   215   320


dem =

   215   215   215   215   215   215   215   430   215
```

Al ejecutar la función "pdtcd" se obtienen los siguientes resultados intermedios:

Tabla 25.- Subrutinas (PDTCD) ejecutadas en la resolución de la red de la Figura 33 (método 2).

	matred	**asignaflujo**	**marcajeff + compruebasaturacd**	**aumentaflujo**	**modificamr**	**nuevaCU**
Iteración 1	1	6	7	1	5	1
Iteración 2	1	7	10	3	6	1
Iteración 3	1	8	10	2	7	1
Iteración 4	1	4	5	1	3	1
Iteración 5	1	4	5	1	3	1
Iteración 6	1	4	4	1	3	0

Se observa que los cálculos intermedios son análogos a los del método 1, con la salvedad de la presencia de costes no infinitos en la última fila de la matriz de costes relativos (asociados a los arcos ficticios que unen el nodo 9 con todos los nodos de la red, y que no aparecían en la estandarización de la red según el método 1)

```
CDM=calculacostemarginal(CDA);
[flujo]=pdtcd(C,of,dem,U,nodoscd,CDM)
```

```
Costes marginales
     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
     0     0     0     0     0     0     0     0     0

Capacidades modificadas
   Inf    75    30   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf    40    30    50    30   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    40    60   Inf   Inf   Inf
   Inf   Inf   Inf   Inf    60   Inf   100   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    40    40   Inf
   Inf   Inf   Inf   Inf   Inf   Inf    50    70   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   100   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf


ITERACIÓN 1
Costos relativos
     0    15    10   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
     0     0     0     0     0     0     0     0     0
```

```
Flujo actual
   215     0     0     0     0     0     0     0     0
     0   215     0     0     0     0     0     0     0
     0     0   215     0     0     0     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   215     0     0     0     0
     0     0     0     0     0   215     0     0     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   215   105

Oferta y demanda
   110
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0   110


Marcaje del algoritmo FF
     0   110     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     0     0     0     0     0     0     0
   110     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0

Nueva Mat Red
     0     5     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     5    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    25     8   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
    10     0     0     0     0     0     0     0     0
```

```
Marcaje del algoritmo FF
     0   110     1
     0     0     0
     3    30     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
   110     0    30     0     0     0     0     0     0
     1     0     1     0     0     0     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10    10    10     5   Inf   Inf   Inf
   Inf   Inf     0   Inf    20     3   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
    15     0     5     0     0     0     0     0     0


Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     1     1     0     0     0     0     0     0
   110    75    30     0     0     0     0     0     0
     1     1     1     0     0     0     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     7     7     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    17     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
    18     3     8     0     0     0     0     0     0
```

```
Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     0     0     0
     0     0     0
     6    30     1
     0     0     0
     0     0     0
     0     0     0

     1     1     1     0     0     3     0     0     0
   110    75    30     0     0    30     0     0     0
     1     1     1     0     0     1     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     0     0     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    10     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    23     8   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
    25    10    15     0     0     7     0     0     0


Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     4    30     1
     5    50     1
     6    30     1
     0     0     0
     0     0     0
     0     0     0

     1     1     1     2     2     3     0     0     0
   110    75    30    30    50    30     0     0     0
     1     1     1     1     1     1     0     0     0

Nueva Mat Red
     0     0     0   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0    10     0     0     2   Inf   Inf   Inf
   Inf   Inf     0   Inf    10     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    37   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf     2     2   Inf
   Inf   Inf   Inf   Inf   Inf     0    15     0   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0   Inf
    33    18    23     8     8    15     0     0     0
```

```
Marcaje del algoritmo FF
     0   110     1
     2    75     1
     3    30     1
     4    30     1
     5    50     1
     6    30     1
     7    30     1
     8    30     1
     8    30     1

     1     1     1     2     2     3     9     6     9
   110    75    30    30    50    30    30    30    30
     1     1     1     1     1     1     1     1     1

Flujo actual
   215     0    30     0     0     0     0     0     0
     0   215     0     0     0     0     0     0     0
     0     0   185     0     0    30     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   215     0     0     0     0
     0     0     0     0     0   185     0    30     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   185   135

Oferta y demanda
    80
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0    80
```

Marcaje del algoritmo FF

0	80	1
2	75	1
0	0	0
4	30	1
5	50	1
0	0	0
0	0	0
0	0	0
0	0	0

1	1	1	2	2	0	0	0	0
80	75	0	30	50	0	0	0	0
1	1	0	1	1	0	0	0	0

Nuevos costes marginales

0	15	16.3	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Nuevas capacidades

Inf	75	31	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 2

Costos relativos

0	15	16.3	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	80	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	0	1.3	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
15	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	80	1
2	75	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	1	0	0	0	0	0	0	0
80	75	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	8.7	8.7	3.7	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
16.3	1.3	0	0	0	0	0	0	0

```
Marcaje del algoritmo FF
     0     80      1
     2     75      1
     3      1      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0

     1      1      1      0      0      0      0      0      0
    80     75      1      0      0      0      0      0      0
     1      1      1      0      0      0      0      0      0

Nueva Mat Red
     0      0      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0    3.7      5      5      0    Inf    Inf    Inf
   Inf    Inf      0    Inf   21.3    4.3    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     30     15    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0    Inf
    20      5    3.7      0      0      0      0      0      0

Marcaje del algoritmo FF
     0     80      1
     2     75      1
     3      1      1
     0      0      0
     0      0      0
     6     30      1
     0      0      0
     0      0      0
     0      0      0

     1      1      1      0      0      2      0      0      0
    80     75      1      0      0     30      0      0      0
     1      1      1      0      0      1      0      0      0

Nueva Mat Red
     0      0      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0    3.7      0      0      0    Inf    Inf    Inf
   Inf    Inf      0    Inf   16.3    4.3    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     25     10    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0    Inf
    25     10    8.7      0      0      5      0      0      0
```

```
Marcaje del algoritmo FF
     0    80     1
     2    75     1
     3     1     1
     4    30     1
     5    50     1
     6    30     1
     0     0     0
     0     0     0
     0     0     0

     1     1     1     2     2     2     0     0     0
    80    75     1    30    50    30     0     0     0
     1     1     1     1     1     1     0     0     0
```

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	0	0	0	Inf	Inf	Inf
Inf	Inf	0	Inf	16.3	4.3	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	0	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
35	20	18.7	10	10	15	0	0	0

```
Marcaje del algoritmo FF
     0    80     1
     2    75     1
     3     1     1
     4    30     1
     5    50     1
     6    30     1
     7    40     1
     8    30     1
     8    30     1

     1     1     1     2     2     2     5     6     9
    80    75     1    30    50    30    40    30    30
     1     1     1     1     1     1     1     1     1
```

```
Flujo actual
   215    30    30     0     0     0     0     0     0
     0   185     0     0     0    30     0     0     0
     0     0   185     0     0    30     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   215     0     0     0     0
     0     0     0     0     0   155     0    60     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   155   165

Oferta y demanda
    50
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0    50


Marcaje del algoritmo FF
     0    50     1
     2    45     1
     3     1     1
     4    30     1
     5    45     1
     8    40     1
     7    40     1
     8    40     1
     8    40     1

     1     1     1     2     2     6     5     5     9
    50    45     1    30    45    40    40    40    40
     1     1     1     1     1     1     1     1     1
```

Flujo actual

215	70	30	0	0	0	0	0	0
0	145	0	0	40	30	0	0	0
0	0	185	0	0	30	0	0	0
0	0	0	215	0	0	0	0	0
0	0	0	0	175	0	0	40	0
0	0	0	0	0	155	0	60	0
0	0	0	0	0	0	215	0	0
0	0	0	0	0	0	0	215	0
0	0	0	0	0	0	0	115	205

Oferta y demanda

10								
0								
0								
0								
0								
0								
0								
0								
0								
0	0	0	0	0	0	0	0	10

Marcaje del algoritmo FF

0	10	1
2	5	1
3	1	1
4	5	1
5	5	1
0	0	0
7	5	1
0	0	0
0	0	0

1	1	1	2	2	2	5	5	0
10	5	1	5	5	0	5	0	0
1	1	1	1	1	0	1	0	0

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	0	0	-4.3	Inf	Inf	Inf
Inf	Inf	0	Inf	16.3	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	-4.3	Inf
Inf	Inf	Inf	Inf	Inf	0	19.3	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	40.7	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
39.3	24.3	23	14.3	14.3	15	4.3	0	0

```
Marcaje del algoritmo FF
     0    10     1
     2     5     1
     3     1     1
     4     5     1
     5     5     1
     6     1     1
     7     5     1
     8     1     1
     8     1     1

     1     1     1     2     2     3     5     6     9
    10     5     1     5     5     1     5     1     1
     1     1     1     1     1     1     1     1     1

Flujo actual
   215    70    31     0     0     0     0     0     0
     0   145     0     0    40    30     0     0     0
     0     0   184     0     0    31     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   154     0    61     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   114   206

Oferta y demanda
     9
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0     9
```

Marcaje del algoritmo FF

0	9	1
2	5	1
0	0	0
4	5	1
5	5	1
0	0	0
7	5	1
0	0	0
0	0	0

1	1	1	2	2	0	5	0	0
9	5	0	5	5	0	5	0	0
1	1	0	1	1	0	1	0	0

Nuevos costes marginales

0	15	23.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Nuevas capacidades

Inf	75	32	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 3

Costos relativos

0	15	23.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	9	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	0	8.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
15	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	9	1
2	5	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	1	0	0	0	0	0	0	0
9	5	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	5	5	0	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
20	5	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	9	1
2	5	1
3	5	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	1	2	0	0	2	0	0	0
9	5	5	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-5	Inf	Inf	Inf
Inf	Inf	0	Inf	20	3	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
25	10	5	0	0	0	0	0	0

Marcaje del algoritmo FF

0	9	1
2	5	1
3	5	1
4	5	1
5	5	1
0	0	0
0	0	0
0	0	0
0	0	0

1	1	2	2	2	0	0	0	0
9	5	5	5	5	0	0	0	0
1	1	1	1	1	0	0	0	0

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-8	Inf	Inf	Inf
Inf	Inf	0	Inf	20	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	42	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	7	7	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
28	13	8	3	3	0	0	0	0

```
Marcaje del algoritmo FF
     0     9     1
     2     5     1
     3     5     1
     4     5     1
     5     5     1
     6     5     1
     0     0     0
     0     0     0
     0     0     0

     1     1     2     2     2     3     0     0     0
     9     5     5     5     5     5     0     0     0
     1     1     1     1     1     1     0     0     0
```

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-8	Inf	Inf	Inf
Inf	Inf	0	Inf	20	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	0	Inf
Inf	Inf	Inf	Inf	Inf	0	23	8	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
35	20	15	10	10	7	0	0	0

```
Marcaje del algoritmo FF
     0     9     1
     2     5     1
     3     5     1
     4     5     1
     5     5     1
     6     5     1
     7     5     1
     0     0     0
     0     0     0

     1     1     2     2     2     3     5     5     0
     9     5     5     5     5     5     5     0     0
     1     1     1     1     1     1     1     0     0
```

Nueva Mat Red

0	0	3.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	0	0	0	-8	Inf	Inf	Inf
Inf	Inf	0	Inf	20	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	-8	Inf
Inf	Inf	Inf	Inf	Inf	0	23	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	37	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
43	28	23	18	18	15	8	0	0

```
Marcaje del algoritmo FF
     0     9     1
     2     5     1
     3     5     1
     4     5     1
     5     5     1
     6     5     1
     7     5     1
     8     5     1
     8     5     1

     1     1     2     2     2     3     5     6     9
     9     5     5     5     5     5     5     5     5
     1     1     1     1     1     1     1     1     1

Flujo actual
   215    75    31     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   179     0     0    36     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   149     0    66     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   109   211

Oferta y demanda
     4
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0     4
```

```
Marcaje del algoritmo FF
     0     4     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     1     0     0     0     0     0     0     0
     4     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

```
Nueva Mat Red
     0    -3.2     0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0     0      0      0     -8    Inf    Inf    Inf
   Inf    Inf     0    Inf     20      0    Inf    Inf    Inf
   Inf    Inf   Inf      0     30    Inf     35    Inf    Inf
   Inf    Inf   Inf    Inf      0    Inf      0     -8    Inf
   Inf    Inf   Inf    Inf    Inf      0     23      0    Inf
   Inf    Inf   Inf    Inf    Inf    Inf      0     37    Inf
   Inf    Inf   Inf    Inf    Inf    Inf    Inf      0    Inf
  46.2     28    23     18     18     15      8      0      0
```

```
Marcaje del algoritmo FF
     0     4     1
     3     1     1
     3     1     1
     4     1     1
     5     1     1
     6     1     1
     7     1     1
     8     1     1
     8     1     1

     1     2     1     2     2     3     5     6     9
     4     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1
```

```
Flujo actual
   215    75    32     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   178     0     0    37     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   148     0    67     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   108   212

Oferta y demanda
     3
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0     3


Marcaje del algoritmo FF
     0     3     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
     3     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

Nuevos costes marginales

0	15	30.7	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Nuevas capacidades

Inf	75	33	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 4

Costos relativos

0	15	30.7	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	3	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-15.7	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
30.7	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	3	1
0	0	0
3	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	1	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0

Nueva Mat Red

0	-23.7	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	13	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	17	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
38.7	0	8	0	0	0	0	0	0

Marcaje del algoritmo FF

0	3	1
0	0	0
3	1	1
0	0	0
0	0	0
6	1	1
0	0	0
0	0	0
0	0	0

1	0	1	0	0	3	0	0	0
3	0	1	0	0	1	0	0	0
1	0	1	0	0	1	0	0	0

Nueva Mat Red

0	-38.7	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	28	10	10	20	Inf	Inf	Inf
Inf	Inf	0	Inf	2	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
53.7	0	23	0	0	15	0	0	0

```
Marcaje del algoritmo FF
     0     3     1
     2     1     1
     3     1     1
     4     1     1
     5     1     1
     6     1     1
     7     1     1
     8     1     1
     8     1     1

     1     9     1     9     9     3     9     6     9
     3     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1

Flujo actual
   215    75    33     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   177     0     0    38     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   147     0    68     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   107   213

Oferta y demanda
     2
     0
     0
     0
     0
     0
     0
     0
     0

     0     0     0     0     0     0     0     0     2

Marcaje del algoritmo FF
     0     2     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
     2     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

Nuevos costes marginales

0	15	38.8	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Nuevas capacidades

Inf	75	34	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 5

Costos relativos

0	15	38.8	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	2	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-23.8	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
38.8	0	0	0	0	0	0	0	0

```
Marcaje del algoritmo FF
     0     2     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     0     0     0     0
     2     0     1     0     0     0     0     0     0
     1     0     1     0     0     0     0     0     0
```

Nueva Mat Red

0	-31.8	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	13	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	17	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
46.8	0	8	0	0	0	0	0	0

```
Marcaje del algoritmo FF
     0     2     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     6     1     1
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     3     0     0     0
     2     0     1     0     0     1     0     0     0
     1     0     1     0     0     1     0     0     0
```

Nueva Mat Red

0	-46.8	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	28	10	10	20	Inf	Inf	Inf
Inf	Inf	0	Inf	2	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
61.8	0	23	0	0	15	0	0	0

Marcaje del algoritmo FF

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

1	9	1	9	9	3	9	6	9
2	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1

Flujo actual

215	75	34	0	0	0	0	0	0
0	140	5	0	40	30	0	0	0
0	0	176	0	0	39	0	0	0
0	0	0	215	0	0	0	0	0
0	0	0	0	175	0	0	40	0
0	0	0	0	0	146	0	69	0
0	0	0	0	0	0	215	0	0
0	0	0	0	0	0	0	215	0
0	0	0	0	0	0	0	106	214

Oferta y demanda

1
0
0
0
0
0
0
0
0

0	0	0	0	0	0	0	0	1

Marcaje del algoritmo FF

0	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nuevos costes marginales

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Nuevas capacidades

Inf	75	35	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 6

Costos relativos

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
0	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-32.5	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
47.5	0	0	0	0	0	0	0	0

Marcaje del algoritmo FF

0	1	1
0	0	0
3	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0

Nueva Mat Red

0	-40.5	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	13	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	17	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	Inf
55.5	0	8	0	0	0	0	0	0

```
Marcaje del algoritmo FF
     0     1     1
     0     0     0
     3     1     1
     0     0     0
     0     0     0
     6     1     1
     0     0     0
     0     0     0
     0     0     0

     1     0     1     0     0     3     0     0     0
     1     0     1     0     0     1     0     0     0
     1     0     1     0     0     1     0     0     0
```

```
Nueva Mat Red
     0    -55.5      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0     28     10     10     20    Inf    Inf    Inf
   Inf    Inf      0    Inf      2      0    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     15      0    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0    Inf
  70.5      0     23      0      0     15      0      0      0
```

```
Marcaje del algoritmo FF
     0     1     1
     2     1     1
     3     1     1
     4     1     1
     5     1     1
     6     1     1
     7     1     1
     8     1     1
     8     1     1

     1     9     1     9     9     3     9     6     9
     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1
```

Alcanzando finalmente el óptimo:

```
flujo =

   215    75    35     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   175     0     0    40     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   145     0    70     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   105   215
```

Deshaciendo el cambio de variable para obtener el óptimo de la red original:

```
[flujoreal,Kreales,z]=postproccd(flujo,arcos,C,nodoscd,CDA)
```

```
flujoreal =

   215    75    35     0     0     0     0     0     0
     0   140     5     0    40    30     0     0     0
     0     0   175     0     0    40     0     0     0
     0     0     0   215     0     0     0     0     0
     0     0     0     0   175     0     0    40     0
     0     0     0     0     0   145     0    70     0
     0     0     0     0     0     0   215     0     0
     0     0     0     0     0     0     0   215     0
     0     0     0     0     0     0     0   105   215

Kreales =

   110     0     0     0     0     0     0  -110     0

z =

        3907.5
```

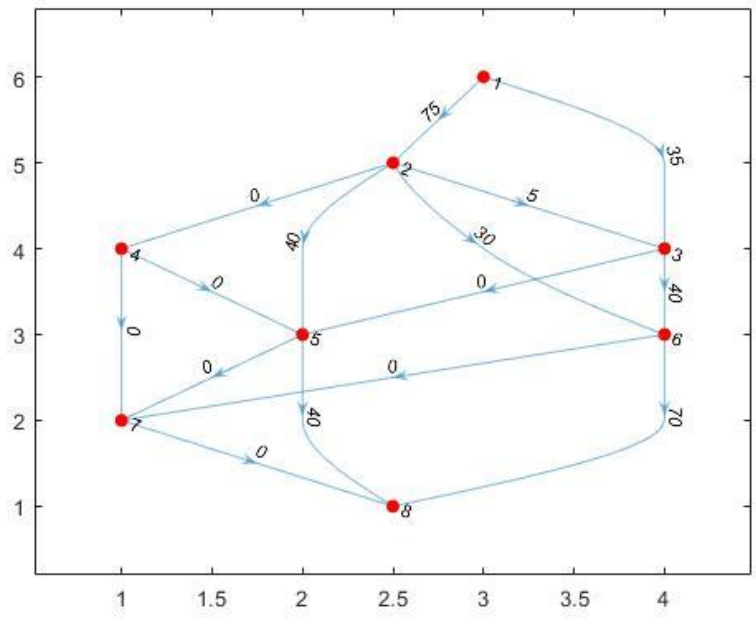


Figura 64.- Flujo óptimo en la red de la Figura 33 (método 2).

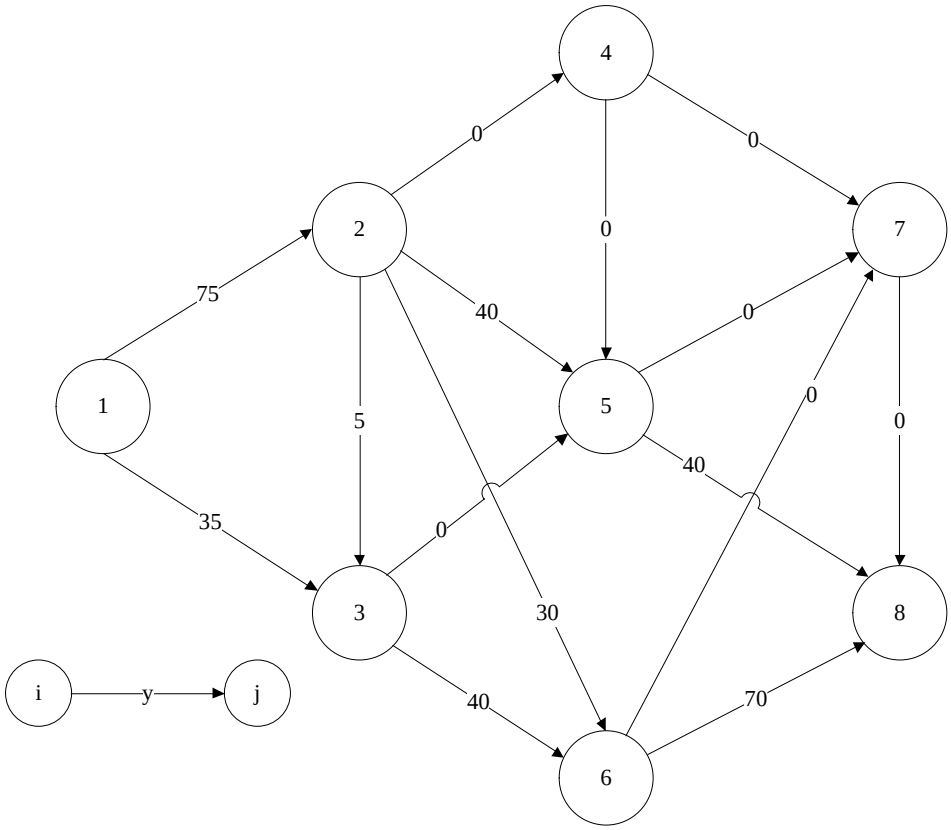


Figura 65.- Grafo realista equivalente al grafo de la Figura 64.

C.3. Método 3.

Respecto a la resolución empleando el método 3 para el equilibrado de la red, todos los resultados esenciales se recogen en el Apartado 3.7 (ver página 122). En dicho apartado, se muestran los resultados intermedios de la función "pdtcd" correspondientes únicamente a la primera iteración del algoritmo. En adelante, se indican el resto de resultados intermedios (desde la segunda iteración) de manera completa para el problema que se está abordando.

```
ITERACIÓN 2
Costos relativos
           0          15        16.3         Inf         Inf         Inf         Inf         Inf         Inf
         Inf           0           5          10          10           5         Inf         Inf         Inf
         Inf         Inf           0         Inf          25           8         Inf         Inf         Inf
         Inf         Inf         Inf           0          30         Inf          45         Inf         Inf
         Inf         Inf         Inf         Inf           0         Inf          10          10         Inf
         Inf         Inf         Inf         Inf         Inf           0          30          15         Inf
         Inf         Inf         Inf         Inf         Inf         Inf           0          45         Inf
         Inf         Inf         Inf         Inf         Inf         Inf         Inf           0           0
           0         Inf         Inf         Inf         Inf         Inf         Inf         Inf         Inf
```

Marcaje del algoritmo FF

1	80	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	190	1

9	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	0	1.3	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Marcaje del algoritmo FF

1	80	1
2	75	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	190	1

9	1	0	0	0	0	0	0	0
80	75	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	8.7	8.7	3.7	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     1     80      1
     2     75      1
     3      1      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    190      1

     9      1      1      0      0      0      0      0      0
    80     75      1      0      0      0      0      0      0
     1      1      1      0      0      0      0      0      0

Nueva Mat Red
     0      0      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0    3.7      5      5      0    Inf    Inf    Inf
   Inf    Inf      0    Inf   21.3    4.3    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     30     15    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf

Marcaje del algoritmo FF
     1     80      1
     2     75      1
     3      1      1
     0      0      0
     0      0      0
     6     30      1
     0      0      0
     0      0      0
     0    190      1

     9      1      1      0      0      2      0      0      0
    80     75      1      0      0     30      0      0      0
     1      1      1      0      0      1      0      0      0

Nueva Mat Red
     0      0      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0    3.7      0      0      0    Inf    Inf    Inf
   Inf    Inf      0    Inf   16.3    4.3    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     25     10    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1     80      1
     2     75      1
     3      1      1
     4     30      1
     5     50      1
     6     30      1
     0      0      0
     0      0      0
     0    190      1

     9      1      1      2      2      2      0      0      0
    80     75      1     30     50     30      0      0      0
     1      1      1      1      1      1      0      0      0
```

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	0	0	0	Inf	Inf	Inf
Inf	Inf	0	Inf	16.3	4.3	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	0	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     1     80      1
     2     75      1
     3      1      1
     4     30      1
     5     50      1
     6     30      1
     7     40      1
     8     30      1
     0    190      1

     9      1      1      2      2      2      5      6      8
    80     75      1     30     50     30     40     30     30
     1      1      1      1      1      1      1      1      1
```

```
Flujo actual
    50    30    30     0     0     0     0     0     0
     0    80     0     0     0    30     0     0     0
     0     0    80     0     0    30     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0   110     0     0     0     0
     0     0     0     0     0    50     0    60     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0    50    60
    60     0     0     0     0     0     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   160

     0     0     0     0     0     0     0     0   160


Marcaje del algoritmo FF
     1    50     1
     2    45     1
     3     1     1
     4    30     1
     5    45     1
     8    40     1
     7    40     1
     8    40     1
     0   160     1

     9     1     1     2     2     6     5     5     8
    50    45     1    30    45    40    40    40    40
     1     1     1     1     1     1     1     1     1
```

Flujo actual

10	70	30	0	0	0	0	0	0
0	40	0	0	40	30	0	0	0
0	0	80	0	0	30	0	0	0
0	0	0	110	0	0	0	0	0
0	0	0	0	70	0	0	40	0
0	0	0	0	0	50	0	60	0
0	0	0	0	0	0	110	0	0
0	0	0	0	0	0	0	10	100
100	0	0	0	0	0	0	0	0

Oferta y demanda

0
0
0
0
0
0
0
0
120

0	0	0	0	0	0	0	0	120

Marcaje del algoritmo FF

1	10	1
2	5	1
3	1	1
4	5	1
5	5	1
0	0	0
7	5	1
0	0	0
0	120	1

9	1	1	2	2	2	5	5	0
10	5	1	5	5	0	5	0	0
1	1	1	1	1	0	1	0	0

Nueva Mat Red

0	0	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	3.7	0	0	-4.3	Inf	Inf	Inf
Inf	Inf	0	Inf	16.3	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	35	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	0	-4.3	Inf
Inf	Inf	Inf	Inf	Inf	0	19.3	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	40.7	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     1    10     1
     2     5     1
     3     1     1
     4     5     1
     5     5     1
     6     1     1
     7     5     1
     8     1     1
     0   120     1

     9     1     1     2     2     3     5     6     8
    10     5     1     5     5     1     5     1     1
     1     1     1     1     1     1     1     1     1

Flujo actual
     9    70    31     0     0     0     0     0     0
     0    40     0     0    40    30     0     0     0
     0     0    79     0     0    31     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0    70     0     0    40     0
     0     0     0     0     0    49     0    61     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0     9   101
   101     0     0     0     0     0     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   119

     0     0     0     0     0     0     0     0   119
```

Marcaje del algoritmo FF

1	9	1
2	5	1
0	0	0
4	5	1
5	5	1
0	0	0
7	5	1
0	0	0
0	119	1

9	1	1	2	2	0	5	0	0
9	5	0	5	5	0	5	0	0
1	1	0	1	1	0	1	0	0

Nuevos costes marginales

0	15	23.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Nuevas capacidades

Inf	75	32	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	215
110	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 3

Costos relativos

0	15	23.2	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     1      9      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    119      1

     9      0      0      0      0      0      0      0      0
     9      0      0      0      0      0      0      0      0
     1      0      0      0      0      0      0      0      0

Nueva Mat Red
     0      0    8.2    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0      5     10     10      5    Inf    Inf    Inf
   Inf    Inf      0    Inf     25      8    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     30     15    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1      9      1
     2      5      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    119      1

     9      1      0      0      0      0      0      0      0
     9      5      0      0      0      0      0      0      0
     1      1      0      0      0      0      0      0      0

Nueva Mat Red
     0      0    3.2    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0      0      5      5      0    Inf    Inf    Inf
   Inf    Inf      0    Inf     25      8    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     30     15    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1     9     1
     2     5     1
     3     5     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0   119     1

     9     1     2     0     0     2     0     0     0
     9     5     5     0     0     0     0     0     0
     1     1     1     0     0     0     0     0     0
```

```
Nueva Mat Red
     0     0   3.2   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     0     0     0    -5   Inf   Inf   Inf
   Inf   Inf     0   Inf    20     3   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    45   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf    10    10   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

```
Marcaje del algoritmo FF
     1     9     1
     2     5     1
     3     5     1
     4     5     1
     5     5     1
     0     0     0
     0     0     0
     0     0     0
     0   119     1

     9     1     2     2     2     0     0     0     0
     9     5     5     5     5     0     0     0     0
     1     1     1     1     1     0     0     0     0
```

```
Nueva Mat Red
     0     0   3.2   Inf   Inf   Inf   Inf   Inf   Inf
   Inf     0     0     0     0    -8   Inf   Inf   Inf
   Inf   Inf     0   Inf    20     0   Inf   Inf   Inf
   Inf   Inf   Inf     0    30   Inf    42   Inf   Inf
   Inf   Inf   Inf   Inf     0   Inf     7     7   Inf
   Inf   Inf   Inf   Inf   Inf     0    30    15   Inf
   Inf   Inf   Inf   Inf   Inf   Inf     0    45   Inf
   Inf   Inf   Inf   Inf   Inf   Inf   Inf     0     0
     0   Inf   Inf   Inf   Inf   Inf   Inf   Inf   Inf
```

```
Marcaje del algoritmo FF
     1      9      1
     2      5      1
     3      5      1
     4      5      1
     5      5      1
     6      5      1
     0      0      0
     0      0      0
     0    119      1

     9      1      2      2      2      3      0      0      0
     9      5      5      5      5      5      0      0      0
     1      1      1      1      1      1      0      0      0

Nueva Mat Red
     0      0    3.2    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0      0      0      0     -8    Inf    Inf    Inf
   Inf    Inf      0    Inf     20      0    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     35    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf      0      0    Inf
   Inf    Inf    Inf    Inf    Inf      0     23      8    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1      9      1
     2      5      1
     3      5      1
     4      5      1
     5      5      1
     6      5      1
     7      5      1
     0      0      0
     0    119      1

     9      1      2      2      2      3      5      5      0
     9      5      5      5      5      5      5      0      0
     1      1      1      1      1      1      1      0      0

Nueva Mat Red
     0      0    3.2    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0      0      0      0     -8    Inf    Inf    Inf
   Inf    Inf      0    Inf     20      0    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     35    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf      0     -8    Inf
   Inf    Inf    Inf    Inf    Inf      0     23      0    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     37    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1      9      1
     2      5      1
     3      5      1
     4      5      1
     5      5      1
     6      5      1
     7      5      1
     8      5      1
     0    119      1

     9      1      2      2      2      3      5      6      8
     9      5      5      5      5      5      5      5      5
     1      1      1      1      1      1      1      1      1
Flujo actual
     4     75     31      0      0      0      0      0      0
     0     35      5      0     40     30      0      0      0
     0      0     74      0      0     36      0      0      0
     0      0      0    110      0      0      0      0      0
     0      0      0      0     70      0      0     40      0
     0      0      0      0      0     44      0     66      0
     0      0      0      0      0      0    110      0      0
     0      0      0      0      0      0      0      4    106
   106      0      0      0      0      0      0      0      0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   114

     0      0      0      0      0      0      0      0    114
```

```
Marcaje del algoritmo FF
     1      4      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    114      1

     9      1      0      0      0      0      0      0      0
     4      0      0      0      0      0      0      0      0
     1      0      0      0      0      0      0      0      0
```

```
Nueva Mat Red
     0     -3.2      0     Inf     Inf     Inf     Inf     Inf     Inf
   Inf        0      0       0       0      -8     Inf     Inf     Inf
   Inf      Inf      0     Inf      20       0     Inf     Inf     Inf
   Inf      Inf    Inf       0      30     Inf      35     Inf     Inf
   Inf      Inf    Inf     Inf       0     Inf       0      -8     Inf
   Inf      Inf    Inf     Inf     Inf       0      23       0     Inf
   Inf      Inf    Inf     Inf     Inf     Inf       0      37     Inf
   Inf      Inf    Inf     Inf     Inf     Inf     Inf       0       0
     0      Inf    Inf     Inf     Inf     Inf     Inf     Inf     Inf
```

```
Marcaje del algoritmo FF
     1      4      1
     3      1      1
     3      1      1
     4      1      1
     5      1      1
     6      1      1
     7      1      1
     8      1      1
     0    114      1

     9      2      1      2      2      3      5      6      8
     4      1      1      1      1      1      1      1      1
     1      1      1      1      1      1      1      1      1
```

```
Flujo actual
     3     75     32      0      0      0      0      0      0
     0     35      5      0     40     30      0      0      0
     0      0     73      0      0     37      0      0      0
     0      0      0    110      0      0      0      0      0
     0      0      0      0     70      0      0     40      0
     0      0      0      0      0     43      0     67      0
     0      0      0      0      0      0    110      0      0
     0      0      0      0      0      0      0      3    107
   107      0      0      0      0      0      0      0      0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   113

     0      0      0      0      0      0      0      0    113


Marcaje del algoritmo FF
     1      3      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    113      1

     9      0      1      0      0      0      0      0      0
     3      0      0      0      0      0      0      0      0
     1      0      0      0      0      0      0      0      0
```

Nuevos costes marginales

0	15	30.7	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Nuevas capacidades

Inf	75	33	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	215
110	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 4

Costos relativos

0	15	30.7	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Marcaje del algoritmo FF

1	3	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	113	1

9	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-15.7	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Marcaje del algoritmo FF

1	3	1
0	0	0
3	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	113	1

9	0	1	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0

Nueva Mat Red

0	-23.7	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	13	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	17	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Marcaje del algoritmo FF

1	3	1
0	0	0
3	1	1
0	0	0
0	0	0
6	1	1
0	0	0
0	0	0
0	113	1

9	0	1	0	0	3	0	0	0
3	0	1	0	0	1	0	0	0
1	0	1	0	0	1	0	0	0

Nueva Mat Red

0	-38.7	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	28	10	10	20	Inf	Inf	Inf
Inf	Inf	0	Inf	2	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     1      3      1
     0      0      0
     3      1      1
     0      0      0
     0      0      0
     6      1      1
     0      0      0
     8      1      1
     0    113      1

     9      0      1      0      0      3      0      6      8
     3      0      1      0      0      1      0      1      1
     1      0      1      0      0      1      0      1      1

Flujo actual
     2     75     33      0      0      0      0      0      0
     0     35      5      0     40     30      0      0      0
     0      0     72      0      0     38      0      0      0
     0      0      0    110      0      0      0      0      0
     0      0      0      0     70      0      0     40      0
     0      0      0      0      0     42      0     68      0
     0      0      0      0      0      0    110      0      0
     0      0      0      0      0      0      0      2    108
   108      0      0      0      0      0      0      0      0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   112

     0      0      0      0      0      0      0      0    112
```

Marcaje del algoritmo FF

1	2	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	112	1

9	0	1	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nuevos costes marginales

0	15	38.8	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Nuevas capacidades

Inf	75	34	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	215
110	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 5
Costos relativos

0	15	38.8	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     1      2      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    112      1

     9      0      0      0      0      0      0      0      0
     2      0      0      0      0      0      0      0      0
     1      0      0      0      0      0      0      0      0
```

```
Nueva Mat Red
     0    -23.8      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0      5     10     10      5    Inf    Inf    Inf
   Inf    Inf      0    Inf     25      8    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     30     15    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1      2      1
     0      0      0
     3      1      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    112      1

     9      0      1      0      0      0      0      0      0
     2      0      1      0      0      0      0      0      0
     1      0      1      0      0      0      0      0      0
```

```
Nueva Mat Red
     0    -31.8      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0     13     10     10      5    Inf    Inf    Inf
   Inf    Inf      0    Inf     17      0    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     30     15    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

Marcaje del algoritmo FF

1	2	1
0	0	0
3	1	1
0	0	0
0	0	0
6	1	1
0	0	0
0	0	0
0	112	1

9	0	1	0	0	3	0	0	0
2	0	1	0	0	1	0	0	0
1	0	1	0	0	1	0	0	0

Nueva Mat Red

0	-46.8	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	28	10	10	20	Inf	Inf	Inf
Inf	Inf	0	Inf	2	0	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	15	0	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Marcaje del algoritmo FF

1	2	1
0	0	0
3	1	1
0	0	0
0	0	0
6	1	1
0	0	0
8	1	1
0	112	1

9	0	1	0	0	3	0	6	8
2	0	1	0	0	1	0	1	1
1	0	1	0	0	1	0	1	1

```
Flujo actual
     1    75    34     0     0     0     0     0     0
     0    35     5     0    40    30     0     0     0
     0     0    71     0     0    39     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0    70     0     0    40     0
     0     0     0     0     0    41     0    69     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0     1   109
   109     0     0     0     0     0     0     0     0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   111

     0     0     0     0     0     0     0     0   111


Marcaje del algoritmo FF
     1     1     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0   111     1

     9     0     1     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0
```

Nuevos costes marginales

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Nuevas capacidades

Inf	75	35	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	70	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	215
110	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 6

Costos relativos

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Marcaje del algoritmo FF

1	1	1
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	111	1

9	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

Nueva Mat Red

0	-32.5	0	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	15	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     1      1      1
     0      0      0
     3      1      1
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0      0      0
     0    111      1

     9      0      1      0      0      0      0      0      0
     1      0      1      0      0      0      0      0      0
     1      0      1      0      0      0      0      0      0
```

```
Nueva Mat Red
     0    -40.5      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0     13     10     10      5    Inf    Inf    Inf
   Inf    Inf      0    Inf     17      0    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     30     15    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1      1      1
     0      0      0
     3      1      1
     0      0      0
     0      0      0
     6      1      1
     0      0      0
     0      0      0
     0    111      1

     9      0      1      0      0      3      0      0      0
     1      0      1      0      0      1      0      0      0
     1      0      1      0      0      1      0      0      0
```

```
Nueva Mat Red
     0    -55.5      0    Inf    Inf    Inf    Inf    Inf    Inf
   Inf      0     28     10     10     20    Inf    Inf    Inf
   Inf    Inf      0    Inf      2      0    Inf    Inf    Inf
   Inf    Inf    Inf      0     30    Inf     45    Inf    Inf
   Inf    Inf    Inf    Inf      0    Inf     10     10    Inf
   Inf    Inf    Inf    Inf    Inf      0     15      0    Inf
   Inf    Inf    Inf    Inf    Inf    Inf      0     45    Inf
   Inf    Inf    Inf    Inf    Inf    Inf    Inf      0      0
     0    Inf    Inf    Inf    Inf    Inf    Inf    Inf    Inf
```

```
Marcaje del algoritmo FF
     1      1      1
     0      0      0
     3      1      1
     0      0      0
     0      0      0
     6      1      1
     0      0      0
     8      1      1
     0    111      1

     9      0      1      0      0      3      0      6      8
     1      0      1      0      0      1      0      1      1
     1      0      1      0      0      1      0      1      1

Flujo actual
     0     75     35      0      0      0      0      0      0
     0     35      5      0     40     30      0      0      0
     0      0     70      0      0     40      0      0      0
     0      0      0    110      0      0      0      0      0
     0      0      0      0     70      0      0     40      0
     0      0      0      0      0     40      0     70      0
     0      0      0      0      0      0    110      0      0
     0      0      0      0      0      0      0      0    110
   110      0      0      0      0      0      0      0      0

Oferta y demanda
     0
     0
     0
     0
     0
     0
     0
     0
   110

     0      0      0      0      0      0      0      0    110
```

Marcaje del algoritmo FF

0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	110	1

9	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Nuevos costes marginales

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	29.4	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

Nuevas capacidades

Inf	75	35	Inf	Inf	Inf	Inf	Inf	Inf
Inf	Inf	40	30	50	30	Inf	Inf	Inf
Inf	Inf	Inf	Inf	40	60	Inf	Inf	Inf
Inf	Inf	Inf	Inf	60	Inf	100	Inf	Inf
Inf	Inf	Inf	Inf	Inf	Inf	40	40	Inf
Inf	Inf	Inf	Inf	Inf	Inf	50	71	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	100	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf	215
110	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

ITERACIÓN 7

Costos relativos

0	15	47.5	Inf	Inf	Inf	Inf	Inf	Inf
Inf	0	5	10	10	5	Inf	Inf	Inf
Inf	Inf	0	Inf	25	8	Inf	Inf	Inf
Inf	Inf	Inf	0	30	Inf	45	Inf	Inf
Inf	Inf	Inf	Inf	0	Inf	10	10	Inf
Inf	Inf	Inf	Inf	Inf	0	30	29.4	Inf
Inf	Inf	Inf	Inf	Inf	Inf	0	45	Inf
Inf	Inf	Inf	Inf	Inf	Inf	Inf	0	0
0	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf

```
Marcaje del algoritmo FF
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0   110     1

     9     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0
```

Obteniendo finalmente el óptimo:

```
flujo =

     0    75    35     0     0     0     0     0     0
     0    35     5     0    40    30     0     0     0
     0     0    70     0     0    40     0     0     0
     0     0     0   110     0     0     0     0     0
     0     0     0     0    70     0     0    40     0
     0     0     0     0     0    40     0    70     0
     0     0     0     0     0     0   110     0     0
     0     0     0     0     0     0     0     0   110
   110     0     0     0     0     0     0     0     0
```

REFERENCIAS

[1] A. Cortéz, «Teoría de la Complejidad Computacional y Teoría de la Computabilidad,» *Rev. investig. sist. inform.*, nº 1(1), pp. 102-105, 2004.

[2] K. G. Murty, «Chapter 5. Algorithms for minimum cost flow problems in pure networks,» de *Network Programming*, New Jersey, Prentice Hall, 1992, pp. 384-520.

[3] R. K. Ahuja, T. L. Magnanti y J. B. Orlin, «Chapter 9. Minimum cost flow: basic algortihms,» de *Network Flows. Theory, Algorithms and Applications*, New Jersey, Prentice Hall, 1993, pp. 294-356.

[4] D. Bertsekas, «A unified framework for primal-dual methods in minimum cost network flow problems,» *Mathematical Programming*, nº 32, pp. 125-145, 1985.

[5] D. Rani y M. Moreira, «Simulation-Optimization Modeling: a Survey Potential Application in Reservoir Systems Operation,» *Water Resour Manage*, nº 24, pp. 1107-1138, 2010.

[6] G. Kuczera, «Network Linear Programming Codes for Water Supply Headworks Modeling,» *Journal of Water Resourcs Plannig and Management*, nº 119, 1993.

[7] R. K. Ahuja, T. L. Magnanti y J. B. Orlin, «Chapter 10. Minimum cost flows: polynomial algorithms,» de *Network Flows. Theory, Algorithms ans Applications*, New Jersey, Prentice Hall, 1993, pp. 357-401.

[8] B. Ghasemishabankabareh, X. Li, M. Ozlen y F. Neumann, «Probabilistic tree-based representation for solving minimum cost integer flow problems with nonlinear non-convex cost functions,» *Applied Soft Computing Journal*, nº 86, 2020.

[9] D. Fontes, E. Hadjiconstantinou y N. Christofides, «A dynamic programming approach for solving single-source uncapacited concave minimum cost network problems,» *European J. Oper. Res*, nº 174(2), pp. 1205-1219, 2006.

[10] R. E. Burkard, H. Dollani y P. T. Thach, «Linear approximations in a dynamic programming approach for the uncapacited single-source minimum concave cost network flows in acylic networks,» *J. Global Optim.*, nº 19 (2), pp. 121-139, 2001.

[11] M. Minoux, «Solving integer minimum cost flows with separable convex cost objective polynomially,» *Mathematical Programming Study*, nº 26, pp. 237-239, 1986.

[12] M. Minoux, «A polynomial algorithm for minimum quadratic cost flow problems,» *European Journal of Operational Research*, nº 18, pp. 377-387, 1984.

[13] R. Cottle y J. Pang, «On the convergence of a block successive overrelaxation method for a class of linear complementarity problems,» *Technical Report SOL 80-17. Dept Operations Researck, Stanford University, CA*, 1980.

[14] P. Wolfe, «The simplex method for quadratic programming,» *Econometrica.,* nº 27(3), pp. 382-398, 1959.

[15] C. Lemke, «A method of solution for quadratic programs,» *Management Science,* nº 8, pp. 442-455, 1962.

[16] R. Ahuja, D. Hochbaum y J. Orlin, «A cut-based algorithm for the nonlinear dual of the minimum cost network flow problem,» *Algorithmica,* nº 39, pp. 189-208, 2004.

[17] R. Bellman, «On the theory of dynamic programming,» *Proceeding of the National Academy of Sciences,* nº 38(8), 1952.

[18] R. Erickson, C. Monma y A. Veinott Jr, «Send-and-split method for minimum-concave-cost network flows,» *Math Oper Res,* nº 12(4), pp. 634-664, 1987.

[19] P. Kovacs, «Minimum-cost flow algorithms: an experimental evaluation,» *Optim Methods Softw,* nº 30(1), pp. 94-127, 2015.

[20] H. Kim y J. Hooker, «Solving fixed-charge network flow problems with a hybrid optimization and constraint programming approach,» *Ann Oper Res,* nº 115(1-4), pp. 95-124, 2002.

[21] D. Fontes, E. Hadjiconstantinou y N. Christofides, «A branch-and-bound algorithm for concave network flow problems,» *J Glob Optim,* nº 34(1), pp. 127-155, 2006.

[22] G. Guisewite y P. Pardalos, «Algorithms for the single-source uncapacited minimum concave-cost network flow problem,» *J Glob Optim,* nº 1(3), pp. 245-265, 1991.

[23] R. Horst y N. Thoai, «An integer concave minimization approach for the minimum concave cost capacited flow problem in networks,» *Oper Res Spektrum,* nº 20(1), pp. 47-53, 1998.

[24] W. Gutjahr, «ACO algorithms with guaranteed convergence to optimal solutions,» *Information processing letters,* nº 82(3), pp. 145-153, 2002.

[25] M. Toksari, «Ant colony optimization for finding the global minimum,» *Applied Mathematics and Computation,* nº 176, pp. 308-316, 2006.

[26] M. Monteiro, D. Fontes y F. Fontes, «An ant colony optimization algorithm to solve the minimum cost network flow problem with concave cost functions,» *Proceedings of the 13 annual conference on Genetic and evolutionary computation,* pp. 139-146, 2011.

[27] M. Monteiro, D. Fontes y F. Fontes, «Concave minimum cost network flow problems solved with a colony of ants,» *J Heuristics,* nº 19(1), pp. 1-33, 2013.

[28] S. Yan, Y. Shih y C. Wang, «An ant colony sistem-based hybrid algorithm for square root concave cost transhipment problems,» *Eng Optim,* nº 42(11), pp. 983-1001, 2010.

[29] F. Xie y R. Jia, «Nonlinear fixed charge transportation problem by minimum cost flow-based genetic algorithm,» *Comput Ind Eng,* nº 63(4), pp. 763-778, 2012.

[30] Z. Michaelewicz, G. Vignaux y M. Hobbs, «A nonstandard genetic algorithm for the nonlinear transportation problem,» *ORSA J Comput,* nº 3(4), pp. 307-316, 1991.

[31] B. Ghasemishabankareh, M. Ozlen, X. Li y K. Deb, «A genetic algorithm with local search for solving single-source single-sink nonlinear non-convex minimum cost flow problems,» *Soft Computing*, nº 24, pp. 1153-1169, 2020.

[32] F. Tar y Z. Hashemi, «A priority based genetic algorithm for nonlinear transportation cost problems,» *Comput. Ind. Eng.*, nº 96, pp. 86-95, 2016.

[33] J. Mendes, J. Gonçalves y M. Resende, «A random key based genetic algorithm for the resource constrained project scheduling problem,» *Comput. Oper, Red.*, nº 36(1), pp. 92-109, 2009.

[34] B. Fontes y J. Gonçalves, «A multi-population hybrid based random key genetic algorithm for hop-constrained trees in nonlinear cost flow networks,» *Optim. Lett.*, nº 7(6), pp. 1303-1324, 2013.

[35] F. García Benítez, Redes de Transporte. Teoría y algoritmos básicos, Sevilla: TED Technical Editions, 2003.