Cálculo del Flujo Máximo en una Red (Grafo Dirigido)

Gean Piers Marín Gonzales

Resumen— El presente proyecto de fin de grado esboza una solución informática al problema del flujo máximo, para lo cual se ha optado por utilizar el algoritmo de Ford-Fulkerson, al ser éste el más conocido y difundido, y que permite llegar a una solución exacta del problema en un tiempo relativamente corto. Dicho problema tiene una amplia gama de aplicaciones, que van desde cálculo de rutas disjuntas para redes de comunicaciones, circulación con capacidad, programación de líneas aéreas, selección de proyectos, entre otras. El problema del flujo máximo fundamentalmente consiste en: dada una red (o grafo) de arcos y nodos, cada arco con una capacidad determinada, y con un nodo fuente y otro destino, se trata de hallar la cantidad máxima de flujo que puede circular desde el nodo fuente hasta el nodo destino, de manera que el flujo individual que va por cada arco no supere la capacidad de sí mismo; esto último es conocido como restricción de capacidad del arco. Existe una vasta y variada cantidad de contextos que pueden modelarse como un problema de flujo máximo, las principales serán brevemente explicadas en la memoria descriptiva.

Palabras clave—Grafos, flujo máximo, algoritmo Ford-Fulkerson, nodo origen, nodo destino, arcos, capacidad, aplicativo, Java, TXT.

Abstract— The present project of end of degree outlines a computer solution to the problem of the maximum flow, for which it has chosen to use the algorithm of Ford-Fulkerson, being the one most known and diffused, and that allows to arrive to an exact solution of the problem in a relatively short time. This problem has a wide range of applications, ranging from calculation of disjoint routes for communications networks, capacity circulation, airline programming, project selection, among others. The problem of maximum flow basically consists of: given a network (or directed graph) of arcs and nodes, each arc with a determined capacity, and with a source node and target node, it is a matter of finding the maximum amount of flow that can flow from the source node to the destination node, so that the individual flow going through each arc does not exceed the capacity of itself; the last is known as the arc capacity constraint. There is a vast and varied number of contexts that can be modeled as a problem of maximum flow, the main ones will be briefly explained in this document.

Index Terms— Graphs, Max Flow, algorithm Ford-Fulkerson, Source Node, Target Node, Arcs, Capacity, Application, Java, TXT.

0 Introducción

El presente proyecto de fin de grado comprende el desarrollo de un software que permite obtener la solución al problema del flujo máximo, el cual forma parte del estudio de los modelos de redes, tema perteneciente al área de investigación operativa. Dicho software, constituye una herramienta que puede ser utilizada tanto en la enseñanza de dicha materia como en aplicaciones prácticas del mencionado modelo en situaciones de la vida real. Los problemas estudiados por la investigación operativa tratan, básicamente, de obtener una solución óptima a una situación planteada, sujeta a un determinado número de restricciones. Uno de los temas más importantes e interesantes de dicha ciencia es el modelo de redes, el cual hace uso de arcos y nodos para la representación gráfica y mejor comprensión de la situación objeto de estudio, uno de los cuales es el ya mencionado problema del flujo máximo. El problema del flujo máximo consiste en calcular la cantidad máxima de flujo que puede ser transportada de un punto de partida u origen a uno de llegada o destino [1]. Entiéndase por flujo, una cantidad medible de materia que puede circular por cualquiera de los arcos. Dicho flujo está sometido a las restricciones de capacidad de cada arco, es decir, a la cantidad máxima de flujo que puede soportar cada uno de los mismos. Ahora bien, el material que fluye a través de la red dependerá de cada situación particular donde el problema se aplique, por ejemplo, el flujo será un líquido en modelos de tuberías, aviones en modelos de programación de aerolíneas, etc. Como en el presente proyecto se estudia el modelo general del problema del flujo máximo, se hará mención del concepto de "flujo" en términos generales. El objetivo del proyecto es implementar un software que permita resolver automáticamente el problema del flujo máximo, el cual pueda ser utilizado para la enseñanza de la investigación de operaciones, así como en aplicaciones prácticas que se basen en dicho problema. Para la implementación se ha seguido el modelo en cascada o ciclo

- *E-mail del contacto: geanpiers.marin@e-campus.uab.cat*
- *Mención realizada: Tecnologies de la Informació.*
- *Trabajo tutorizado por: Joaquim Borges Ayats (Departament d'Enginyeria de la Informació i de les Comunicacions)*
- *Curso 2016/17*

de vida clásico del software [2], y para la redacción del documento se ha seguido los documentos de proyectos del grado en Ingeniería Informática [3], la guía de elaboración del documento del grado [4] y los índices base [5], los cuales fueron redactados por los docentes de la Universidad Autónoma de Barcelona, de la Escuela de Ingeniería, especialidad de Ingeniería Informática, y que están publicadas en Internet. Así, el informe descriptivo se ha organizado en seis capítulos, los cuales se explican a continuación:

1. Capítulo 1: Objetivos: Capítulo en el cual se establece la definición del problema que se va a resolver y a donde se pretende llegar en su desarrollo.

2. Capítulo 2: Generalidades: Explicación de la teoría fundamental necesaria para entender la situación estudiada.

3. Capítulo 3: Teoría Previa: Explicación de la teoría fundamental del algoritmo Ford-Fulkerson.

4. Capítulo 4: Estado del Arte: Se detalla los métodos de solución existentes de dicho problema y algunas aplicaciones de software que permiten calcular la solución al mismo.

5. Capítulo 5: Metodología: En este capítulo se describe la planificación del proyecto, la descripción de la solución elaborada, la arquitectura de la solución y se sintetizan los principales puntos de la implementación del software.

6. Capítulo 6: Resultados: Capítulo destinado al diseño de la interfaz gráfica, los resultados obtenidos del mismo y algunos consejos sobre el uso y ampliación del producto desarrollado.

7. Capítulo 7: Conclusiones: Conclusiones y recomendaciones. Se mencionan los puntos más resaltantes del proyecto.

Finalmente, se incluye en el documento los anexos que sustentan el desarrollo del proyecto y que corresponden al software elaborado según la metodología seguida.

1 OBJETIVOS

Los objetivos de este proyecto se centran en una aplicación que nos facilite el cálculo del flujo máximo de una red, utilizando uno de los algoritmos más famosos de todos los tiempos, el de Ford-Fulkerson. Esta aplicación nos presentará una interfaz intuitiva para poder generar todo tipo de grafos, ya sea a través de introducir los datos previos a mano o cargar los grafos a partir de un fichero de texto.

Una vez conseguido el grafo haremos uso del método para poder dar una solución al problema del flujo máximo, poder mirar las iteraciones que se realizan para llegar a una solución y con este resultado poder analizar las distintas variantes que se pueda tener, como:

- Redes con diversos orígenes y destinos.
- Redes con capacidad en los nodos (y en los arcos)
- Redes con capacidad mínimas en los arcos y búsqueda del flujo mínimo.

2 GENERALIDADES

En este apartado se hará la explicación de la teoría fundamental necesaria para entender la situación estudiada.

2.1 Definición del Problema

Previamente a la definición del problema del flujo máximo, es importante puntualizar el concepto de red o grafo, el cual es un conjunto de varios nodos (puntos en el espacio) y arcos (líneas) que salen de un determinado nodo y van hacia otro. Los grafos pueden ser dirigidos, cuando cada arco tiene un nodo origen y uno destino (es decir los arcos tienen un sentido), o no dirigidos en caso contrario. Las imágenes 1 y 2 ilustran de modo gráfico los conceptos de grafo no dirigido y dirigido, asimismo, más adelante en el presente capítulo se establece la definición formal y matemática de los grafos.

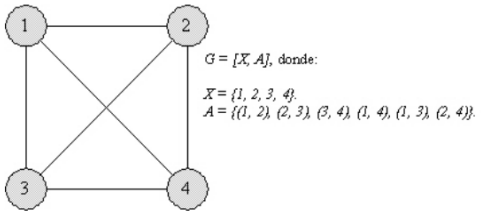


Imagen 1: Ejemplo grafo no dirigido

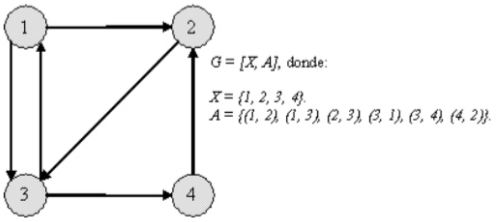


Imagen 2: Ejemplo grafo dirigido

2.2 Flujo Máximo

Dicho lo anterior, el problema del flujo máximo en su forma más pura consiste en que: Existe un grafo dirigido o no dirigido (comúnmente dirigido en la mayoría de aplicaciones reales), donde uno de los vértices es considerado como el "origen" y otro como el "destino", de tal manera que algún material u objeto puede fluir desde el origen hasta el destino; a la cantidad de material u objeto que circula por el grafo se le denomina flujo. Entre el origen y el destino existe una cantidad determinada de nodos interconectados entre sí a través de arcos, cada uno de estos arcos tiene una capacidad máxima que puede transportar entre los nodos que conecta, la cual puede variar de un arco a otro. Esto quiere decir, que cada arco solo podrá soportar un flujo menor o igual a su capacidad, de tal manera que, si un flujo mayor quiere discurrir a través de un arco, solo una parte de dicho flujo (de valor igual a la capacidad de ese arco) viajará a través de él, y el resto deberá ir por otro arco que salga del mismo nodo, de no haber otro arco, entonces el flujo se verá reducido.

El objetivo del problema del flujo máximo es determinar la máxima cantidad de material u objetos (flujo) que pueden fluir en el grafo desde la fuente hacia el sumidero. En aplicaciones del mundo real, conocer el valor del flujo máximo permite a la fuente saber exactamente cuánto producir y enviar a través de una ruta sin generar desperdicios. [6]

2.3 Redes con múltiples orígenes y destinos

En las diferentes aplicaciones del problema del flujo máximo, las redes de flujo generadas no necesariamente poseen un solo origen y un solo destino, sino que pueden tener dos, tres o más de estos. Sin embargo, cualquiera que sea la cantidad de orígenes o destinos, esta situación se puede reducir a un problema de flujo máximo ordinario, lo único que se debe hacer es agregar una súper-origen con arcos de capacidad infinita (o muy grande) que partan de ella y vayan hacia cada una de los orígenes originales de la red. Similarmente, se agrega un súper-destino con arcos de capacidad infinita (o muy grande) que partan de cada uno de los destinos originales de la red y vayan hacia el súper-destino creado. Hecho esto, el problema se verá reducido a un problema de flujo máximo conocido. [7]

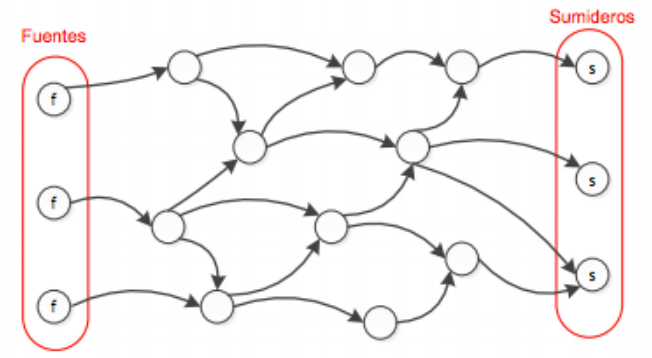


Imagen 3: Red con varios orígenes y destinos

2.4 Redes con capacidad en los nodos

La solución de esta variante es sustituir cada vértice v_k (de capacidad q) por dos vértices virtuales v_k^- y v_k^+, sin capacidad; v_k^- recibe todos los arcos que recibía v_k, y de v_k^+ salen todos los arcos que salían de v_k. Además, se deberá agregar un arco de capacidad q que unan v_k^- y v_k^+. Se resolverá el problema clásico con los nuevos vértices y los nuevos arcos. [8]

3 Teoría Previa

A continuación, antes del teorema y el algoritmo nos tocara introducir un poco de notación. A partir de aquí, supondremos siempre que *G (V, A)* es una red de transporte donde cada arco *(u, v)* tiene capacidad q y flujo f. Naturalmente, asumimos siempre que el flujo es compatible.

3.1 Métodos de solución

El principal método de solución del problema del flujo máximo es el de Ford-Fulkerson, aunque también se han desarrollado algunas mejoras al mismo. También existen otros algoritmos que, aunque no calculan exactamente el valor del flujo máximo, calculan valores suficientemente cercanos al mismo, es decir, son métodos aproximados.

3.1.1 Método de Ford – Fulkerson

El método de Ford-Fulkerson, considerado un método más que un algoritmo [7], está basado en tres conceptos: red residual, trayectorias de aumento y cortes; así como en el teorema del flujo máximo / corte mínimo. [25]

3.1.1.1 Red residual

La red residual es aquella que deriva de una red original y que está constituida por los arcos de dicha red que pueden admitir más flujo. A partir de este concepto se puede definir la capacidad residual de un arco, la cual es la cantidad de flujo restante que todavía puede aceptar dicho arco. A los arcos que pertenecen a la red residual, se les denomina arcos residuales, además un arco (u,v) de una red puede aparecer en la red residual solo si por lo menos uno de los arcos (u,v) o (v,u) aparecen en la red original. [7]

3.1.1.2 Trayectoria de aumento

Se denomina trayectoria de aumento a una trayectoria simple que va del origen al destino en la red residual. Por ende, cada uno de sus arcos admite cierta cantidad de flujo positivo sin violar la restricción de capacidad de los mismos. Además, la máxima cantidad en la que se puede incrementar el flujo en cada uno de los arcos de una trayectoria de aumento de tal manera que dicho incremento sea el mismo en todos los arcos, pero también se puede tomar los arcos en sentido contrario para bajarles el flujo (siempre y cuando no tengan un flujo de 0), se denomina capacidad residual de la trayectoria de aumento. [7]

3.1.1.3 Corte de red de flujo

Un corte es una división de la red de flujo en dos redes, una de las cuales contiene al origen y la otra al destino. La capacidad de un corte es igual a la suma de las capacidades de los arcos que van desde la parte que contiene al origen, hacia la parte que contiene al destino. Basado en estos conceptos, se denomina corte mínimo a aquel corte que posee la mínima capacidad entre todos los cortes que se le pueden hacer a la red de flujo. El corte mínimo es importante pues su capacidad limita la cantidad de flujo que puede circular a través de la red. [7]

3.1.1.4 Teorema de flujo máximo-corte mínimo

Sea f un flujo que circula en una red de flujo $G = (V, E)$, con origen s y destino t, f es el flujo máximo de G si la red residual de G no contiene trayectorias de aumento o el valor de f es igual a la capacidad de alguno de los cortes de G, ambas reglas son equivalentes.

Básicamente, el teorema establece que el flujo máximo que puede circular a través de una red es igual a la capacidad del corte mínimo de la misma. Este teorema es la base para el diseño del algoritmo de Ford-Fulkerson. [7]

3.1.1.5 Algoritmo de Ford-Fulkerson

El algoritmo inicia otorgando un valor de flujo de cero a cada uno de los arcos de la red. Luego, se pasa a una fase iterativa, en la cual en cada repetición se busca una trayectoria de aumento, y se incrementa el valor del flujo que circula en cada arco de dicha trayectoria por el valor de la capacidad residual de la misma (es decir, el flujo en esos arcos se incrementa en el menor valor de entre todas las capacidades de los arcos que constituyen la mencionada trayectoria), hasta que ya no se puedan encontrar trayectorias de aumento en la red. Entonces, por el teorema del flujo máximo – corte mínimo, una vez que ya no existan trayectorias de aumento el flujo máximo habrá sido calculado. [7]

3.1.2 Método de Edmonds - Karp

El método de Edmonds-Karp es idéntico al de Ford-Fulkerson, con la diferencia de que la búsqueda de trayectorias de aumento está definida de manera que la trayectoria de aumento a encontrar sea la más corta entre el origen y el destino, es decir, la trayectoria que tenga la menor cantidad de arcos [7]. De esta manera, se logra una mejora en el tiempo de ejecución del algoritmo. [26]

3.1.3 Método de Programación Lineal

La programación lineal es el campo de la optimización matemática dedicado a maximizar o minimizar (optimizar) una función lineal, denominada función objetivo, de tal forma que las variables de dicha función estén sujetas a una serie de restricciones expresadas mediante un sistema de ecuaciones o inecuaciones también lineales. El método tradicionalmente usado para resolver problemas de programación lineal es el Método Simplex. [9]

4 Estado del Arte

A continuación, se detalla el estado del arte actual del problema del flujo máximo, es decir, los métodos de solución existentes de dicho problema y algunas aplicaciones de software que permiten calcular la solución al mismo.

4.1 Aplicaciones similares existentes

A continuación, se describen algunas aplicaciones similares al software que se está desarrollando, ya existentes en el mercado global; para luego hacer un análisis comparativo entre las mismas.

4.1.1 Descripción de las aplicaciones

En esta apartado se presenta una lista de algunas de las aplicaciones encontradas que resuelven el problema del flujo máximo.

4.1.1.1 Grafos y Rutas

Grafos y Rutas forman parte de un proyecto de investigación y desarrollo de aplicaciones informáticas orientadas hacia la docencia, investigación y labores profesionales en ingeniería y optimización. [10] Desarrollado por un grupo de investigación en ingeniería y optimización a cargo Alejandro Rodríguez Villalobos, profesor de la Universidad Politécnica de Valencia. [11]

La filosofía de grafos básicamente consiste en "dibujar, modelar, resolver y analizar", por ello busca que el usuario tenga la absoluta libertad para tratar y abordar los problemas de grafos. Grafos permite dibujar libremente la red antes de preocuparse del problema que se va a resolver o el algoritmo que se necesitará para ello. Además, el software reconoce alguna condición no factible o algún requerimiento faltante para la solución de algún problema, notificando al usuario de este hecho. [12] Ejemplo: Apéndice 1 y Apéndice 2.

4.1.1.2 Graphing Calculator

Es un software comercial que permite analizar problemas matemáticos gráficamente. Ha sido desarrollado por la empresa PacificTec. [13]

Es un programa que permite visualizar objetos matemáticos en dos y tres dimensiones. Además, permite crear gráficos animados, resolver ecuaciones gráficamente y escoger la perspectiva de observación de los objetos dibujados. Adicionalmente, el software permite graficar funciones que estén dadas de manera explícita, implícita o parametrizada, tanto en dos o tres dimensiones. [14] Ejemplo: Apéndice 3.

4.1.1.3 Mathematica y Mapple

Mathematica es un software utilizado con fines científicos, de ingeniería, matemáticos y computacionales. Originalmente concebida por el científico inglés Stephen Wolfram, quien continúa liderando el grupo investigador dedicado al desarrollo de dicho software en su compañía Wolfram Research. [15]

Es un poderoso software que permite visualizar gráficas en dos y tres dimensiones, manejar matrices, solucionar sistemas de ecuaciones; además brinda herramientas para el procesamiento de imágenes, estadística multivariable, minería de datos entre otras. Además, ofrece un lenguaje de programación para desarrollar soluciones a problemas complejos. [16] Ejemplo: Apéndice 4.

Mapple por otro lado fue desarrollado originalmente por el Grupo de Cálculo Simbólico en la Universidad de Waterloo en Waterloo, Ontario, Canadá. Es un programa orientado a la resolución de problemas matemáticos, capaz de realizar cálculos simbólicos, algebraicos y de álgebra computacional. Se basa en un pequeño núcleo escrito en C, que proporciona el lenguaje Maple. Maple es un lenguaje de programación interpretado. Las expresiones simbólicas son almacenadas en memoria como grafos dirigidos sin ciclos. [17] Ejemplo: Apéndice 5.

4.1.1.4 Invop

Este software pretende integrar los temas asociados a los modelos de redes, tanto en la teoría como en la práctica. Desarrollado por la profesora Beatriz Loubet, docente de la Universidad Nacional de Cuyo en Argentina.

El software tiene propósito netamente académico, es decir la enseñanza de la investigación de operaciones. Dentro de esta área el programa maneja los problemas de asignación, transporte, recorridos de redes y flujos. Básicamente, se elige el tipo de problema que se desea resolver, luego se ingresa los datos de la red a través de una matriz de datos; para luego solicitar el cálculo del resultado deseado. [18]
Ejemplo: Apéndice 6.

4.1.1.5 WinQSB

Es un software interactivo de ayuda a la toma de decisiones que contiene herramientas muy útiles para resolver distintos tipos de problemas en el campo de la investigación operativa, con distintos módulos para tipo o modelo de problema dado. [19] Es propiedad intelectual del Dr. Yih-Long Chang, experto en investigación de operaciones y profesor del Georgia Tech en Estados Unidos.

Es un paquete de herramientas desarrollado para encontrar soluciones automatizadas a problemas complejos. El software incluye módulos para el análisis de muestreos, programación dinámica, elaboración de pronósticos, inventarios, procesos y cadenas de Markov, planificación de recursos, modelado de redes, programación no lineal, PERT, CPM, programación cuadrática, y otros problemas similares. [20] Ejemplo: Apéndice 7.

4.1.1.6 Lingo

Lingo es una completa herramienta diseñada para la construcción y resolución de modelos de optimización lineal, no lineal y entera de manera fácil rápida y eficiente. [21] Desarrollado por la empresa Lindo Systems Inc, la cual se ha caracterizado por proveer soluciones informáticas rápidas y fáciles de usar en el campo de la optimización matemática durante más de 21 años.

LINGO (Linear, Interactive and General Optimizer) es un lenguaje de modelación matemática que provee un entorno en el cual se puede desarrollar, ejecutar y modificar modelos matemáticos. [22] El resultado que LINGO proporciona es una optimización que ayuda a encontrar la mejor solución: la de mayor ganancia o la de menor costo, problemas que generalmente involucran el uso eficiente de recursos. [23]
Ejemplo: Apéndice 8.

4.2 Análisis comparativo entre aplicaciones

A continuación, se presenta un análisis comparativo entre las soluciones previamente presentadas, en la que se incluye el software a desarrollar en el proyecto.

Nro.	Descripción	Grafos y Rutas	Graphic Calculator	Mathematica	Invop	WinQSB	Lingo	Software Desarrollado
1	El software permite calcular el flujo máximo que puede circular en una red de flujo previamente dada con el uso del algoritmo de Ford-Fulkerson.	x			x	x		x
2	El software permite el ingreso de la red de flujo y sus características vía archivos de texto						x	x
3	El software permite el ingreso de la red de flujo y sus características gráficamente	x	x	x				
4	El software permite visualizar los resultados del algoritmo en archivos de texto						x	x
5	El software manejará una sintaxis específica para el ingreso de variables y ejecución de funciones y procedimientos			x			x	x
6	El software permite visualizar los resultados del algoritmo gráficamente	x	x	x				x
7	El software permite el procesamiento del problema como uno general de investigación operativa (con una función a optimizar y sus restricciones)		x				x	
8	El software permite solucionar los problemas con el método del simplex.		x	x			x	
9	El software puede ser utilizado con fines académicos de enseñanza de la investigación de operaciones y, en particular, del problema del flujo máximo	x			x	x	x	x
10	El software permitirá resolver problemas de flujo máximo con redes de más de 100 nodos		x	x			x	
11	El software permite resolver otros problemas de modelos de redes.	x	x	x	x	x	x	
12	El software permite resolver otros problemas de investigación operativa	x	x	x	x	x	x	
13	El software permite resolver problemas de otras áreas matemáticas además de la investigación operativa.		x	x			x	
14	El software permitirá el ingreso de los datos de la red por medio de matrices.				x	x		x
15	El software puede ser ejecutado en cualquier plataforma o sistema operativo, es decir, es portable.							x
16	El software es libre y sin costo	x			x	x		x

Tabla 1: Comparación de aplicaciones

5 METODOLOGÍA

Para la ejecución del presente proyecto se utiliza una metodología derivada del ciclo de vida clásico del desarrollo de software, según el cual, el trabajo se realiza linealmente desde la identificación de requisitos hasta el despliegue del software [2]. Se toma esta opción pues la cantidad de recursos empleados para el proyecto (un solo desarrollador) facilita en gran manera la planificación, coordinación e integración del software, así como de la documentación que lo sustenta. Además, se utilizan algunos de los principales modelos recomendados por RUP (Rational Unified Process), necesarios para el ordenado y adecuado desarrollo del producto, principalmente porque el tamaño del software hace innecesario el uso de toda la gama de procedimientos señalados por dicha metodología. Las etapas y actividades desarrolladas para el proyecto son detalladas a continuación:

Imagen 4: Diagrama WBS

5.1 Dependencias del Proyecto

ID	Nombre Tarea	Predecesora	Trabajo
1	**1. Iniciación**		10h
2	1.1 Reunión con el tutor		
3	1.2 Búsqueda de información y material complementario	2	
4	1.2 Realización y entrega de la documentación inicial	3	
5	**2. Planificación**	1	20h
6	2.1 Estudio de viabilidad	4	
7	2.2 Decidir las tareas del proyecto	6	
8	2.3 Asignar prioridades	6	
9	**3. Análisis y Diseño**	5	20h
10	3.1 Escoger la metodología del trabajo	8	
11	3.2 Diseño inicial del prototipo	10	
12	3.3 Pruebas iniciales	11	
13	**4. Desarrollo**	12	200h
14	4.1 Desarrollo del primer prototipo	12	
15	4.2 Ejecución del primer prototipo	14	
16	4.3 Control del primer prototipo	15	
17	4.4. Desarrollo del segundo prototipo	16	
18	4.5 Ejecución del segundo prototipo	17	
19	4.6 Control del segundo prototipo	18	
20	4.7 Desarrollo del tercer prototipo	19	
21	4.8 Ejecución del tercer prototipo	20	
22	4.9 Control del tercer prototipo	21	
23	**5. Resultados**	13	30h
24	5.1 Realización de la documentación	22	
25	5.2 Entrega del documento final y manual usuario	24	
26	5.3 Presentación y defensa del proyecto	24	
27	**6. Finalización**	23	20h
28	6.1 Cierre del trabajo	26	

Total, horas de trabajo: 300 h

Tabla 2: Cronograma de actividades del proyecto

5.2 Requerimientos

A continuación, se presentan los requisitos que debe cumplir el software a implementar en el presente proyecto.

5.2.1 Requerimientos funcionales

No.	Descripción	Prioridad
1	El software a desarrollar permitirá el ingreso de la red en la cual se desea resolver el problema del flujo máximo.	1
2	El software a desarrollar permitirá almacenar la red ingresada y sus características para futuras visualizaciones.	1
3	El software a desarrollar permitirá cargar redes existentes previamente guardadas y poder modificarlas o calcular el flujo máximo que circula por él.	1
4	El software a desarrollar permitirá comprobar la validez del archivo que contiene las características de la red (formato TXT), así como de sus nodos y arcos, a fin de verificar el cumplimiento de las condiciones necesarias para la aplicación del método de solución elegido.	1
5	El software permitirá ingresar redes con más de un origen o más de un destino y resolverlos sin complicación adicional.	1
6	El software a desarrollar permitirá calcular el flujo máximo que puede circular por la red ingresada, así como el flujo sobre cada uno de los arcos de la misma con el uso del algoritmo de Ford – Fulkerson.	1
7	El software a desarrollar permitirá mostrar y almacenar los resultados de la ejecución del algoritmo en archivos de texto.	1

Tabla 3: Requerimientos Funcionales

5.2.2 Requerimientos no funcionales

No.	Descripción	Tipo	Prioridad
1	Se utilizará lenguaje Java	Implementación	1
2	Los resultados de la ejecución del algoritmo de cálculo de caudales podrán almacenarse en archivos de texto	Implementación	1
3	La aplicación podrá funcionar sobre sistemas operativos Windows, Linux o MacOS, con lo que se aprovechará la portabilidad ofrecida por Java.	Implementación	1

Tabla 4: Requerimientos no funcionales

5.2.3 Prioridades

Número	Descripción
1	Alta
2	Media
3	Baja

Tabla 5: Prioridades

6 Resultados

A continuación, se presenta una pequeña explicación del programa desarrollado y de los resultados que se han ido obteniendo.

Imagen 5: Menú principal

6.1 Interfaz

El programa presenta una sencilla e intuitiva interfaz, la cual en un primer plano nos proporcionara dos opciones para poder cargar un problema en memoria: Introduciendo los datos de forma manual o abriendo un archivo

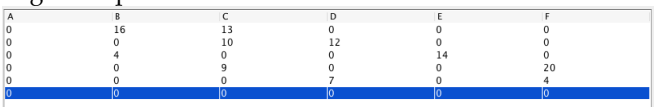
TXT previamente guardado con todos los datos necesarios para su cálculo.

6.2 Carga Manual

Centrándonos en la primera opción nos llevará a una segunda pantalla donde pedirá que introduzcamos el número de nodos que se desea, en este caso se hizo una prueba con 6 nodos.

Imagen 6: Casilla nodos

Al darle a continuación al botón de "Construir", la app nos proporciona una tabla en la cual se introducirán los datos del grafo. Estos datos son las capacidades de los arcos que unen los nodos, que deberán ser siempre números mayores o iguales que cero



A	B	C	D	E	F
0	16	13	0	0	0
0	0	10	12	0	0
0	4	0	0	14	0
0	0	9	0	0	20
0	0	0	7	0	4
0	0	0	0	0	0

Imagen 7: Matriz manual del grafo

Una vez que se han terminado de introducir los datos en la tabla, en la parte inferior nos pedirá la fuente que en este ejemplo se le dará el valor de 1 y el flujo que será 6.

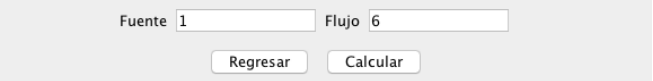
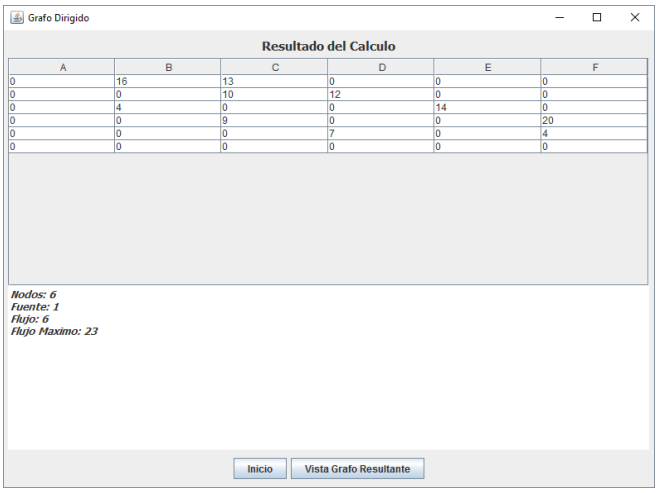


Imagen 8: Casillas fuente y flujo

Con todos los datos regulados le daremos al botón de calcular y con esto obtener el flujo máximo de este grafo que será 23 como se visualiza en la imagen.



A	B	C	D	E	F
0	16	13	0	0	0
0	0	10	12	0	0
0	4	0	0	14	0
0	0	9	0	0	20
0	0	0	7	0	4
0	0	0	0	0	0

Imagen 9: Pantalla del resultado del grafo

En programa nos dará también la opción de poder visualizar el problema en modo grafo dándole al botón de "Vista Grafo Resultante".

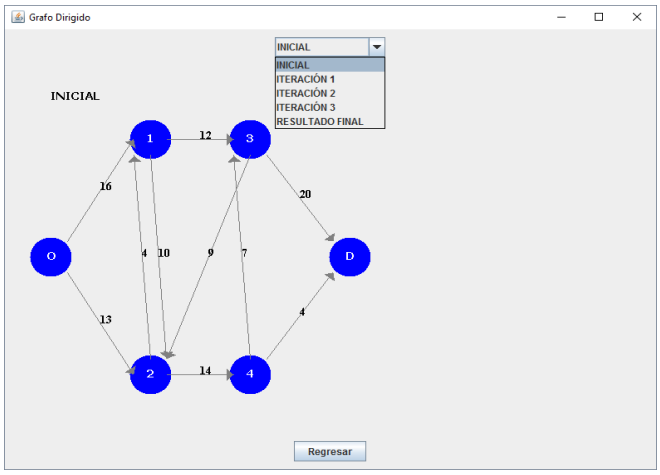


Imagen 10: Pantalla resultado inicial

Como observamos, se tiene a disposición una casilla desplegable que nos muestra el número de iteraciones que ha tenido que hacer el algoritmo para poder llegar al resultado final del flujo máximo. Al seleccionar cualquiera de las iteraciones, el programa nos marcara en "rojo" el camino escogido, las siguientes imágenes mostraran el grafo de las iteraciones 1 y 2.

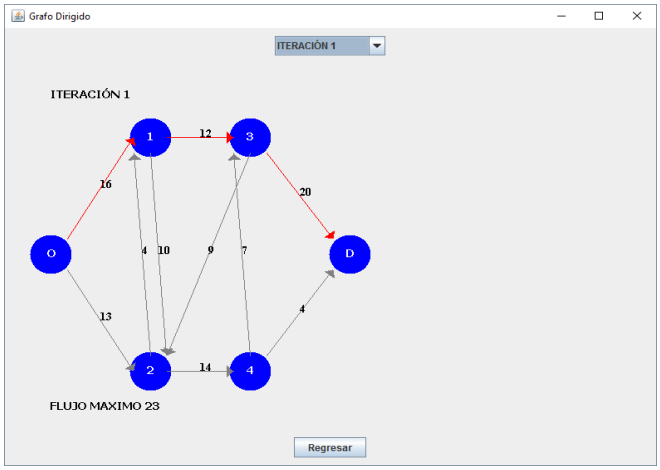


Imagen 11: Pantalla resultado iteración 1

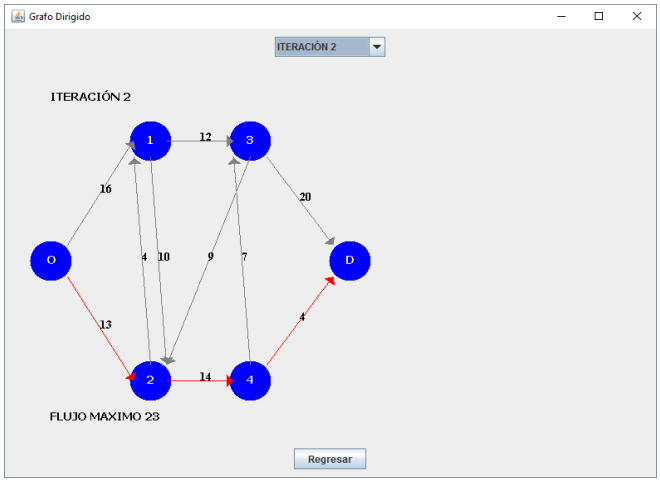


Imagen 12: Pantalla resultado iteración 2

6.3 Carga fichero TXT

Pasando ahora a la opción de cargar los datos del grafo mediante un fichero, en la pantalla de inicio al seleccionar la opción TXT no saldrá la siguiente pantalla.

Imagen 13: Pantalla de carga fichero TXT

Donde nos dará unas pautas a seguir para la correcta carga de los datos del grafo del fichero. El fichero datosEjemplosGrafo.txt será nuestro ejemplo a realizar, vemos en su interior el formato que deberá cumplir.

```
NODOS:6
FUENTE:1
FLUJO:6
0;16;13;0;0;0
0;0;10;12;0;0
0;4;0;0;14;0
0;0;9;0;0;20
0;0;0;7;0;4
0;0;0;0;0;0
```

Imagen 14: Contenido fichero TXT

Una vez creado y comprobado nuestro fichero le daremos al botón de "Buscar TXT" y seleccionaremos el fichero de ejemplo.

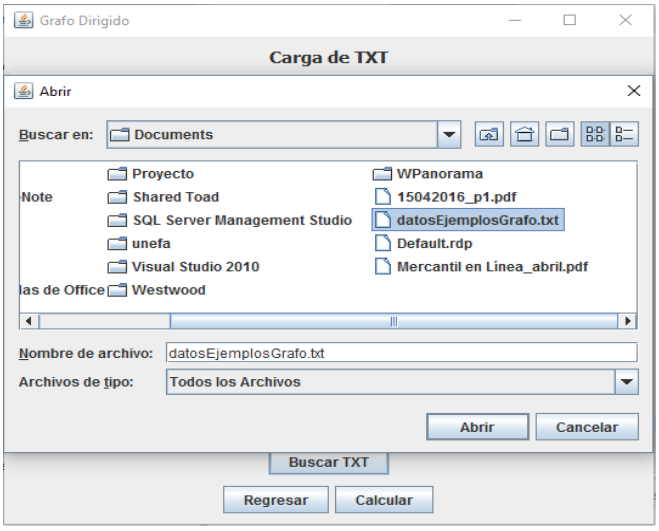


Imagen 15: Pantalla selección/carga fichero TXT

Le daremos a "Abrir" y al momento nos cargará los datos del fichero y calculará el resultado correspondiente del problema, que en este caso nos dará un flujo de 23.

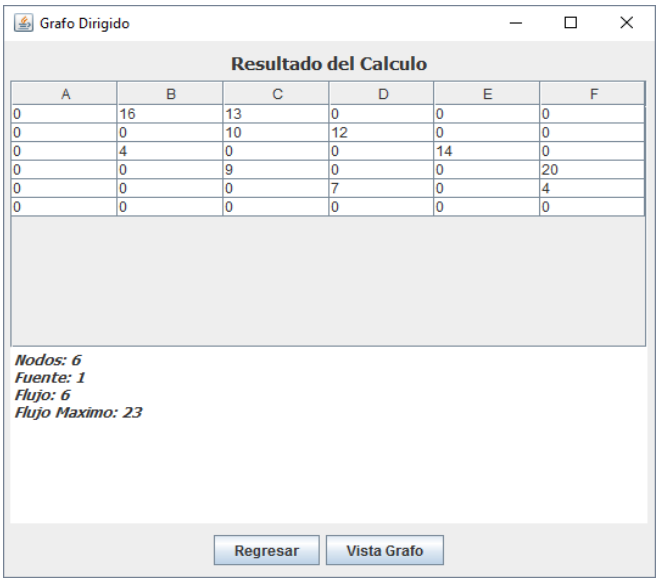


Imagen 16: Pantalla resultante de la carga del fichero TXT

Los pasos en este punto, serán los mismos que en el apartado anterior para poder visualizar el problema en vista de grafo y revisar las iteraciones realizadas para su solución.

7 Conclusiones

El software implementado puede ser utilizado tanto en la enseñanza de la investigación de operaciones como en aplicaciones prácticas sobre el problema del flujo máximo. Dada la cantidad de información e iteraciones que se utilizan en el problema del flujo máximo, y en los problemas de investigación de operaciones en general, se hace casi imposible la solución de dichos problemas manualmente, por lo cual se hace necesario que estos sean resueltos a través de programas computacionales. Asimismo, dada la cantidad de datos que puede contener una red, ya sea para el problema del flujo máximo o para otros problemas de la investigación de operaciones, podría darse el caso que los métodos que calculen la solución exacta no encuentren resultado por la cantidad de recursos que requerirían, por lo que podría ser necesario que se requieran nuevos algoritmos que, si bien no obtengan la solución exacta, encuentren soluciones aproximadas suficientemente adecuadas.

7.1 Inconvenientes

La mayor dificultad con la he me encontrado durante el proyecto ha sido la complejidad en el desarrollo de la aplicación, ya que en un principio pensé en utilizar el lenguaje C++. Lo cual lo vi muy factible en ese momento para el desarrollo del aplicativo. Pero me encontré con la problemática de que, con C++, estaba muy limitado a un código poco dinámico y moldeable, podía escribir programas orientados a objetos, pero no orientados a objetos o mezcla de ambos (por ejemplo, el poder tener clases con funciones globales en el mismo programa). Esto, de cara al futuro, me

generó un problema grave a la hora de modificar cosas de la aplicación, como cambios inesperados de último momento, poder arreglar algún problema pequeño o grave sin tener que tocar todo el programa en su totalidad o incluso algo tan esencial como tener un código visualmente presentable y no líneas eternas de código que yo mismo llegue a perderme entre ellas.

Por eso que me replantee el lenguaje a utilizar para el desarrollo y optar al lenguaje en Java. Ya que este era puramente orientado a objetos, me ayudo a que cualquier función pueda pertenecer a alguna clase (un método), la facilidad de proporcionar de que todos los objetos puedan heredar automáticamente de una clase en concreto, lo cual me ahorraba muchas líneas de código a implementar que incluso podrían ser repetitivas en el C++. También dejaba de lado el uso de punteros, ya que en Java me facilitaba el poder acceder a los objetos mediante referencias, esto hacia que mi trabajo fuera moldeable a la hora de implementar nuevas ideas en la aplicación y no tuviera que preocuparme en cambiar algo importante o corromper el código ya creado. [24]

Aunque, este cambio me genero un retraso importante en el trabajo ya realizado en la interfaz de la aplicación, ya que tuve que pasar el código C++ al formato de Java. Esto me llevo su tiempo adaptarlo sin mencionar también que siempre surgía algún nuevo error por el camino. Aparte, a comparación de otras personas que se les puede dar muy bien la programación y encontrar errores al momento, yo por otra parte tenía que tomarme mi tiempo e investigar posibles soluciones al detalle. Por lo que iba desarrollando el proyecto según lo planeado, pero no pude llegar a implementar todos los puntos extras que se deseaba por ir contra el tiempo encima y problemas técnicos con las herramientas de trabajo en las últimas semanas. Así que las distintas variantes del algoritmo para la resolución del problema del flujo máximo se dejan en un paréntesis abierto para su continuación en el desarrollo y finalización.

7.3 Complejidad Algorítmica

En este apartado observaremos algunas de las variantes que han ido surgiendo del algoritmo y como fue mejorando su complejidad con el paso de los años.

Año	Autor	Complejidad
1955	Ford-Fulkerson	$\mathcal{O}(mnU)$
1970	Dinic	$\mathcal{O}(mn^2)$
1970	Edmonds-Karp	$\mathcal{O}(m^2n)$
1972	Dinic, Edmons-Karp	$\mathcal{O}(m^2 \log U)$
1973	Dinic, Gabow	$\mathcal{O}(mn \log U)$
1974	Karzanov	$\mathcal{O}(n^3)$
1977	Cherkassky	$\mathcal{O}(n^2m^{1/2})$
1980	Galil-Naamad	$\mathcal{O}(mn(\log n)^2)$
1983	Sleator-Tarjan	$\mathcal{O}(mn \log n)$
1986	Goldberg - Tarjan	$\mathcal{O}(mn \log(n^2/m))$
1987	Ahuja-Orlin	$\mathcal{O}(mn + n^2 log U)$
1987	Ahuja-Orlin-Tarjan	$\mathcal{O}(mn \log(2 + n\sqrt{\log U/m}))$
1990	Cheriyan-Hagerup-Mehlhorn	$\mathcal{O}(n^3/\log n)$
1990	Alon	$\mathcal{O}(mn + n^{\tilde{3}} \log n)$

Tabla 6: Evolución histórica de la complejidad algorítmica (U: Capacidad máxima de los arcos)

7.4 Recomendaciones y trabajos futuros

En el futuro se podría personalizar el software para implementar alguna de las aplicaciones particulares que se modelen a través del problema del flujo máximo. Asimismo, se podrían implementar nuevos algoritmos de solución a fin de compararlos y determinar los casos en los cuales cada uno de ellos es el más adecuado. Adicionalmente, se podrían investigar algoritmos de inteligencia artificial que resuelven dicho problema con una mayor sofisticación. Finalmente, se podrían implementar maneras alternativas de ingresar y visualizar las redes, como por ejemplo interfaces gráficas.

BIBLIOGRAFÍA

[1] **Winston, Wayne L.** Investigación de operaciones: aplicaciones y algoritmos. 4ta. México D.F: Thompson Learning, 2005.

[2] **Pressman, Roger S**. Ingeniería del Software. Un enfoque práctico. 6ta. México D.F.: Mc Graw Hill, 2005.

[3] Campus Virtual UAB, Aplicació de seguiment dels TFG - 1r Semestre. Bellatera, Barcelona, España: s.n, 14 de septiembre 2015. http://aules.uab.cat/2016-17/course/view.php?id=385

[4] "Informació General.pdf", Aplicació de seguiment dels TFG - 1r Semestre. Bellatera, Barcelona, España: s.n, 14 de septiembre. https://tfe.uab.cat/tfe/espai/show/50935

[5] "Avaluació.pdf", Aplicació de seguiment dels TFG - 1r Semestre. Bellatera, Barcelona, España: s.n, 14 de septiembre. https://tfe.uab.cat/tfe/espai/show/50935

[6] **Hudson, Ian.** The Maximum Flow Problem. Granville, Ohio, Estados Unidos: s.n., 1 de mayo de 2004.

[7] **Cormen, Thomas H.** Introduction to algorithms. 2da. Boston: McGraw-Hill, 2001.

[8] "MDCap4.pdf", Diapositivas Matemáticas Discretas, Capítol 4: Xarxes de Transport, Bellatera UAB, Barcelona, España: s.n, 2011-2012

[9] Wikipedia, "Programación Lineal", España: s.n, 11 de septiembre 2016. https://es.wikipedia.org/wiki/Programaci%C3%B3n_lineal#Variables

[10] **Rodríguez Villalobos, Alejandro.** Grafos. Universidad Politécnica de Valencia. [En línea] [Citado el: 14 de diciembre de 2009.] http://personales.upv.es/arodrigu/grafos/FordFulkerson.htm

[11] **Alejandro Rodríguez Villalobos.** Universidad Politécnica de Valencia. [En línea] [Citado el: 2009 de diciembre de 14.] http://personales.upv.es/arodrigu/Inicio.html.

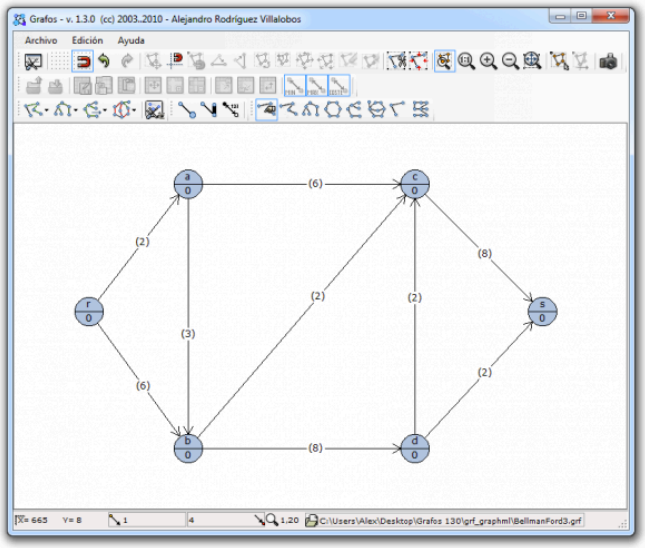
[12] ¿Qué es grafos? Universidad Politécnica de Valencia. [En **línea] [Citado el: 14 de diciembre de 2009.] http://personales**.upv.es/arodrigu/grafos/index.htm.

[13] **Pacific Tech.** Graphing Calculator. Pacific Tech. [En línea] [Citado el: 16 de diciembre de 2009.] http://www.pacifict.com/.
[14] **Ávila Herrera, Juan Félix.** Graficación animada. Pacific Tech. [En línea] [Citado el: 16 de diciembre de 2009.] http://www.pacifict.com/Usando-GraphingCalculator.pdf.
[15] **Wolfram Research.** Wolfram Mathematica 7. Wolfram Research. [En línea] [Citado el: 16 de diciembre de 2009.] http://www.wolfram.com/products/mathematica/index.html.
[16] History and background. Wolfram Mathematica. [En línea] [Citado el: 16 de diciembre de 2009.] http://www.wolfram.com/products/mathematica/history.html.
[17] Wikipedia, "Maple (Software)", España: s.n, 15 de noviembre 2016. https://es.wikipedia.org/wiki/Maple_(software)
[18] **Loubet, Beatriz.** Ayuda Invop. Mendoza, Argentina: s.n., 1998.
[19] **Universidad de Valencia.** Universidad de Valencia. Universidad de Valencia. [En línea] [Citado el: 14 de abril de 2010.] http://www.uv.es/martinek/material/WinQSB2.0.pdf.
[20] **Bello, Juan, y otros.** Universidad Santa María, Caracas - Venezuela. Investigación de operaciones. [En línea] octubre de 2004. [Citado el: 21 de abril de 2010.] http://www.investigacion-operaciones.com/Software/Manual%20WinQSB.pdf.
[21] **Lindo Systems Inc.** Lindo Systems Products. Sitio web de Lindo Systems. [En línea] 15 de enero de 2010. [Citado el: 2010 de abril de 2010.] http://www.lindo.com/index.php?option=com_content&view=article&id=2&Itemid=10.
[22] **J., Ponce, G., Solis y L., Ulfe.** Guía básica de LINGO. Investigación de operaciones S.A. [En línea] [Citado el: 21 de abril de 2010.] http://www.iosa.com.pe/descargas/Programacion%20Lineal/GuiaLINGO%20(Intro%20y%20modelos).pdf.
[23] **Canizo, Erica y Lucero, Paola.** Universidad Tecnológica Nacional. Facultad Regional Mendoza. [En línea] 2002. [Citado el: 2010 de abril de 21.] http://www.frm.utn.edu.ar/iopera-tiva/lingo_lindo.pdf.
[24] **H.M. Deitel, P.J. Deitel.** Java, how to program. 5th Edition. s.l.: Pearson Education, 2003. págs. 1-18.
[25] **Ford, L. R.; Fulkerson, D. R**. *(1956).* "Maximal flow through a network". Canadian Journal of Mathematics.
[26] **Jack Edmonds and Richard M. Karp** (1972). «Theoretical improvements in algorithmic efficiency for network flow problems» Journal of the ACM 19 (2): 248-264.

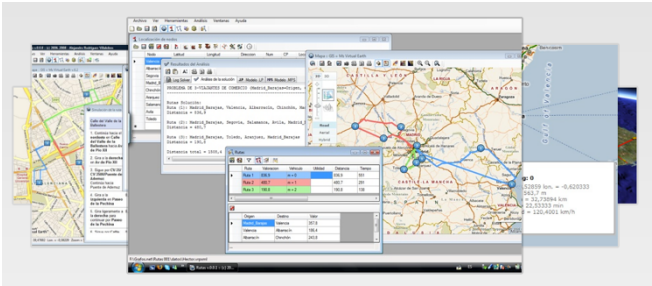
APENDICE

A1. EJEMPLO DE GRAFOS Y RUTAS

.

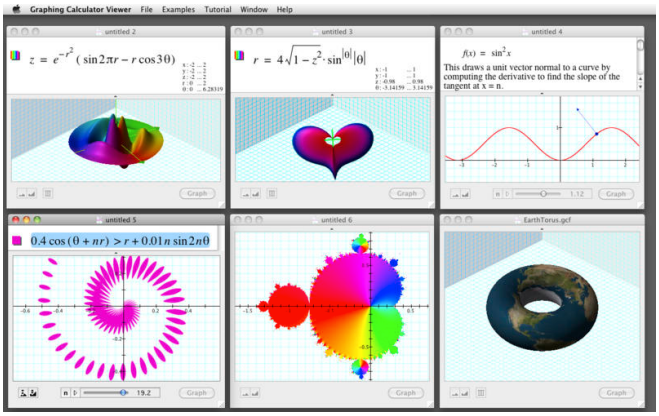


Apéndice 1: Captura de la app de Grafos



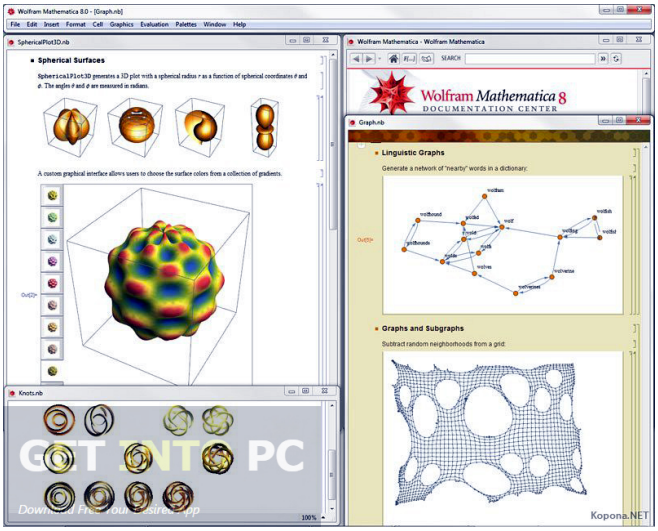
Apéndice 2: Captura de las funcionalidades de Rutas

A2. EJEMPLO DE GRAPHING CALCULATOR

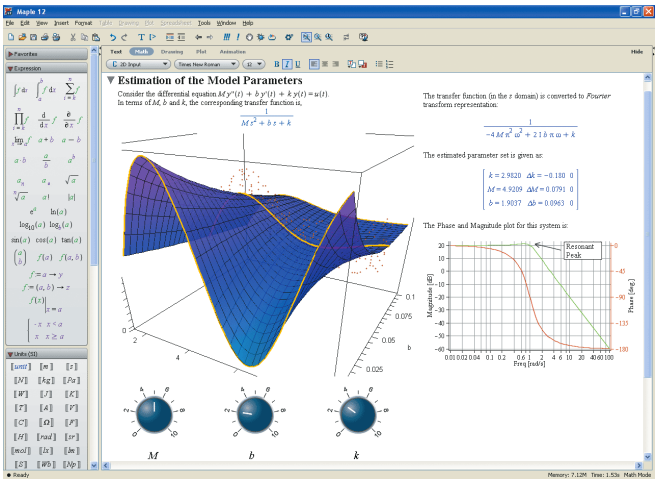


Apéndice 3: Captura de las funcionalidades de Graphing Calculator para MAC

A3. Ejemplo de Mathematica y Mapple

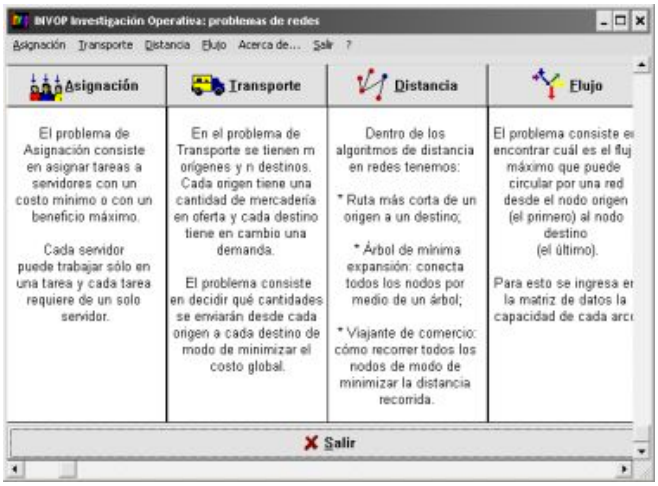


Apéndice 4: Captura de las funcionalidades de Mathematica



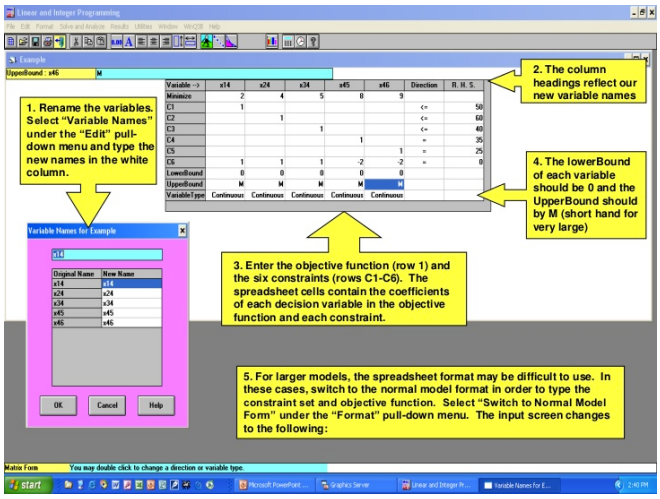
Apéndice 5: Captura de las funcionalidades de Maple

A4. Ejemplo de Invop



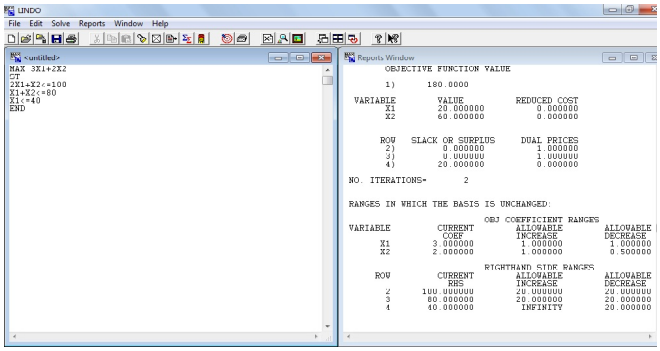
Apéndice 6: Captura de las funcionalidades de Invop

A5. Ejemplo de WinQSB



Apéndice 7: Captura de las funcionalidades de WinQSB

A6. Ejemplo de Lingo



Apéndice 8: Captura de las funcionalidades de Lingo