Universidad Nacional Autónoma de México

FACULTAD DE INGENIERIA

SISTEMA DE METODOS DE SOLUCION DE REDES DE FLUJO (IO)

T E S I S

PARA OBTENER EL TITULO DE INGENIERO EN COMPUTACION

P R E S E N T A

BRISIA TOMASA JON SERRANO

Director de Tesis:
DR. SERGIO FUENTES MAYA

MEXICO, D. F. 1991

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE

PROLOGO

Desde la aparición del hombre sobre la Tierra, se ha tenido la necesidad de realizar diversos tipos de cálculo, lo que propició la invención de procedimientos e instrumentos. A medida que éstos evolucionaban, permitieron efectuar dichos cálculos, cada vez con mayor eficacia y rapidez.

La tecnología está siempre en busca de mejores soluciones para satisfacer las necesidades actuales, logrando avances importantes, entre ellos, lograr que la información llegue en menos tiempo a su destino. Hoy en día la información toma un papel importante en el desarrollo de cualquier empresa o institución, sobre todo para aquellas que manejan grandes volumenes de información, ya que de esto depende la planificación de diversas actividades a realizar en determinado período de tiempo y a planificar decisiones futuras.

La computadora es el instrumento más complejo que ha inventado el hombre y por muchas décadas permanecerá como el más potente. Nuestras vidas se ven constantemente afectadas por las aplicaciones cada vez mayores de las computadoras. Hoy en día, los instrumentos de cálculo poseen un alto grado de sofisticación, gracias al empleo de las técnicas electrónicas, las cuales permiten que dichos instrumentos también posean una gran rapidez y precisión. La aplicación de instrumentos como la computadora digital abarca casi todos los campos de la actividad humana, ya sean estas científicas, tecnológicas, administrativas o sociales.

En el futuro, servicios utilitarios de computadora serán tan corrientes como los servicios telefónicos y energía eléctrica del presente. En la ingeniería, la computadora ha llegado a ser una herramienta indispensable, la cual obliga al profesional a un conocimiento tan amplio en este campo como sea posible.

La presente tesis es el desarrollo de un SISTEMA DE METODOS DE SOLUCION DE PROBLEMAS DE REDES DE FLUJO (IM), el cual permitirá dar solución a los problemas aplicados a redes de flujo de una forma más eficiente y óptima, de este modo se realizarán más rapidamente lo que beneficiará en tiempo y costo su aplicación.

Este proyecto surge como una necesidad por parte de las personas y empresas que se encuentran diariamente con este tipo de problemas y para los cuales requieren soluciones rápidas y confiables, las soluciones se pueden obtener por medio de los métodos contenidos en este sistema.

El sistema está diseñado para ser utilizado en microcomoputadoras PC y Sistemas Personales. Este sistema está diseñado en un lenguaje que facilite la operación en las redes de flujo.

Este trabajo se ha dividido en fases, las cuales permiten llevar a cabo un control efectivo del sistema durante las etapas de su desarrollo; cada una de estas se desarrolla en los capítulos que a continuación se presentan.

En el capítulo uno se trata el tema de Introducción, el cual nos muestra la justificación del presente trabajo y el objetivo que tiene el sistema realizado.

En el capítulo dos se hace referencia a los conceptos necesarios para la realización del sistema; esto se realiza con el fin de que los usuarios que accesen a este sistema tengan un panorama general sobre las estructuras de datos y su terminología básica.

En el capítulo tres se definirán los métodos de solución que integrarán el sistema, se da una visión general de cada uno de ellos con sus respectivos ejemplos para que el lector tenga un conocimiento sobre los métodos que puede aplicar a sus problemas y poder escoger el más adecuado al tipo de problema que posee.

En el capítulo cuatro se hace referencia al sistema de software, esto es, análisis, diseño , desarrollo e implementación y pruebas del sistema.

En el capítulo cinco se presentarán las conclusiones logradas durante el desarrollo de este sistema.

Se presentarán los apéndices A y B relacionados a este sistema, el Apéndice A es un resumen de teoría de Redes y el Apéndice B es un resumen de Elementos del Análisis y Diseño Estructurado.

CAPITULO

UNO

INTRODUCCION

I.1 INTRODUCCION

La información es un proceso de intercambio. El proceso de recibir y utilizar información consiste en ajustarnos a las contingencias de nuestro medio y de vivir de manera efectiva dentro de él. Las necesidades y la complejidad de la vida moderna plantean este fenómeno del intercambio de información como demandas más intensas que en cualquier otra época; la prensa, los museos los laboratorios científicos, las universidades, las bibliotecas y los libros de texto han de satisfacerlas. El lugar que ocupa en la actualidad el análisis de datos no es trivial, ni fortuito, ni nuevo, por lo que se ha generado un avance paralelo en la computación para satisfacer las necesidades que nos implica la diaria existencia.

Se puede decir que la década de los sesenta en el campo del "software" fue la época de los lenguajes, ya que en ella se difundió el *Fortran* (diseñado en 1954), y se desarrollaron otros lenguajes considerados fundamentales como *Algol 60* y *68*, *Cobol*, *PL/1*, etc. En la década de los setentas hubo un cambio, se desarrolla la programación estructurada, que encabezado por E.W. Dijkstra puso de relieve la dificultad que la creciente complejidad de los lenguajes representaba para la construcción de los programas, proponiendo como nuevo criterio de valoración de un lenguaje sus posibilidades para crear, a partir de sus componentes básicos programas fiables e inteligibles.

Con base a estas ideas, se constituyó el diseño del lenguaje PASCAL, realizado por *Niklaus Wirth* (1971), el cual también participó con su metodología de refinamiento gradual para la construcción de programas.

Por otra parte, la extensa actividad conocida por investigación de Operaciones ha tenido un avance también explosivo, debido principalmente porque la gente de esta área ha estado trabajando en diversas circunstancias.

Recuérdese el inicio de la Investigación de Operaciones, lo que aportaron los científicos a los problemas operativos fue su aspecto científico. Y de hecho, esta fue su contribución principal. Aportaron ideas nuevas, desconfiaron de los conceptos preconcebidos y obraron sólo ante la evidencia. En la actualidad, los nuevos problemas no son tan accesibles como lo fueron muchas de las operaciones militares.

El objetivo general de este trabajo es sintetizar y describir los conceptos fundamentales de la estructura de datos y su aplicación a los métodos de solución de redes de flujo. Otro aspecto de este trabajo es la implantación de algunos algoritmos típicos para la solución de problemas modelados como redes.

Este trabajo pretende unificar las ideas trascendentales que existen sobre el tema de algoritmos, sus principios fundamentales,haciendo hincapié en los conceptos de diseño de algoritmos para que sean más fácilmente pensados. Así, el decidir es propio del hombre, ya que la decisión implica la selección conciente entre varias soluciones posibles.

1.2 JUSTIFICACION:

La computadora representa, de alguna manera, una herramient capaz de cumplir los deseos de rapidez y eficiencia, en l trabajos y tareas de cálculo en diversas áreas y la organización una gran cantidad datos.

Se puede hablar de computadoras analógicas y digitales; las computadoras digitales son aquellas que manejan la información de manera discreta (por medio de bits), y las analógicas trabajan por medio de funciones continuas (generalmente representación de señales eléctricas).

La computadora digital moderna fue inventada e ideada como un dispositivo que debe facilitar y acelerar operaciones de cálculo complicadas y que consumen mucho tiempo en su elaboración. En la mayoría de las aplicaciones su capacidad de almacenar y permitir la entrada a grandes cantidades de información desempeña la parte dominante y se considera como su caracerística principal.

Recientemente las ciencias de la computación han cobrado gran importancia debido a los avances científicos y tecnológicos cuyos resultados han permitido su utilización en áreas tales como: la industria, la economía, la administración, la educación, la salud, la medicina, la ingeniería y otras actividades que realiza el ser humano día a día.

En todos estos casos, la enorme cantidad de información que se procesa en algún sentido representa una parte de la realidad. La información de que se dispone para llevar a cabo algún proceso en la computadora consta de un conjunto determinado de datos acerca del problema real a resolver.

Los datos representan una cierta parte de la realidad en el sentido de que ciertas propiedades y características de los objetos reales son ignorados por ser periféricos e irrelevantes para el problema específico.

Al resolver un problema con o sin computadora, se necesita elegir y definir un conjunto de datos que representará la situación real. Esta selección debe ser guiada por el problema que debe resolverse. Luego sigue una elección de la representación de esta información. Esta elección es guiada por la herramienta que se usa para resolver el problema. Al resolver el problema con una computadora, la herramienta que se utiliza son los recursos que ofrece la computadora.

Para realizar la tarea en una computadora es necesario realizar una definición cuidadosa de dos tipos de objetos: los *datos* y las operaciones o funciones que actuen sobre ellos. Puede existir algún problema en el momento de indagar como se pueden describir los pasos a efectuar y el modo de como decírselo a la computadora. De ello surgió la necesidad de tener un programa para decirle a la computadora como deberá efectuar los pasos para realizar la tarea.

El problema concreto es escribir un programa que haga lo que se desea en la computadora.

La elección de la representación con los *datos*, con frecuencia es considerablemente difícil y no se determina solo por los recursos de que se dispone. Siempre debe tomarse de acuerdo a las operaciones que se realicen con los datos.

Generalmente se sabe que las computadoras utilizan una representación interna basada en dígitos binarios(bits). Esta representación es inadecuada para los seres humanos debido al gran número de cifras comprendidas, pero es más adecuada para circuitos electrónicos porque los valores 0 y 1 pueden representarse en forma ventajosa y confiable por la presencia o ausencia de corrientes eléctricas, carga eléctrica o campos magnéticos.

Un lenguaje de programación nos permite comunicarnos con la computadora para decirle como debe realizar las tareas. Por supuesto, una computadora real representa todos los datos, serie de números, conjuntos o secuencias como una masa grande de bits. Pero esto es irrelevante para el programador en tanto no tenga que preocuparse por los detalles de la representación de las abstracciones elegidas y en tanto pueda confiar en que la representación correspondiente, elegida por la computadora (o compilador) es razonable para los fines propuestos.

La información es un proceso de intercambio de datos. Este proceso de utilización de información nos ayuda a realizar de una forma más efectiva el trabajo en el área en que nos desarrollamos. Actualmente, los sistemas de información ocupan un lugar importante en nuestra vida diaria, ya que satisfacen las necesidades y requerimientos del manejo de información y solución de las tareas que tenemos día con día.

La *Investigación de Operaciones* ha tenido un gran desarrollo, ya que desde sus inicios tendía a optimizar diferentes actividades. La máquina cibernética ha sido un pilar en la evolución de las técnicas de investigación de operaciones, tanto por su potencial de cálculo como por su rapidez y exactitud.

Puede expresarse ciertamente que hoy en día es imprescindible el uso de la computadora para realizar procesos de investigación de operaciones en tareas que aunque parezcan sencillas, involucran una gran cantidad de datos, que dificultarían el resolverlos en forma manual.

La teoría de redes es una clase de modelos matemáticos que involucra la representación gráfica de ciertos problemas de optimización. Las redes tienen una aplicación extensa en diversos campos como son: planeación, administración, ingeniería, química, educación, etc.; en estos campos innumerables situaciones pueden formularse como modelos matemáticos de redes. Algunos ejemplos son: sistemas de producción-distribución, tráfico urbano, transporte colectivo, comunicación, redes eléctricas, reemplazo de equipo, inventarios, presas, flujo de dinero, tuberías, oleoductos, asignación de recursos y otros más.

Este trabajo nos muestra la aplicación de la computación en algunas tareas de investigación de operaciones. Se desarrollará la aplicación de las *estructuras de datos* (computación) en la solución a problemas de *redes de flujo* (investigación de operaciones), para resolver de manera óptima y eficaz este tipo de problemas.

Se implantarán algunos algoritmos fundamentales dentro del área de las *redes de flujo*, para la solución de tareas como: Flujo máximo, árbol de expansión mínima, ruta más corta, búsqueda en anchura, búsqueda en profundidad, conectividad y algunas otras más.

De este modo se pretende realizar la unificación de los aspectos más fundamentales de *redes de flujo* en un programa integrado, para una optimización de la solución de los problemas que se encuentren en ésta área.

1.3 OBJETIVO

La tecnología está siempre en busca de mejores soluciones para satisfacer las necesidades actuales, logrando avances importantes, entre ellos, lograr que la información llegue en menos tiempo a su destino.

Hoy en día la información toma un papel importante en el desarrollo de cualquier institución y más aún para aquellas que manejan grandes volúmenes de información, ya que de esto depende la planificación de diversas actividades a realizar en determinado periodo de tiempo y a planificar decisiones futuras.

Debido al campo que se aborda, se hablará de los avances logrados en comunicación de datos y en los diferentes procesos de información por medio de la computadora que ha crecido considerablemente.

Como la teoría de redes es una clase de modelos matemáticos que involucra la representación gráfica de ciertos problemas de optimización, las redes tienen una aplicación extensa en diversos campos como: administración, química, educación, etc ; debido a su extensa aplicación en diversos campos la representación gráfica de las redes se puede lograr por medio de la ayuda de la computadora, lo cual ayudaría mucho al realizar esta tarea de una forma más rápida.

Gracias a ésto, y a la estructura especial que presentan los modelos de redes, se han desarrollado algoritmos eficientes para la solución de los problemas formulados en estos diversos campos. Incluso se ha desarrollado más de un algoritmo para resolver el mismo tipo de problema en base a las restricciones que en él se consideran.

Además, los algoritmos son, en su mayoría, de relativa facilidad de comprensión y aplicación, de acuerdo a las necesidades y al campo en que se desarrollarán, ya que surgen de manera natural en el desarrollo de la teoría. Otra ventaja de algunos algoritmos es que, a través de ellos, es posible detectar cuando un problema de redes no tiene solución.

En la teoría de redes existe un conjunto bien definido de problemas básicos como: ruta más corta, flujo máximo, flujo a costo mínimo, entre otros. Se conocen con el nombre de problemas básicos ya que otros problemas pueden ser formulados en términos de éstos problemas. Por ejemplo, el problema de transporte a costo mínimo puede ser formulado como un problema de flujo a costo mínimo, el problema de acoplamiento de cardinalidad máxima puede formularse como uno de flujo máximo, algunos problemas de reemplazo de equipo pueden formularse como uno de ruta más corta, y otros más.

El objetivo de este trabajo es la aplicación de la computación a la teoría de redes para resolver el tipo de problemas antes mencionados de manera óptima y eficaz.

La aplicación de la computación para este tipo de problemas se realizará por medio de las estructuras de datos, ya que por medio de estas se pueden resolver los problemas de gráficas (la teoría de

redes involucra la representación gráfica de los problemas), en este caso redes, de una manera más eficientes.

La aplicación de las estructuras de datos a estos de problemas hará que la solución de estos sea lo más eficiente y óptimo posible de acuerdo al tipo de problemas a resolver.

Se implantarán los algoritmos llamados básicos y otros más dentro del área de redes. Está implantación se hará por medio de un sistema de software que unificará estos algoritmos para la solución de varios tipos de problemas en un mismo sistema, o la solución de un problema a través de varios algoritmos que lo puedan solucionar, para seleccionar la mejor solución.

1.4 METODOLOGIA DE TRABAJO.

Debido al gran avance progresivo de la computación, y en particular en el área del software, el desarrollo de programas es de gran importancia, ya que existen varias metodologías para la programación de computadoras digitales.

Para esto es necesario seguir algunos principios metodológicos a fin de unificar los diferentes estilos individuales que permiten la comunicación, modificación, transportabilidad y mantenimiento de los sistemas de programación estructurada.

Las computadoras se han venido utilizando para dar solución a problemas cada vez más grandes y más complejos. Los programadores responsables de implementar las soluciones a estos problemas tienen responsabilidad de:

1. Organizar la información referente al problema.
2. Plantear un método de solución.
3. Conjugar esto en una representación entendible para la computadora.

En términos de ingeniería de programación estas son el análisis del problema, el diseño de su solución y la codificación.

La tareas de escribir un programa de computadora se hace más simple si el programa puede ser analizado en términos de subprogramas. El proceso de estructuración en la solución del problema usualmente se refleja en el programa haciendolo modular, es decir, consistente de varios subprogramas.

El concepto de modularidad en los programas no es nuevo. Desde hace algunos años existen sistemas operativos que fueron construidos en forma modular. Los frabricantes de la computadoras proporcionan al usuario un sistema operativo que consiste en muchos programas (módulos). Ya que el programa del sistema operativo, está en un estado continuo de cambio, los cambios pueden hacerse más fácilmente si todo el programa se divide en módulos cuyas interrelaciones son simples y claramente definidas.

Un programa complejo usualmente no puede ser escrito como un conjunto de módulos a menos que sus solución sea estructurada u organizada de esta forma. La programación de sistemas grandes usualmente envuelve a varias personas y las decisiones hechas por algunas de ellas no debe afectar a las demás. Esto puede lograrse solamente si la descripción y codificación de cada módulo y sus interfases se hacen tan clara y simple como sea posible. La modularidad de la mayoría de los sistemas, y específicamente del presentado en este trabajo puede ser representada como una estructura jerárquica con la que se muestra en la figura I.1.

El concepto de estructurar jerárquicamente un problema es una parte fundamental en la solución de problemas. Esta es la forma de organización o estructuración que nos permite entender un sistema de diferentes niveles y facilita hacer cambios de un nivel sin tener que comprender completamente descripciones más detalladas de los niveles superiores.

También es importante en este proceso de estructuración jerárquica la posibilidad de entender un módulo en un cierto nivel independientemente a los demás módulos del mismo nivel.

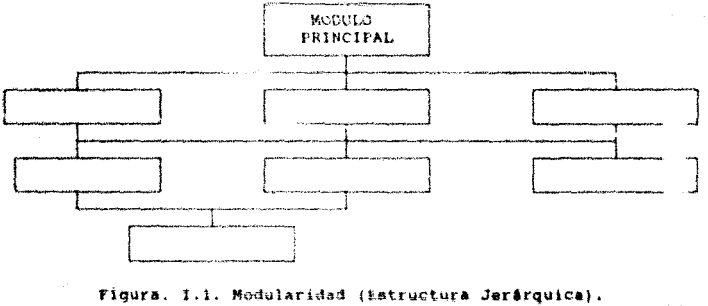


Figura. I.1. Modularidad (Estructura Jerárquica).

CAPITULO

DOS

CONCEPTOS

BASICOS

II.1 GENERALIDADES.

Las máquinas más antiguas funcionaban como el mecanismo de un reloj, sin admitir variación después de iniciado el movimiento; pero las modernas, poseen órganos sensoriales, es decir, mecanismo de recepción de información que provienen del exterior, tales como los proyectiles teledirigidos, el mecanismo de apertura automática de las puertas, etc. Claro está, la información no se toma en bruto, sino que pasa a través de los mecanismos especiales de transformación que posee el aparato. Por consiguiente, los datos adquieren una nueva forma utilizable en las etapas ulteriores de la actividad.

La riqueza de la información que se tiene que manejar en la actualidad da a entender un incremento en el sistema computacional cuyo manejo automático de esta información, deberá asignársele una estructura que facilite al sistema recibir y guardar información, y poder restablecerla para propósitos específicos.

El proceso de diseño de una base de datos requiere que se piense primero en la forma en que los usuarios van a preguntar sobre los datos. Así, datos y atributos son los términos importantes en el diseño de programas. Los datos están constituidos por la información que llega al programa. Los atributos son los tipos de datos que conforman el programa. En este respecto, la necesidad para estructurar la información requiere relaciones intrínsicas del sistema.

Estructuras de datos son los métodos que se emplean en programación para organizar y representar la información en una computadora. En general, un programa esta formado por un algoritmo(el que) y las estructuras de datos(el cómo).

El estudio de algoritmos es el corazón de la ciencia de la computación. En los últimos años se ha caracterizado por un avance significativo en el campo de los algoritmos; dichos avances se han desarrollado principalmente hacia algoritmos más rápidos, así como a ciertos problemas para los cuáles los algoritmos eran ineficientes.

II.1.1 ALGORITMOS.

Un algoritmo es una manera formal y sistemática de representar la descripción de un proceso. Los algoritmos son una parte de la computación que ha avanzado tecnológicamente al paso de los años, ya que ahora los algoritmos son mucho más rápidos que en años anteriores. Los algoritmos hoy en día son más eficientes ya que su desarrollo ha sido muy significativo.

Algunas características con las que cuenta un algoritmo son las siguientes:

1.- Finito.
Un algoritmo siempre terminará después de un número finito de pasos o procesos.

2.- Definido.
Los pasos(procesos) de un algoritmo deben ser bien definidos, esto es, cada paso debe estar bien especificado.

3.- Entradas.
Un algoritmo tiene cero o más entradas, esto es, realizar cuantitativamente la inicialización del algoritmo antes de empezarlo.

4.- Salidas.
Un algoritmo tiene una o más salidas, esto es, la relación específica que tiene con las entradas.

5.- Efectividad.
Un algoritmo debe ser eficaz, esto es, todas las operaciones y procesos deben ser desde un principio lo más exactas posibles.

Se hace la observación que la característica de 'finito' debe ser fuertemente usada. Un algoritmo útil debe requerir no solamente un número finito de pasos, sino un número finito de pasos razonable.

En la práctica no solo se quieren algoritmos, sino que se desean buenos algoritmos en algun sentido. un criterio de bondad es la longitud de tiempo que tarda la ejecución del algoritmo, esto puede ser expresado en términos del número de veces que cada paso es ejecutado. Otros criterios pueden ser la adaptabilidad del algoritmo a las computadoras, su simplicidad, su elegancia, etc.

En ocasiones se tienen varios algoritmos para el mismo problema, y se tiene que decidir cuál es el mejor. Esto nos permite introducirnos al importante campo del análisis de algoritmos, es decir, dado un algoritmo, el problema consiste en determinar sus características de ejecución.

II.2 ESTRUCTURAS DE DATOS BASICAS.

Los datos pueden organizarse en muchas formas diferentes; el modelo matemático ó lógico de una organización particular recibe el nombre de estructuras de datos, la elección de un modelo de dat s es particular, y depende de varias cuestiones, dos de ellas son as siguientes:

a) Debe ser lo suficientemente complejo para mostrarnos la relación entre los datos y lo que representan en sí.

b) La estructura debe ser lo suficientemente simple para que los datos puedan ser procesados de forma eficiente cuando se necesario.

Las cuestiones anteriores parecen contradictorias entre sí, ya que una nos dice que el sistema debe ser complejo y la otra simple, lo cual al combinarlo nos da como resultado una estructura de datos adecuada para nuestras tareas.

Una estructura de datos en su forma más general, consiste en una colección de nodos o registros que mantienen importantes relaciones entre sí. Como se muestra en la figura II.1

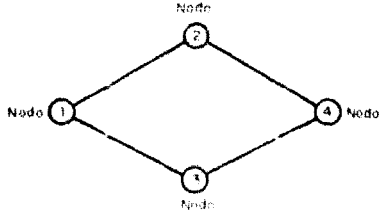


Figura II.1

Un nodo es el elemento básico para mantener la información en una estructura de datos. Un nodo puede subdividirse en campos de tal manera que resulte más fácil el manejo de la información.

Las estructuras de datos son importantes en al ámbito de la computación, ya que se siguen tres ideas principales en la aplicación de ellas, estas ideas son las siguientes:

a) Organización de la información.
b) Representación de la información en la computadora.
c) Operaciones que se realizan sobre la información.

Las estructuras de datos se han organizado de la siguiente forma:

1) Estructuras de datos elementales.

Son aquellos datos(números enteros, números reales, caracteres y otros tipos de datos) cuya manipulación y representación se ha estandarizado en los lenguajes de programación.

2) Estructuras de datos compuestas.

Son las pilas, colas, gráficas, árboles y otros tipos de datos, que requieren programarse, cuya manipulación y representación requieren del ingenio del usuario para que su utilidad sea de mayor provecho.

Con las estructuras de datos se pueden efectuar labores de manera más sencilla, ya que nos ayuda a realizar tareas de forma más óptima y eficaz, que sin su ayuda no se realizaría.

II.2.1 NECESIDAD PARA ESTRUCTURAR UN CONJUNTO DE DATOS.

Conforme a pasado el tiempo, la información que se tiene ha resultado cada vez mayor, por lo que en la actualidad se ha hecho necesario al manejar esta información por medio de sistemas de computación, esta información se maneja por medio de un sistema de información computacional.

Estos sistemas computacionales se pueden manejar automáticamente, por lo cual se les asigna una estructura, por medio de la cual se facilite la entrada, el proceso y salida de la información.

Los requerimientos para que la información sea procesada por un sistema de información, son los siguientes:

1) La estructura caracteriza la tecnología computacional actual, con la cual se logra una mejoría en la solución de los problemas.

2) Estructurar la información es útil porque permite un análisis adecuado de nuestra información.

3) La estructura de la información permite organizarla, para un mejor desarrollo de ella.

Actualmente las computadoras juegan un papel importante en la comodidad y rapidez para la realización de cálculos complicados y onerosos de tiempo, siendo la característica principal la gran capacidad de almacenamiento y acceso a grandes masas de información.

En la mayoría de las aplicaciones, la gran masa de información que es necesario procesar representa una abstracción de la realidad.

Es por ello que la información utilizada por la computadora consiste de una selección de datos de la realidad, esto es, un conjunto de datos que es relevante para el problema bajo estudio.

Al resolver cualquier problema, es necesario elegir una abstracción de la realidad guiada por el problema a resolver. Después debe seleccionarse la forma de representar esta información en base a las posibilidades concretas que ofrece la computadora. Generalmente estas dos etapas de diseño no son totalmente independientes una de otra.

Pero ¿como elegir entre dos o mas posibles estructuras de datos? La elección de la representación de los datos es a menudo bastante difícil y no está determinada exclusivamente por los instrumentos disponibles, ya que al igual que en la elección de algoritmos distintos, debe considerarse primero los requerimientos del programa (por ejemplo, la necesidad de ahorrar memoria o la posibilidad de acceder fácilmente a un registro particular). Si no existen diferencias entre estructuras, entonces debe considerarse la elegancia y claridad relativa de dichas estructuras.

El objetivo de la estructura de datos es el de poder manipular los datos de los programas en función de su representación lógica en lugar de su almacenamiento físico.

La importancia de utilizar un lenguaje que ofrezca un conjunto conveniente de operaciones fundamentales para resolver la mayoría de los problemas que se presentan en el procesamiento de datos, reside principalmente en la confiabilidad de los programas resultantes.

Es más fácil diseñar un programa razonado sobre conceptos familiares de números, conjuntos, sucesiones y repeticiones, que razonado con bits, "palabras", y saltos de secuencia.

Se define como una estructura de datos a la colección de datos cuya organización se caracteriza por las funciones de acceso que se usan para almacenar y acceder a elementos individuales de datos.

Cada estructura de datos puede analizarse desde tres perspectivas:

1.- NIVEL ABSTRACTO O LOGICO. En este nivel, se esquematiza la organización y se especifican los procedimientos y funciones generales de acceso.

2.- NIVEL DE IMPLANTACION. Aquí se examinan las formas de representación de los datos de memoria y cómo implantar los procedimientos y funciones de acceso en el lenguaje de programación. Se examinan las distintas formas en que pueden implantarse las estructuras de datos.

3.- NIVEL DE APLICACION O USO. En este nivel se presentan ejemplos relativos a los niveles anteriores, y se examinan en detalle algunos casos en los que la estructura de datos representa con precisión las relaciones entre los datos.

II.2.2 ELECCION DE UNA ESTRUCTURA DE DATOS.

Existen diferentes criterios para definir una estructura de datos, algunos de ellos se mencionan a continuación:

a) La estructura de datos a utilizar debe adecuarse a las necesidades de las tareas a ejecutarse sobre los datos.

b) Deben utilizarse los tipos de datos necesarios y adecuados para la resolución de las tareas a ejecutarse sobre los datos. Los tipos de datos a utilizar van desde los elementales(número entero,caracter,etc.), hasta arreglos, apuntadores, árboles, registros, etc.

c) El código de la estructura de datos debe ser lo más claro posible, esto es, para un mejor manejo del programa.

d) La eficiencia de la estructuras de datos es muy importante y esto se va a reflejar en la elección de estas en el diseño del programa.

II.3 TERMINOLOGIA BASICA.

II.3.1 ORGANIZACION ELEMENTAL DE LOS DATOS.

La palabra datos hace referencia a valores simples o a conjuntos de valores. Denominamos *elemento* a una unidad básica de valores. A aquellos elementos que pueden dividirse en otros reciben el nombre de *grupo de elementos*. Por el contrario, los no subdivisibles reciben el nombre de *elementos simples*.

Por ejemplo, el nombre de un empleado puede ser dividido en tres subunidades -nombre,primer apellido y segundo apellido -, pero el número de seguridad social normalmente debe ser tratado como una unidad simple. Las colecciones de datos se organizan jerárquicamente en campos, registros, y archivos.

Una entidad es algo que posee ciertos atributos o propiedades a los cuales pueden asignársele valores. Estos valores pueden ser numéricos o no. Entidades con atributos iguales forman un conjunto de entidades.

Campo es una unidad elemental de información que representa un atributo de una entidad, un registro es una colección de campos de una entidad y un archivo es una colección de registros de las entidades contenidas en un conjunto de entidades.

II.3.2 OPERACIONES CON ESTRUCTURAS DE DATOS.

Los datos que contienen una estructura se procesan por medio de determinadas operaciones, algunas de ellas son:

a) Recorrido. Implica el entrar a un registro una única vez aunque uno o más ítems sean procesados.

b) Busqueda. Implica localización de un registro caracterizado por una determinada clave o también el

acceso a todos los registros que cumplen una o más condiciones.

c) **Inserción**. Es cuando se añaden nuevos registros a la estructura.

d) **Eliminación**. Operación de borrado de un registro de la estructura

e) **Ordenación**. Es la operación de clasificar los registros conforme a un orden lógico determinado.

f) **Mezcla**. Es la operación de combinar dos archivos previamente ordenados en un tercero que también lo está.

II.3.3 VARIABLES, TIPOS DE DATOS.

Cada variable de nuestros algoritmos o programas lleva asociado un tipo que determina el código que utilizan para almacenar su valor correspondiente.

A continuación se mencionan algunos tipos de variables:

a) **Caracter**. Los datos son codificados utilizando algún código estándar.

b) **Real**. los datos son almacenados utilizando la forma exponencial de los datos.

c) **Entero**. los enteros positivos se codifican utilizando la representación binaria, y los negativos mediante variaciones de la misma, tales como el complemento a dos.

d) **Lógica**. Como las variables sólo pueden contener el valor verdadero o falso, pueden ser almacenadas utilizando un único bit, 1 para verdadero y 0 para falso.

Variables locales y globales.

La posibilidad de organizar un programa en varios subprogramas hace aparecer los conceptos de variables locales y globales. Normalmente, cada módulo de programa contiene su propia lista de variables que reciben el nombre de *variables locales*, y que solo pueden accederse desde dicho módulo.

Las variables que pueden ser accedidas por todos los módulos del sistema reciben el nombre de *variables globales*. Una variable es un nombre simbólico asociado a una celda en la memoria de la computadora, esta asociación se realiza de manera automática.

Todas las estructuras de datos utilizadas que son almacenadas en la memoria principal de la computadora, son *variables estáticas*. Estas variables estáticas existirán mientras la parte del programa en la que se encuentran declaradas, se esté ejecutando.

Las variables estáticas como registros y arreglos utilizadas para estructurar nuevos datos, nos sirven para realizar muchas aplicaciones.

Las *variables dinámicas* las utilizamos para evitar algunos problemas que puedan surgir en el manejo de variables estáticas.

CAPITULO

TRES

METODOS DE SOLUCION

III.1 INTRODUCCION.

Una de la áreas más productivas de la investigación de operaciones es la relacionada con la Teoría de Redes. La Teoría de Redes es una clase de modelos matemáticos que involucra la representación gráfica de ciertos problemas de optimización.

Las redes tienen una aplicación extensa en diversos campos como son: planeación, administración, ingeniería, química,etc.; en estos campos innumerables situaciones pueden formularse como modelos matemáticos de redes. Algunos ejemplos son: sistemas de producción y distribución, tráfico urbano, transporte colectivo, redes eléctricas, reemplazo de equipo, inventarios presas, flujo de dinero, tuberías, oleoductos, asignación de recursos y otros más.

A causa de la vasta aplicación de este tipo de modelos y a la valiosa ayuda que proporcionan para el entendimiento de los sistemas, ha habido gran actividad en sus estudio.

En la Teoría de Redes existe una variedad de problemas básicos que han sido resueltos de manera sencilla y eficiente. Tales son con frecuencia, el resultado de problemas complejos de investigación de operaciones como: diseño de redes de carreteras, redes eléctricas y rutas de recolección de objetos.

Los variados métodos de solución desarrollados para ellos ha desprendido principalmente dos ramas desarrolladas paralelamente: la programación lineal y la teoría de redes eléctricas. En los últimos años se han hecho intentos por unificar estas ramas enriqueciendo de este modo la teoría de redes.

Los conceptos y técnicas antes desligados en la literatura se han relacionado dando lugar a una teoría general que engloba los problemas básicos de redes.

La herramienta principal de optimalidad utilizada para los modelos de redes la constituyen los resultados de teoría de dualidad que toman una forma particular en este tipo de problemas.

Se definen los conceptos de flujo y potencial y a partir de ellos los de divergencia y diferencial. Otros conceptos duales que tienen injerencia en la unificación de teorías resultan ser los de trayectoria y corte, que pueden ser vistos como flujo y un diferencial especiales en la red.

De hecho en base a todos estos conceptos surge un resultado conocido como teorema de la red coloreada que se aplica en el desarrollo de resultados teóricos y algoritmos de solución para los modelos de redes. Otros resultados de dualidad relacionan los

problemas de flujo máximo y trayectoria mínima con los de corte mínimo y tensión máxima respectivamente.

III.2 CONECTIVIDAD

III.2.1 MATRIZ DE ADYACENCIA.

Una forma de saber si una gráfica es conexa o no, es con ayuda de la matriz de adyacencia A. Esta matriz es de orden n*n, donde n es el número de nodos de la red.

La matriz de adyacencia $A = (a_{ij})$ de una gráfica G es la matriz de m*m elementos definida como sigue:

$$a_{ij} \begin{cases} 1 & \text{si existe un arco del nodo i al nodo j} \\ 0 & \text{en caso contrario} \end{cases}$$

La matriz de adyacencia A de la gráfica G depende de la ordenación de los nodos de la gráfica G; esto es, diferentes ordenaciones de los nodos pueden resultar en diferentes matrices de adyacencia. Sin embargo, las matrices obtenidas por dos ordenaciones diferentes están fuertemente relacionadas en cuanto que una pueda ser obtenida de la otra simplemente cambiando filas y columnas.

Por lo tanto una forma de saber si la gráfica es conexa o no, es encontrando la siguiente suma:

$$Y = A + A^2 + A^3 + A^4 + \ldots + A^{n-1}$$

donde:

A Es la matriz de adyacencia de la gráfica, cuyas entradas son ceros y unos.

A^2 Es la matriz de adyacencia cuyas entradas corresponden al número de diferentes trayectorias entre dos nodos, las cuales usan dos arcos.

A^3 Es la matriz de adyacencia cuyas entradas son la (i,j)-ésima posición, es el número de diferentes trayectorias entre los nodos (i,j), los cuales usan tres arcos.

A^{n-1} Es la matriz de adyacencia cuyas entradas en la (i,j)-ésima posición indica el número de diferentes trayectorias que existen entre los nodos (i,j), los cuáles usan n-1 arcos.

Si una gráfica está conectada, es posible encontrar una trayectoria entre toda pareja de nodos los cuáles usan a lo más n-1 arcos. De este modo, al menos una de las matrices debe tener una entrada diferente de cero en la posición (i,j)-ésima si existe una trayectoria entre los nodos (i,j).

Por lo tanto si la matriz Y es totalmente diferente de cero, entonces la res es conexa, en caso contrario la red es no conexa.

Ejemplo:

Sea la gráfica G, de la figura III.1

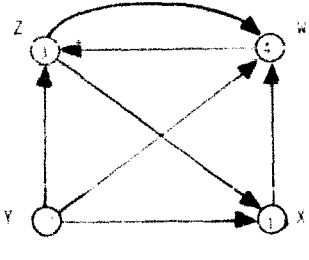


Figura III.1 Gráfica G

Entonces:

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad A^2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 2 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$A^3 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 2 & 2 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \qquad A^4 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 2 & 0 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

Por lo tanto :

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

De modo que la gráfica G no es fuertemente conexa, ya que el nodo Y(V) no es alcanzado por ninguno de los otros nodos.

III.2.2 MATRIZ DE CAMINOS.

Sea G una gráfica dirigida simple con m nodos, $v_1, v_2, \ldots, v_m$. La matriz de caminos o matriz de alcances de G es la matriz m-cuadrada $P = (p_{ij})$ definida de la siguiente manera:

$$p_{ij} \begin{cases} 1 & \text{si hay un camino de } v_i \text{ hasta } v_j \\ 0 & \text{en otro caso} \end{cases}$$

Suponiendo que existe un camino desde v_i hasta v_j, entonces tiene que haber un camino simple de v_i a v_j o un ciclo de v_i a v_j si $v_i = v_j$. Como G sólo tiene m nodos, un camino simple así ha

de tener longitud m - 1 o menor, o un ciclo así ha tener longitud m o menor.

Se cuenta con un algoritmo para obtener la matriz de caminos, llamado *Algoritmo de Warshall*.

ALGORITMO DE WARSHALL.

Sea G una gráfica dirigida con m nodos, v_1, v_2,...,v_m. Supónga que se quiere encontrar la matriz de caminos P para la gráfica G. Warshall dió un algoritmo para este propósito que es mucho más eficiente que calcular la matriz de adyacencia.

Para describir este algoritmo, primero se definirán las matrices m-cuadradas booleanas $P_0, P_1, \ldots, P_m$ como sigue. Sea $P_k[i,j]$ la entrada ij de la matriz P_k. Así definimos:

$$P_k[i,j] \begin{cases} 1 & \text{si existe un camino simple de } v_i \text{ a } v_j \text{ que no usa otros nodos aparte de posiblemente } v_1, v_2, \ldots, v_k \\ 0 & \text{en otro caso} \end{cases}$$

Warshall observó que $P_k[i,j]=1$ sólo puede ocurrir si se da uno de los dos siguientes casos:

1) Existe un camino simple de v_i a v_j que no usa otros nodos excepto posiblemente $v_1, v_2, \ldots, v_{k-1}$; por tanto:

$$P_{k-1}[i,j] = 1$$

2) Existe un camino simple de v_i a v_k y otro camino simple de v_k a v_j que no usan otros nodos excepto posiblemente $v_1, v_2, \ldots, v_{k-1}$; por tanto

$$P_{k-1}[i,j] = 1 \qquad \text{y} \qquad P_{k-1}[k,j] = 1$$

El algoritmo de Warshall se muestra a continuación: Sea una gráfica dirigida G con M nodos que se encuentra en memoria representada por su matriz de adyacencia A. Este algoritmo encuentra la matriz de caminos(booleana) P de la gráfica G.

ALGORITMO DE WARSHALL.

Paso 1.
[Inicializar P]. Repetir para I,J=1,2,...,M;
Si A[I,J]=0 entonces: Hacer P[I,J]:=0;
Si no: Hacer P[I,J]:=1.
[Fin de ciclo]

Paso 2.
[Actualizar P]. Repetir pasos 3 y 4 para k=1,2,...,M:

Paso 3.
Repetir paso 4 para I=1,2,...,M:

Paso 4.
Repetir para J=1,2,...,M:
Hacer P[I,J]:=P[I,J] $\vee$ (P[I,k] $\wedge$ P[K,J])
[Fin de ciclo]
[Fin del ciclo paso 3]
[Fin del paso 2]

Paso 5. Salir.

Ejemplo:

Conociendo la matriz Y (tomándola del ejemplo de la figura III.1,de la matriz de adyacencia).

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

Reemplazando las entradas positivas por 1 en la matriz anterior, se obtiene la matriz de caminos P de la red G, la cual se muestra a continuación:

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

La matriz de caminos P nos dice si hay o no caminos entre los nodos. Ahora se requiere encontrar una matriz $Q = (q_{ij})$ que tenga las longitudes de los caminos mínimos entre los nodos. Esto es, una matriz $Q = (q_{ij})$, donde

q_{ij} = longitud del camino mínimo de v_i a v_j

Para encontrar la matriz de longitud mínima entre los nodos de la red, se definirá una secuencia de matrices Q_1, Q_2, ..., Q_n (análogas a las anteriores matrices P_0, P_1, ...,P_n), cuyas entradas vienen dadas por:

Q[i,j] = La menor de las longitudes de los anteriores caminos de v_i a v_j o la suma de las longitudes de los anteriores caminos de v_i a v_k y de v_k a v_j .

Más exactamente,

$$Q[i,j] = MIN\ (Q_{k-1}[i,j], Q_{k-1}[i,k] + Q_{k-1}[k,j])$$

La matriz inicial Q_0 es la misma que la matriz de pesos W excepto que cada 0 de W se reemplaza por infinito (o un número grande, muy grande). La matriz final Q_n será la matriz Q deseada.

Ejemplo:

Considere la red con peso de la figura III.2. Tomando en cuenta que v_1 = R, v_2 = S, v_3 = T y v_4 = U, y la matriz de pesos W de la red G es la siguiente:

$$W = \begin{bmatrix} 7 & 5 & 0 & 0 \\ 7 & 0 & 0 & 2 \\ 0 & 3 & 0 & 0 \\ 4 & 0 & 1 & 0 \end{bmatrix}$$

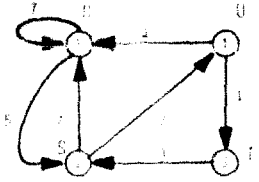


Figura III.. Red con Peso.

Aplicando el algoritmo de Warshall, obtendremos la matriz de caminos mínimos Q.

$$Q_0 = \begin{bmatrix} 7 & 5 & \alpha & \alpha \\ 7 & \alpha & \alpha & 2 \\ \alpha & 3 & \alpha & \alpha \\ 4 & \alpha & 1 & \alpha \end{bmatrix} \qquad Q_1 = \begin{bmatrix} 7 & 5 & \alpha & \alpha \\ 7 & 12 & \alpha & 2 \\ \alpha & 3 & \alpha & \alpha \\ 4 & 9 & 1 & \alpha \end{bmatrix}$$

$$Q_4 = \begin{bmatrix} 7 & 5 & 8 & 7 \\ 7 & 12 & 3 & 2 \\ 9 & 3 & 6 & 5 \\ 4 & 4 & 1 & 6 \end{bmatrix}$$ MATRIZ DE CAMINOS MINIMOS

III.2.3 BUSQUEDA EN ANCHURA.

Algunas redes de flujo requieren que se examinen sus nodos y sus aristas, esto se puede realizar por medio de la búsqueda en anchura.

La idea general de la búsqueda en anchura es iniciado en el primer nodo, el será el nodo de partida de la red. Primero se examinará el nodo de partida de la red, luego se examinarán los nodos vecinos del nodo de partida, después de examinarán los vecinos de los vecinos del nodo de partida, y así sucesivamente hasta llegar al último nodo de la red. Se tiene que seguir el camino que van tomando los vecinos de un nodo ya que no se puede visitar dos veces el mismo nodo.

Ejemplo:

Considérese la red G de la figura III.3:

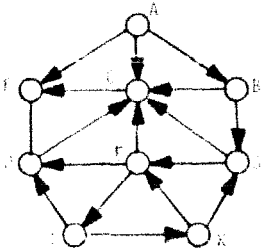


Figura III.3 Red G

Suponga que la red G representa los vuelos diarios entre ciudades de una compañía aérea, y suponga que deseamos volar de la ciudad A a la ciudad J con el numero mínimo de paradas, de otro modo, se requiere saber el camino mínimo P desde la ciudad A hasta la ciudad J. (Considere que cada arista tiene un valor unitario).

Solución:

El camino mínimo P que se requiere se iniciará en la ciudad A y terminará en la ciudad J.

Se iniciará la busqueda en el nodo de partida: Ciudad A
se examinan los nodos vecinos del nodo de partida: los vecinos del nodo de partida son los nodos: F, C, B

Nodos visitados	A, F, C, B
Nodos origen	0, A, A, A

Ahora se examinarán los vecinos del vecino F del nodo A; el vecino del nodo F es el nodo D

Nodos visitados	A, F, C, B, D
Nodos origen	0, A, A, A, F

Ahora se examinarán los vecinos del vecino C del nodo A; el vecino del nodo C es el nodo F, el cual ya se tiene.

Nodos visitados	A, F, C, B, D
Nodos origen	0, A, A, A, F

Ahora se examinarán los vecinos del vecino B del nodo A; los vecinos del nodo B son los nodos C y G, dond el nodo C ya se examinó.

Nodos visitados	A, F, C, B, D, G
Nodos origen	0, A, A, A, F, B

Ahora se examinarán los vecinos del vecino D del vecino F del nodo A: el vecino del nodo D es el nodo C, que ya se examinó.

Nodos visitados	A, F, C, B, D, G
Nodos origen	0, A, A, A, F, B

Ahora se examinarán los vecinos del vecino G del vecino B del nodo A: el vecino del nodo G es el nodo E

Nodos visitados	A, F, C, B, D, G, E
Nodos origen	0, A, A, A, F, B, G

Ahora se examinarán los vecinos del vecino E del vecino G del vecino B del nodo A; el vecino del nodo E es el nodo J.

Nodos visitados	A, F, C, B, D, G, E
Nodos origen	0, A, A, A, F, B, G, E

Como ya se visitó el nodo J que es el destino final, se tiene que el camino mínimo buscado es el siguiente:

Nodo Final	J
Nodo origen del nodo final	E
Nodo origen del nodo E	G
Nodo origen del nodo G	B
Nodo origen del nodo B	A

el cual es el camino mínimo de la ciudad A a la ciudad J.

III.2.4 BUSQUEDA EN PROFUNDIDAD.

Algunas redes de flujo requieren que se examinen sus nodos y sus aristas, esto se puede realizar por medio de la búsqueda en profundidad.

La idea general de la búsqueda en profundidad comenzando con un nodo inicial es la siguiente. Primero se examinará el nodo inicial, luego se examinará cada nodo N de un camino P que comience en A; esto es, se visita un vecino de A, luego un vecino del vecino de A y así sucesivamente, hasta llegar al nodo final del camino P, se regresa por el mismo camino P hasta continuar por otro camino P'. Y así sucesivamente, hasta recorrer toda la red.

III.3 RUTA MAS CORTA.

III.3.1 DESCRIPCION DEL PROBLEMA.

Se presentan métodos de solución para los siguientes problemas de rutas más cortas en una red R=[X,A,a]:

1.- Ruta más corta entre dos vértices específicos f y t.

2.- Rutas más cortas entre un vértice específico s y todo vértice x en la red.

3.- Ruta más corta entre todo par de vértices.

Para ejemplificar consideremos el siguiente problema.

En una terminal de camiones de pasajeros se desea establecer la ruta que deberá seguir el autobus que presta servicio de la ciudad s a la ciudad t de tal manera que la distancia recorrida sea lo más corta posible. A este problema se le puede asociar una red R [X,A,d] donde:

X = {Ciudades a las cuales se ofrece el servicio}

A = {Tramos de carretera entre las ciudades}

d : A -> R donde, para todo a ∈ A, d(a)=longitud o distancia del tramo de carretera a.

En general, en una red R=(X,A,d), al número d(a) asociado a cada arco se le llama longitud o costo de a. Por otro lado se define longitud de una ruta o camino como la suma de las longitudes de los arcos que la forman; aquella ruta tal que su longitud sea mínima se le llama ruta más corta o camino más corto.

El problema de la terminal de autobuses es entonces encontrar la ruta más corta entre dos vértices específicos, los que representan a las ciudades s y t. Obsérvese que en este caso las longitudes definidas son no negativas; sin embargo el problema de encontrar la ruta más corta entre dos vértices específicos puede generalizarse a cualquier red puesto que la función de longitud d, puede representar, además de distancia tiempo, costos o alguna otra cantidad.

Si la red contiene arcos con longitudes negativas pueden presentarse circuitos negativos (circuitos de longitud negativa). En este caso el problema puede ser no acotado puesto que para cada ruta entre s y t que contenga el circuito negativo existe otra mejor, a saber aquella que contiene una vez más al circuito. En la red de la figura III.4 ocurre ésto.

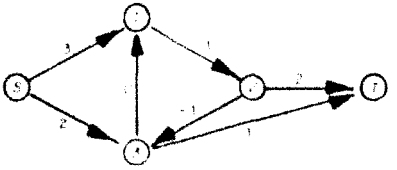


Figura III.4 Red con longitudes Negativas

Una ruta de s a t es: s,1,3,2,3,t de longitud 7. Otra ruta de s a t, mejor que la anterior es: s,1,3,2,3,2,3,t, de longitud 5. De hecho si se considera una ruta que contenga el arco (s,1), M veces el circuito 1,3,2,1, (M > 0) y el arco (3,t), la longitud será 9-2M; de tal modo que si M tiende a infinito, la longitud de la ruta tiende a menos infinito. Luego la ruta más corta entre s y t no existe.

Se concluye entonces que, para que el problema de la ruta más corta entre dos vértices específicos tenga solución, deberá cumplir que:

a) Exista algun camino entre s y t.

b) No existan circuitos negativos tales que haya un camino de s a algún vértice del circuito y otro de algún vértice del circuito a t.

III.3.2 CONCEPTOS UTILES.

Sea G = {X,A} una gráfica dirigida y sea s∈X; entonces s se llama raíz de G, si existe camino de s a x para todo x ∈ X. Una arborescencia es un árbol que tiene una raíz. La gráfica de la figura III.5 es una arborescencia de raíz uno.

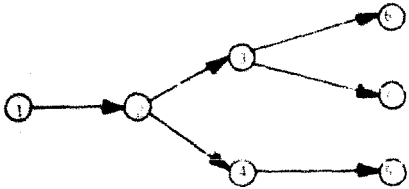


Figura III.5 Arborescencia de Raíz Uno

Sea G = {X,A} una gráfica dirigida. Una arborescencia de G es un árbol expandido de G que tiene un vértice que es raíz.

Considérese ahora una red R={X,A}. Una arborescencia de rutas más cortas de R es aquella arborescencia tal que la única ruta de s a x, para todo x∈X, es una ruta más corta de s a x.

Haciéndose una analogía con el problema de la ruta más corta entre dos vértices específicos puede concluirse que para que exista la arborescencia de rutas más cortas de raíz s en una red cualquiera R={X,A,d}, ésta deberá cumplir que:

i) Existan caminos de s a x, para todo x∈X. Es decir, que s sea raíz de la red.

ii) No existen circuitos negativos en la red R, ya que de presentarse estos el problema sería no acotado.

Para que exista solución en cualquier red R={X,A,d}, deberá cumplirse lo siguiente:

i) Existe, al menos, un camino entre todo par de vértices.

ii) No existen circuitos negativos en la red R.

III.3.3 METODOS DE SOLUCION.

Se presentarán algunos métodos de solución para el problema de la arborescencia de rutas más cortas de raíz s en una red R = (X,A,d). El primero de ellos sólo es aplicable a redes que tienen arcos con costos no negativos y el segundo puede utilizarse para cualquier red.

En la práctica existe gran cantidad de problemas que involucran costos no negativos (tiempo,distancia, etc.); es por esta razón que se justifica el desarrollo de algoritmos que se aplican sólo a estos casos.

Es importante mencionar que estos algoritmos tienen la ventaja de que, además de proporcionar la solucion óptima cuando existe, detectan cuando ésta no existe; ya sea que dicha solución no exista porque s no es raíz de la red o por la presencia de circuitos negativos.

Cabe señalar que estos métodos sirven también para encontrar la solución óptima del problema de la ruta más corta entre todo par de vértices.

Para el problema de la ruta más corta entre todo par de vértices se presenta un algoritmo con la misma ventaja que los anteriores. Más aún, si la ruta más corta existe para algunos pares de vértices y para otros no, el algoritmo proporciona las longitudes de las rutas existentes y detecta para qué pares de vértices no existe ninguna ruta.

Caso de Redes con Costos No Negativos

El método de solución presentado para el problema de la arborescencia de rutas más cortas en redes que tienen arcos con costos no negativos fué desarrollado por **Dijkstra** (1959) y está considerado como el método más eficiente para resolver este problema.

Este método se basa en la asignación de etiquetas **'permanentes'** a los vértices para los cuales ya se conocen las longitudes de las rutas más cortas de la raíz a ellos. Sea S este conjunto de vértices. Las etiquetas de los vértices de S representan precisamente las longitudes de las rutas cortas más buscadas.

Los vértices restantes se etiquetan **'temporalmente'** con una cota superior de la longitud más corta de la raíz al vértice etiquetado.

En la primera iteración el conjunto T contendrá únicamente al vértice de la raíz; es decir, sólo la raíz estará etiquetada permanentemente.

Las etiquetas temporales se mejoran continuamente y en cada iteración se agrega exactamente un vértice x a S; este vértice es aquel tal que la longitud desde la raíz es la más corta posible.

Puesto que todos los arcos tienen costos no-negativos, siempre puede encontrarse una ruta más corta de la raíz a x que pase sólo por vértices de S; en este caso la etiqueta de X representa la longitud de la ruta más corta correspondiente.

Una vez que todos los vértices estén en S, las etiquetas de todos los vértices serán las correspondientes a las longitudes más cortas desde la raíz y por lo tanto se había encontrado la solución deseada.

En el caso de que se desee sólo la ruta más corta entre dos vértices específicos, se obtendrá la solución cuando se etiquete 'permanentemente' el vértice final del camino buscado.

El algoritmo de Dijkstra obtiene la arborescencia de rutas más cortas de raíz s en una red R = (X,A,d) con costos no negativos en los arcos.

ALGORITMO DE DIJKSTRA

Paso 1.

(Iniciación de etiquetas).

Sea d(s)=0 y márquese esta etiqueta como permanente. Sea d(x)=A, para todo x ≠ s y considérense estas etiquetas como temporales. Sean a(x)=x (estas etiquetas indicarán el predecesor de x en la arborescencia). Sea p=s.

Paso 2.

(Actualización de etiquetas).

Para todo $x \in \Gamma^{+}(p)$ que tenga etiqueta temporal, actualizar etiquetas de acuerdo a:

$$d(x) = \min \{ d(x), d(p) + d(p,x) \}$$

si d(x) se modificó, hacer a(x)=p. Sea x^{*} tal que $d(x^{*}) = \min d(x) | d(x)$ es temporal . Si $d(x^{*})=a$, terminar. En este caso no existe arborescencia alguna de raíz s. En otro caso, marcar la etiqueta $d(x^{*})$ como permanente. Sea $p=x^{*}$.

Paso 3.

(i) (Si solo se desea la ruta de s a t). Si p=t, terminar; d(p) es la longitud del camino más corto. Si p≠t, ir al paso 2.

(ii) (Si se desea la arborescencia). Si todos los vértices tienen etiquetas permanentes, terminar; esta es la longitud del camino deseado y el conjunto de arcos (a(x),x) forman la arborescencia de caminos más cortos. En otro caso ir al paso 2.

Ejemplo.

Supóngase que en un aeropuerto se considera la posible adquisición de un equipo que, a causas de ciertas modificaciones, será inútil dentro de tres años. Los costos de utilización y de reventa del equipo usado son conocidos.

El equipo puede ser utilizado duante uno, dos o tres años, revenderse al final del periodo de utilización y comprarse uno nuevo para usarse durante el tiempo restante.

Los costos totales en los que se incurre (costo de compra más costo de utilización menos precio de venta) están dados en la matriz de costos C.

$$C = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{bmatrix} 4 & 8 & 16 \\ - & 5 & 11 \\ - & - & 6 \end{bmatrix} \end{array}$$

El elemento (i,j) de la matriz es igual al costo total en millones de pesos en el que se incurre si se compra un equipo al final del año i, se utiliza hasta el final del año j y se revende. Se desea encontrar la estrategia más económica de compra y reventa de equipo.

Para resolver este problema considérese la red R=(X,A,C) donde: X= 0,1,2,3 y cada elemento de C indica el final de un año; (i,j)∈A si y solo si es posible comprar el equipo a final del año i y revenderlo al final del año j; finalmente C(i,j) es el costo en el que se incurre por compara el equipo al final del año i, usarlo y revenderlo al final del año j.

Una forma esquemática de la red se observa en la figura III.6.

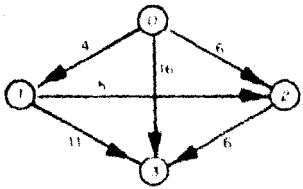


Figura III.6 Forma esquemática de la red de adquisición de equipo.

Obsérvese que una ruta de 0 a 3 en la red corresponde a una estrategia posible y viceversa; por otro lado, el costo de una estrategia posible es el mismo que el de la ruta correspondiente.

Por ejemplo, la ruta 0,1,3 (de longitud 15) corresponde a comprar al final del año 0, utilizar el equipo durante un año, revenderlo y comprar uno nuevo al final del año 1, utilizarlo durante dos años y revenderlo al final del año 3 (con un costo total de 4+11=15 millones). Por otra parte, comprar el equipo al final del año 0, utilizarlo durante tres años y revendarlo al final del año 3 (con un costo total de 16 millones) corresponde a la ruta 0,3 (de longitud 16).

Para encontrar la estrategia óptima de deberá, entonces encontrar la ruta más corta entre los vértices 0 y 3 en al red. Para ésto se aplicará el algoritmo de Dijkstra considerando s=0 y t=3.

Iteración 1. d(0)=0 (etiqueta permanente); d(x)=α , para x=1,2,3 (etiquetas temporales), p=0.

Actualización de etiquetas: $\Gamma^{+}(p)$= 1,2,3

d(1) = min α ,4 = 4 a(1) = 0
d(2) = min α ,5 = 5 a(2) = 0
d(3) = min α ,16 = 16

De donde x^{*}=1 (vértice con mínima etiqueta temporal). Se marca d(1) como permanente; p=1. (p≠t=3)

Iteración 2. Actualización de etiquetas: $\Gamma^{+}(p)$= 2,3

d(2) = min 5, 4+5 = 8 a(2) no cambia
d(3) = min 16,4+11 =15 a(3)=1

De donde $x^{*}=2$ (vértice con mínima etiqueta temporal). Se marca d(2) como permanente; p=2 ($p \neq t$).

Iteración 3. Actualización de etiquetas: $\Gamma^{+}(p)=3$

$$d(3) = \min\ [5,\ 8+6\ =\ 14] \qquad a(3)=2$$

De donde $x^{*}=3$ (vértice con mínima etiqueta temporal). Se marca d(3) como permanente; p=3=t. Se termina.

Para recuperar la ruta más corta de 1 a 3, de longitud d(3)=14, se utilizan las etiquetas a(x). La ruta deseada en sentido inverso, es: 3, a(3)=2, a(2)=0. Luego la estrategia óptima para el problema del aeropuerto es comprar el equipo al final del año 0, utilizarlo durante dos años, revenderlo y comprar un nuevo al final del año 2, utilizar este último durante un año y venderlo al final del año 3 con un costo total de 14 millones de pesos.

Ruta Más Corta entre Todo Par de Vértices.

Una ruta corta entre todo par de vértices en una red R = [X,a,d] es encontrar la arborescencia de rutas más cortas de la raíz x para todo $x \in X$. El algoritmo para encontrar la ruta más corta entre todo par de vértices fue desarrollado por R.W. Floyd (1962) y es aplicable a redes que permiten cualquier costo en sus arcos.

En dicho algoritmo se supondrá la numeración de los vértices de la red 1,2,...,n y se utilizará una matriz C, de orden nxn, para calcular las longitudes de las rutas más cortas entre cada par de vértices; al terminar de aplicar el algoritmo, la longitud de la ruta más corta entre los vértices i y j es dada por el elemento (i,j) de C.

En el algoritmo de Floyd, en la k-ésima iteración se calcula la longitud de la ruta más corta entre i y j que puede admitir a los primeros k-vértices, o a alguno de ellos, como vértices intermedios; este número se almacena en la entrada (i,j) de la matriz C. Al inicio se asigna el costo del arco (i,j), al elemento (i,j) de la matriz C, si $(i,j) \in A$; si dicho arco no existe entonces se asigna a . Los valores de la diagonal se ponen en 0.

Con esto quedan calculadas las longitudes de las rutas más cortas, entre todo par de vértices i y j, que no contengan ningún vértice como vértice intermedio.

Al principio de la k-ésima iteración, la entrada (i,j) de C es igual a la longitud de la ruta más corta, entre i y j, que contiene a los primeros k-1 vértices, o a algunos de ellos, como vértices intermedios.

Durante esta iteración se compara la longitud de esta ruta con la de aquella formada por la unión de las rutas más cortas, que contienen a los primeros k-1 vértices como vértices intermedios, entre i y k, y, k y j; de esta manera se obtiene la ruta más corta, entre i y j, que contiene a los primeros k vértices, o a algunos de ellos, como vértices intermedios.

Procediendo de esta manera se tendrá que, al final de la n-ésima iteración, la entrada (i,j) de C es la longitud de la ruta más corta, entre i y j, que contiene a los primeros n vértices como vértices intermedios o a algunos de ellos; es decir, se habrá calculado la longitud de la ruta más corta entre i y j.

Debe observarse que si, al finalizar de aplicar el algoritmo alguna entrada de C es igual a α, esto querrá decir que no existe ruta alguna entre los vértices correspondientes.

Por otro lado si algún elemento de la diagonal C, *supóngase* el (i,i), es menor que cero en alguna iteración, se habrá encontrado una ruta de i a i de longitud negativa (es decir, un circuito negativo). Luego en este caso, el problema no tiene solución.

Por esto mencionado últimamente, este algoritmo será de gran utilidad en problemas de detección de circuitos negativos.

El algoritmo de Floyd obtiene *las rutas más cortas* entre todo par de vértices en una red $R = (X, A, d)$ con n vértices.

ALGORITMO DE FLOYD

Paso 1.

Constrúyase la matriz C, de nxn, de elementos C_{ij}

$$C_{ij} = \begin{cases} 0 & \text{si } i = j \\ \alpha & \text{si } (i,j) \notin A \\ d(i,j) & \text{si } (i,j) \in A \end{cases}$$

Hágase $k = 0$.

Paso 2.
Hacer k=k+1. Para todo i≠k tal que $C_{ij} \neq \infty$, y para todo j ≠ k tal que $C_{kj} \neq \infty$, hacer:

$$C_{ij} = \min \; C_{ij}, \; C_{ik} + C_{kj}$$

Paso 3.

(i) Si $C_{ii} < 0$ para alguna i, terminar. En este caso existe un circuito negativo que contiene al vértice i y por lo tanto no hay solución.

(ii) Si $C_{ii} \neq 0$, para toda i, y k=n, terminar; C_{ij} es la longitud del camino mínimo más corto de i a j.

(iii) Si $C_{ii} \neq 0$, para toda i, y k < n, ir al paso 2.

Recuperación de las Rutas.

Para recuperar las rutas más cortas puede construirse una matriz A de dimensión nxn; el elemento a_{ij} de esta matriz será el predecesor del vértice j en la ruta i a j encontrada en cada iteración.

Dada la definición de A, sus entradas se inicializarán $a_{ij} = i$, para todo par de vértices i, j ∈ X. A será modificada en el paso 2 de la k-ésima iteración de acuerdo con:

$$a_{ij} = \begin{cases} a_{kj} & \text{si } C_{ik} + C_{kj} < C_{ij} \\ \text{no cambia, si } C_{ij} \leq C_{ik} + C_{kj} \end{cases}$$

Ejemplo.

Considérese la red de la figura III.7 y determínese las rutas más cortas entre todo par de vértices.

Se utilizará el algoritmo de Floyd para la determinación de las rutas.

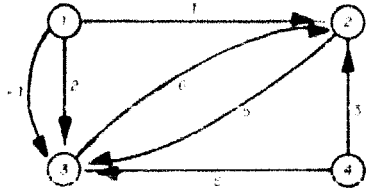


Figura III.7 Red G (Ruta más Corta)

Se utilizará el algoritmo de Floyd para la determinación de las rutas.

Iteración 1. Las matrices C y A iniciales son las siguientes:

$$C = \begin{bmatrix} 0 & 1 & -1 & \alpha \\ \alpha & 0 & 5 & \alpha \\ 2 & 6 & 0 & \alpha \\ \alpha & 3 & -6 & 0 \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{bmatrix}$$

Sea k=1. Se actualizan los valores de los elementos:

$C_{32} = \min\ 6,\ 2 + 1 = 3$ $\qquad a_{32} = a_{12} = 1$

$C_{33} = \min\ 0,\ 2 + -1 = 0$ $\qquad$ no se modifica a_{33}

Entonces, las matrices resultantes de esta iteración son las siguientes:

$$C = \begin{bmatrix} 0 & 1 & -1 & \alpha \\ \alpha & 0 & 5 & \alpha \\ 2 & 3 & 0 & \alpha \\ \alpha & 3 & -6 & 0 \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 1 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{bmatrix}$$

Iteración 2. Se asigna k=2 y se actualizan los elementos:

$C_{13} = \min\ -1, 1+5 = -1$ no se modifica a_{13}

$C_{33} = \min\ 0, 3+5 = 0$ no se modifica a_{33}

$C_{43} = \min\ -6, 3+5 = 0$ no se modifica a_{43}

Luego, las matrices C y A no fueron modificadas durante la segunda iteración.

Iteración 3. Se asigna k=3 y se actualizan los elementos:

$C_{11} = \min\ 0, -1+2 = 0$ no se modifica a_{11}

$C_{12} = \min\ 1, -1+3 = 0$ no se modifica a_{12}

$C_{21} = \min\ \alpha, 5+2 = 0$ $a_{21} = a_{31} = 3$

$C_{22} = \min\ 0, 5+3 = 0$ no se modifica a_{22}

$C_{41} = \min\ \alpha, -6+2 = 0$ $a_{41} = a_{11} = 3$

$C_{42} = \min\ 3, -6+3 = 0$ $a_{43} = a_{13} = 1$

Las matrices A y C resultantes son, respectivamente:

$$C = \begin{bmatrix} 0 & 1 & -1 & \alpha \\ 7 & 0 & 5 & \alpha \\ 2 & 3 & 0 & \alpha \\ -4 & -1 & -6 & 0 \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 3 & 2 & 3 & 3 \\ 3 & 1 & 3 & 3 \\ 3 & 1 & 4 & 4 \end{bmatrix}$$

Iteracion 4. Se asigna k=4. En esta iteración no se realiza ningún cambio puesto que $C_{i4} = \alpha$, para i=1,2,3. Entonces la última matriz C es la matriz de longitudes más cortas. El elemento $a_{i,j}$ de la última matriz A es es el predecesor del vértice j, en la ruta más corta de i a j, siempre y cuando $C_{i,j} \neq \alpha$. Obsérvese que, puesto que $C_{i4} = \alpha$ (para i=1,2,3), entonces no existe ninguna ruta entre los vértices i y 4 (para i=1,2,3); ésto puede verificar fácilmente en la red.

III.4 FLUJO MAXIMO.

Considerando un sistema de tuberías de transporte de agua como se muestra en la figura III.8:

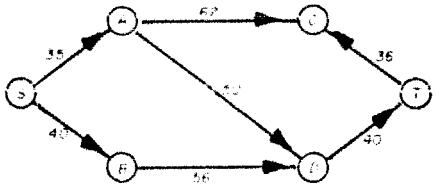


Figura III.8 Sistema de Tuberías de Transporte

La figura III.8 nos muestra una red, donde cada arco representa un tubo y el peso del arco representa la capacidad de agua que resiste la tubería en galones por minuto. Los nodos representan puntos en los cuales las tuberías están unidas y el agua es transportada de un tubo a otro. Los nodos, S y T, representan la fuente de agua, y un usuario de ésta, respectivamente. Esto significa que el agua nace o se origina en el nodo S, es llevada a través de la tubería del sistema hasta el nodo T.

El agua fluye a través de la tubería en una sola dirección y no existen tuberías entrando al nodo S o saliendo del nodo T. Por consiguiente, la red con los números asociados a cada arco (factores de peso) es ideal para modelar esta situación.

Se quiere maximazar la cantidad de agua que fluye desde la fuente hasta su punto de consumo. Aún cuando la fuente puede producir cualquier cantidad de agua a un determinado flujo y el usuario es capaz de consumir agua en la misma proporción, el sistema de tubería puede que no tenga la capacidad suficiente para transportarla desde la fuente hasta el usuario. Esto es, los factores limitantes de todo el sistema es la capacidad de la tubería.

Otros problemas del mundo real son análogos en cuanto a su naturaleza, como por ejemplo, una red eléctrica, un sistema de transporte, un sistema de comunicaciones, o cualquier otro sistema de distribución en el cual uno desea maximizar la cantidad de algún elemento que es enviado desde un punto hasta otro.

III.4.1 TEOREMA DE FLUJO MAXIMO - CORTE MINIMO.

Un procedimiento natural para determinar el flujo máximo en una red es encontrar cadenas aumentantes en esta e incrementar el flujo a través de ellas lo más que se pueda. Sin embargo, se requiere una herramienta que indique cuándo se alcanza la optimalidad.

Esta herramienta es proporcionada por este teorema. Antes, consideremos lo siguiente:

Sean $R = [X,A,q]$ una red y $N \subset X$. Sea $\bar{N} = X - N$. Se denota con $(N,\bar{N})$ al conjunto de arcos que tienen un extremo en N y el otro fuera de N.

El conjunto de arcos $(N,\bar{N})$ es una cortadura de R si $s \in N$ y $t \in \bar{N}$, donde s y t son el origen y el destino de R. Obsérvese que si se remueve este conjunto de arcos ya no existe camino alguno de s a t

La capacidad de una cortadura $(N,\bar{N})$ es la suma de las capacidades de los arcos que la forman. Una cortadura mínima es aquella con mínima capacidad.

Existen ciertas relaciones entre cortaduras y flujos en una red. Una de ellas es la siguiente:

- Sea $R = [X,A,q]$ una red, sea f un flujo factible de valor v y sea $(N,\bar{N})$ una cortadura de R. Entonces $v \leq q(N,\bar{N})$.

En una red $R = [X,A,q]$ el valor del flujo máximo es igual a la capacidad de la cortadura mínima. Esto es, existe un flujo factible en R con valor igual a la capacidad de una cortadura de R.

Para ello, considérese cualquier flujo factible f en R y constrúyase una cortadura aplicando el siguiente procedimiento:

(i) Sea $N = s$

(ii) Si $i \in N$ y $f_{ij} < q_{ij}$ o $f_{ji} > 0$, agréguese j a N

Repítase (ii) hasta que no pueda agregarse vértice alguno a N.

Problema de Flujo Máximo.

Maximizar el flujo de N a N^- sobre todos los flujos x factibles respecto a las capacidades. El supremo en el problema de flujo máximo es la mínima cota superior del conjunto de valores de flujo de N a N^- de todos los posibles flujos x. Si este supremo es finito entonces el flujo x que produzca este valor es la solución requerida del problema de flujo máximo.

Como se puede observar, todas las trayectorias de N a N^- utilizan algún arco de cualquier corte $Q : N \rightarrow N^-$ y de este modo los cortes constituyen 'cuellos de botella' para el valor del flujo de N a N^-.

El problema de corte mínimo es el problema dual del flujo máximo. Supóngase que existe al menos un flujo X que satisface todas las restricciones del problema. Entonces:

$$\left[\begin{array}{l}\text{Supremo en problema}\\ \text{de flujo máximo.}\end{array}\right] = \left[\begin{array}{l}\text{Mínimo en problema}\\ \text{de Corte Mínimo.}\end{array}\right]$$

Cabe señalar que si existe una trayectoria de capacidad ilimitada, entonces ambos problemas tienen valor $+\infty$; en caso contrario, los valores de ambos, iguales, son finitos y el problema de flujo máximo tiene solución.

Una de las conclusiones que puede obtenerse del Teorema de Flujo Máximo - Corte Mínimo es que ambos problemas pueden ser resueltos simultáneamente. Como estos problemas son duales , si se tiene la solución de uno de ellos, a partir de ella puede obtenerse la solución del otro.

Para construir un flujo aumentante, se comienza con cualquier flujo que cumpla las restricciones y en cada iteración se buscará enviar más flujo de N a N^- a través de trayectorias aumentantes de flujo hasta que esto no sea posible. Si existe alguna trayectoria aumentante para el flujo X de capacidad ilimitada, el valor máximo de X no es finito, y por lo tanto, el supremo en el problema de flujo máximo es $+\infty$. Por otro lado, el criterio de terminación se establece mediante la existencia de un corte Q.

El algoritmo que resuelve los problemas de flujo máximo es el algoritmo de Ford - Fulkerson.

El algoritmo de Ford - Fulkerson determina el flujo máximo de N a N^- en una red $R = (X,A,q)$, en la cual no existen trayectorias de capacidad ilimitada.

ALGORITMO DE FORD Y FULKERSON.

Paso 1.
Iniciar con cualquier flujo factible f.

Paso 2.
Etiquetar en origen s con [s,A]

Paso 3.
Elegir un vértice etiquetado y no examinado; sea j éste vértice y sean [k, f(j)] sus etiquetas.

(i) A todo i ∈ $\bar{A}$ (j) que no esté etiquetado y tal que $f_{ji} < q_{ji}$ asignar la etiqueta [+j,f(i)],donde f(i) = min f(j), $q_{ji} - f_{ji}$.

(ii) A todo i ∈ $\bar{A}$ (j) que no esté etiquetado y tal que $f_{ij} > 0$ asignar la etiqueta [-j,f(i)], donde f(i)=min f(j), f_{ij} .

Se dice ahora que el vértice j ha sido examinado.

Paso 4.
Repetir el paso 3 hasta que suceda (i) o (ii):

(i) El vértice destino t no tiene etiqueta y todos los vértices etiquetados han sido examinados. Terminar ya que el flujo factible f es máximo.

(ii) El vértice t recibe etiqueta. Ir al paso 5.

Paso 5.
Sea x = t .

(i) Si la etiqueta de x es de la forma [+z,f(x)] hacer:

$$f_{zx} = f_{zx} + f(t)$$

(ii) Si la etiqueta de x es de la forma [-z,f(x)] hacer:

$$f_{xz} = f_{xz} - f(t)$$

Paso 6.

Si z = s, borrar todas las etiquetas y regresar al paso 2. Si z ≠ s, hacer x = z y regresar al paso 5.

Al terminar de aplicar el algoritmo t no tiene etiqueta; luego, la cortadura de capacidad mínima está dada por el conjunto de arcos:

$$(N,\bar{N}),\ \text{donde } N = \{ X \mid x \text{ no tiene etiqueta} \}$$

Ejemplo.

Un gran número de personas viajan en automóvil de la ciudad A a la ciudad B. Las rutas posibles se muestran en la red de la figura III.9.El departamento de policía de caminos desea construir suficientes casetas de inspección de tal manera que todo automóvil pase por lo menos una de ellas en su trayectoria de A a B. El costo de construcción de las casetas varía según su localización; el costo asociado con cada tramo se proporciona en cada arco de la red (en millones de pesos). Determínese dónde deberán colocarse las casetas si se desea incurrir en el mínimo costo.

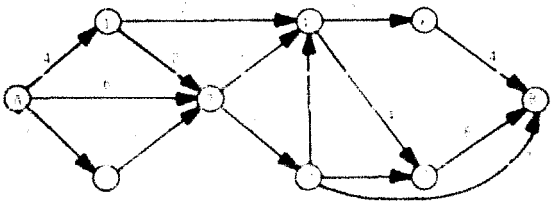


Figura III.9 Rutas posibles para viajar de la ciudad A a la ciudad B

Obsérvese que, en términos de redes, se desea obtener un conjunto de arcos (tramos donde deberán construirse las casetas) de manera tal que si se eliminan esos arcos de la red ya no existen caminos de A a B (es decir, todo automóvil debe utilizar alguno o algunos de estos arcos en su trayectoria); este conjunto de arcos forma entonces una cortadura de la red.

Por otro lado, se desea incurrir en el mínimo costo; debe determinarse entonces la cortadura de mínimo costo. Si se consideran los costos de los arcos como capacidades de un cierto flujo a través de ellos y se determina la cortadura de capacidad mínima, es claro que ésta corresponde a la de mínimo costo. Para ello se determinará el flujo máximo de A a B mediante el algoritmo de Ford y Fulkerson.

Iteración 1. Se aplicará el algoritmo con el flujo factible inicial de valor 11 definido en la siguiente red. Los números asociados a cada arco son el flujo a través de él y su capacidad. Las etiquetas asignadas a cada vértice durante esta iteración también se muestran en la red de la figura III.10.

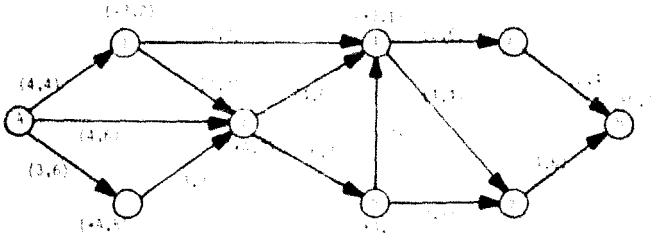


Figura III.10 Etiquetas asignadas a cada vértice en la iteración 1.

Puesto que f(B) = 2 existe una cadena aumentante de capacidad incremental igual a 2; ésta es: A, 1, 3, B. Actualizando el flujo a través de esta cadena se obtiene el definido en la siguiente red.

Iteración 2. De nuevo se asocia a cada vértice su correspondiente etiqueta, como se muestra en la figura III.11.

En esta iteración el vértice B no recibió etiqueta; por lo tanto este flujo de valor 13 es el máximo. El conjunto de vértices no etiquetados es $\bar{N} = \{2\}$ por lo que la cortadura mínima es $(N,\bar{N}) = (A,1), (A,3), (2,3)$ de capacidad $q(N,\bar{N}) = 4 + 6 + 3$ $q(N,\bar{N}) = 13$. Por lo tanto, la solución para el problema del departamento de policía de caminos es construir las casetas de inspección en los tramos (A,1), (A,3), (2,3) con un costo de construcción de 13 millones de pesos.

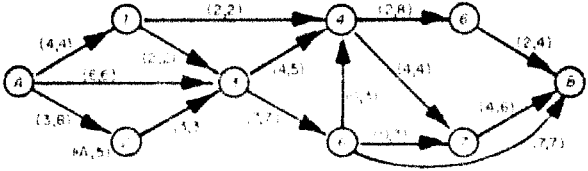


Figura III.11 Nueva Asociación de etiquetas a los vértices

III.4.2 FLUJO CON COSTO MINIMO

Considérese una red dirigida R, que consiste de m nodos y n arcos. Con cada nodo i en la red G se asocia un número b_i, que representa los recursos disponibles de un artículo (si $b_i>0$) o la demanda requerida del artículo (si $b_i<0$). Algunas veces, los nodos con $b_i>0$ se llaman orígenes, y los nodos con $b_i<0$ se llaman destinos. Si $b_i=0$, entonces ningún artículo está disponible en el nodo i y ninguno se requiere.

El problema de flujo con costo mínimo en una red se puede enunciar de la siguiente forma: Embárquense los recursos disponibles a través de la red, para satisfacer la demanda a un costo mínimo.

Matemáticamente, este problema resulta:

$$\text{Minimizar} \quad \sum_{i=1}^{m} \sum_{j=1}^{m} C_{ij} X_{ij}$$

$$\text{Sujeta a} \quad \sum_{j=1}^{m} X_{ij} - \sum_{k=1}^{m} X_{ki} = b_i \quad i = 1,\ldots,m$$

$$X_{ij} \geq 0 \qquad i,j = 1,\ldots,m$$

Las restricciones (sujeta a), se llaman ecuaciones de *conservación de flujo* o ecuaciones de Kirchoff e indican que, en la red, no se puede crear ni destruir flujo.

En las ecuaciones de conservación:

$\sum_{j=1}^{n} X_{ij}$ Es el flujo total que sale del nodo i.

$\sum_{k=1}^{n} X_{ki}$ Es el flujo total que entra al nodo i.

Si $b_i < 0$, entonces el flujo que entra a i debe ser mayor que el que sale de i.

El problema de flujo con costo mínimo se puede originar, por ejemplo, en una red logística en la que los hombres y materia se mueven entre varios puntos de la tierra. También puede estar asociado con el movimiento de locomotoras entre puntos en una red de ferrocarriles con el fin de proporcionar energía al menor costo de viaje.

Problemas de flujo con costo mínimo en redes ocurren en el análisis y diseño de sistemas de comunicaciones, sistemas de oleoductos, programación de tanques, y muchas otras áreas.

El problema del flujo a costo mínimo en una red R=[X,A,q,c] puede resolverse de dos maneras. Una de ellas involucra la determinación de circuitos negativos en la red marginal y la otra involucra la determinación de rutas más cortas en esta red.

El primero de estos procedimientos será llamado algoritmo basado en la eliminación de circuitos negativos y algunas veces se hace referencia a él como algoritmo Primal puesto que empieza a aplicarse a partir de un flujo factible del valor v requerido y, en cada iteración, este flujo se mejora, sin modificar su valor, hasta alcanzar la optimalidad.

El segundo procedimiento será llamado algoritmo basado en rutas más cortas; a veces, en la literatura, se hace referencia a él como algoritmo Dual ya que se aplica a partir de un flujo factible de algún valor menor del valor requerido pero de costo mínimo; en cada iteración se incrementa el valor del flujo, siempre conservando la optimalidad, hasta alcanzar v.

Método Basado en la Eliminación de Circuitos Negativos.

El primer paso de este método consiste en determinar un flujo factible del valor v requerido. Para ello puede utilizarse el algoritmo de Ford y Fulkerson con una modificación en el criterio

de alto. Esta modificación consiste en lo siguiente: si, en el paso 5,la etiqueta f(t) de t más el valor de v' del flujo actual es mayor o igual que v, entonces se modifica el flujo en la cantidadd v - v' y se termina con el flujo del valor v; si éste no es el caso se realiza otra iteración. Obsérvese que si se determina el flujo máximo, utilizando esta modificación del algoritmo de Ford y Fulkerson, el valor de éste es menor que v, entonces el problema no tiene solución.

Una vez determinado el flujo de valor v se procede a 'probar' la optimalidad. La optimalidad consiste en verificar si en la red marginal existen circuitos negativos o no; si no existen, el flujo es de costo mínimo; si existe alguno, éste será 'eliminado'.

Dicho en otras palabras: si existe un circuito negativo en la red marginal, se determina su capacidad incremental y se modifica el flujo a través del ciclo correspondiente, en la red original, en esta cantidad para obtener un flujo de menor costo. Puesto que el nuevo flujo a través de alguno de los arcos del ciclo es igual a su capacidad o a cero, el circuito negativo no estará en la red marginal con respecto a éste nuevo flujo; es decir, el circuito fue 'eliminado'.

Este procedimiento será aplicado hasta que se hayan 'eliminado' todos los circuitos negativos. En este caso el flujo de valor v actual es de costo mínimo. Nótese que, durante este procedimiento, es necesario determinar circuitos negativos, para determinarlos puede utilizarse el algoritmo de Floyd o el algoritmo general para determinar arborescencias de rutas más cortas si es que existe alguna raíz.

En efecto, recuérdese que durante el algoritmo de Floyd se determinan las rutas más cortas entre todo par de vértices, si existen, o se determina algún circuito negativo concluyéndose, en este caso, que no hay solución al problema.

Es posible, entonces, emplear este método para detectar los circuitos negativos en la red marginal o para garantizar la no existencia de éstos en el caso de que el algoritmo termine con las rutas más cortas.

A continuación se presenta detatalladamente el algoritmo basado en la eliminación de circuitos negativos.

ALGORITMO BASADO EN LA ELIMINACION DE CIRCUITOS NEGATIVOS.

Este algoritmo determina el flujo a costo mínimo de valor v en la red R = (X,A,q,c).

Paso 1.

Determínese un flujo factible f de valor v mediante el algoritmo de Ford y Fulkerson.

Paso 2.

Constrúyase la red marginal, con respecto a f, $R'(f)$.

Paso 3.

Mediante algún algoritmo de rutas más cortas, identifíquese algún circuito negativo en $R'(f)$. Si no existen circuitos negativos, terminar. El flujo actual f es el requerido; en otro caso sea C el circuito negativo. Ir al paso 4.

Paso 4.

Sea $d = \text{Min}_{(i,j) \in C} q_{ij}$

(i) Para todo arco $(i,j) \in A$ tal que $(i,j] \in A_1$ C actualizar.

$$f_{ij} = f_{ij} + d$$

(ii) Para todo arco $(i,j) \in A$ tal que $(j,i) \in A_2$ C actualizar.

$$f_{ji} = f_{ij} - d$$

Con este nuevo flujo ir al paso 2.

Ejemplo.

Determínese el flujo a costo mínimo de valor 5 en la red que se muestra en la figura III.12

Iteración 1. Se aplicará el algoritmo a partir del flujo, de costo 44. Este flujo se muestra en la figura III.13. La red marginal con respecto a este flujo se muestra en la figura III.14. Utilizando el algoritmo de Floyd, se determina el circuito negativo 1,s,2,3,1 de costo -2. De aquí se tiene $d = \text{Min}\ q_{1s}, q_{s2}, q_{23}, q_{31} = 1$. Actualizando el flujo se obtiene el definido en la figura III.15.

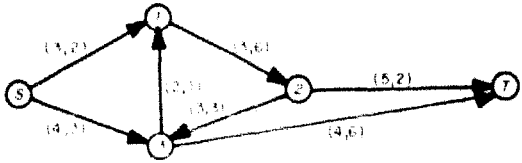


Figura III.12 Red para determinar Flujo a Costo Mínimo

Iteración 2. La red marginal con respecto a este nuevo flujo se muestra en la figura III.16. al utilizar el algoritmo de Floyd se observa que la red marginal no contiene circuitos negativos por lo que el flujo de costo 42 definido en la figura 4 es el flujo a costo mínimo de valor 5.

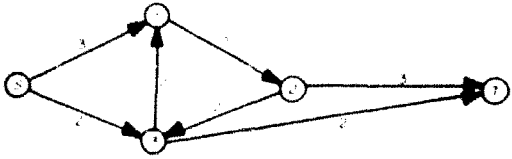


Figura III.13 Flujo de Valor 5 de Costo 44

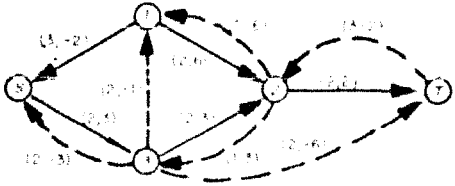


Figura III.14 Red Marginal con Respecto al Flujo definido en la Figura III.13.

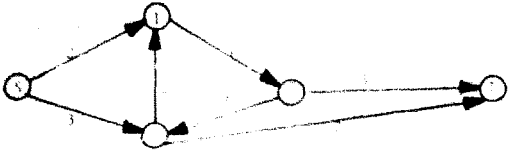


Figura III.15 Nuevo Flujo de Valor 5 de Costo 42 (Flujo Optimo)

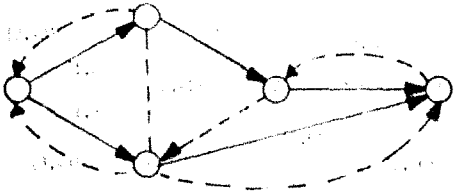


Figura III.16 Red Marginal con Respecto al Flujo Definido en la Figura III.15.

Método Basado en Rutas Mas Cortas.

Primeramente se determinará un flujo factible de valor v', de costo mínimo, donde v' es menor que el valor v requerido. En general se utiliza v' = 0. Para determinar este flujo deben eliminarse todos los circuitos negativos que existan en la red marginal. El siguiente paso consiste en determinar la ruta más corta P, de s a t, en la red marginal; de nuevo es posible utilizar, con este propósito, el algoritmo general para determinar arborescencias de rutas mas cortas. A través de la cadena correspondiente a P se enviará la máxima cantidad posible de s a t; el nuevo flujo así definido será de costo mínimo.

Se procederá de esta forma, en cada iteración, hasta alcanzar el valor v requerido. Si en alguna iteración no existe ruta alguna de s a t en la red marginal y el valor v' del flujo actual f es menor que v entonces no existe ningun flujo del valor requerido ya que, al no existir ruta de s a t en la red marginal, no existen cadenas aumentantes de s a t en la red original y por tanto f es flujo máximo.

Por otro lado, si v' más la capacidad incremental de la cadena encontrada es mayor que v entonces sólo será necesario enviar la cantidad v - v' a través de esta cadena para determinar el flujo definido.

En el algoritmo se denotará con f+k o (P^*) al flujo definido en R de la siguiente manera, donde P es la cadena correspondiente a P^*:

$$f_{ij}^{*} \begin{cases} f_{ij} & \text{si } (i,j) \notin P \\ f_{ij} + k & \text{si } (i,j) \in A_z \quad P^* \\ f_{ij} - k & \text{si } (j,i) \in A_z \quad P^* \end{cases}$$

ALGORITMO BASADO EN RUTAS MAS CORTAS.

Este algoritmo determina el flujo a costo mínimo del valor v en la red R = [X,A,q,c].

Paso 1.
Determínese un flujo factible f de costo mínimo de valor 0 en R.

Paso 2.
Constrúyase la red marginal, con respecto a f, R'(f).

Paso 3.
Determínese la ruta más corta P^* de **s** a **t** en R'(f).

Paso 4.
Sea $d = \min_{(i,j) \in P^*} q_{ij}$

Paso 5.

(i) Si el valor del flujo f+d o (P^*) es menor que v, actualizar f = f + d o (P^*) e ir al paso 2.

(ii) Si el valor del flujo f+d o (P^*) es igual que v, actualizar f = f + d o (P^*) y terminar. En este caso f es el flujo a costo mínimo de valor v.

(iii) Si el valor del flujo f+d 0 (P^*) es mayor que v y v' es el valor de f, actualizar f = f (v-v') o (P^*) y terminar, ya que f es el flujo requerido.

Ejemplo:

Considere la red de flujo figura III.17:

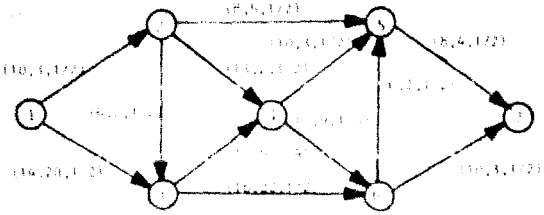


Figura III.17

Determine el flujo máximo a costo mínimo.

Iteración 1. El flujo cero en cada arco es óptimo cuando se demanda un flujo igual a cero en el nodo sumidero denotado por el nodo 7. La red marginal es en este caso la red original y el árbol de rutas más cortas se muestra en la figura III.18.

De dicho árbol se observa que la ruta más corta de 1 a 7 está dado por los arcos (1,2), (2,5), y (5,7), y que el flujo máximo que llega al nodo sumidero es:

$$AF_T = \min \quad 10a_1^*,\ a_5^*\ a_{10}^*\ ,\ 8a_5^*\ a_{10}^*\ ,\ 8\ a_{10}^*$$

$$AF_T = \min \quad 10\left(\tfrac{1}{4}\right)\ \left(\tfrac{1}{3}\right)\ \left(\tfrac{1}{2}\right),\ 8\left(\tfrac{1}{3}\right)\ \left(\tfrac{1}{2}\right),\ 8\left(\tfrac{1}{2}\right) \quad = \frac{10}{24}$$

Asimismo el flujo en cada arco es:

$$f_1 = AF_T/a_1^*\ a_5^*\ a_{10}^* = 10$$

$$f = AF\ /a^*\ a^* \qquad = 5/2$$

$$f_{10}^* = AF_T/a_{10}^* \qquad = 5/6$$

y cero en el resto de los arcos.

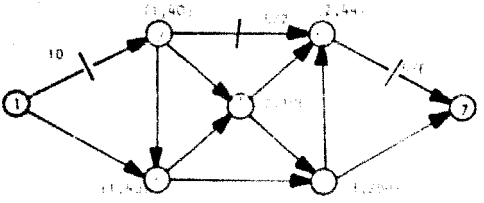


Figura III.18 Arbol de Rutas más cortas

Iteración 2. La red marginal asociada al flujo óptimo se muestra en la figura III.19:

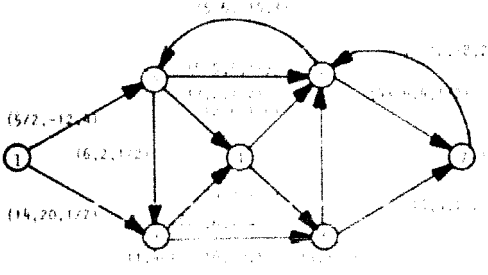


Figura III.19 Red Marginal Asociada al Flujo Optimo

El árbol de rutas cortas es dado por la figura III.20, y el flujo máximo aumentante es:

$$AF_T = \min \left\{ 14\left(\frac{1}{2}\right), \left(\frac{1}{5}\right), \left(\frac{1}{2}\right), 16\left(\frac{1}{6}\right), \left(\frac{1}{2}\right), 10\left(\frac{1}{2}\right) \right\} = \frac{7}{10}$$

El nuevo flujo en la red es dado en la figura III.21. Usando la red marginal es inmediato que dicho flujo es el máximo flujo a costo mínimo (pues los arcos (1,2) y (1,3) están en su cota mínima).

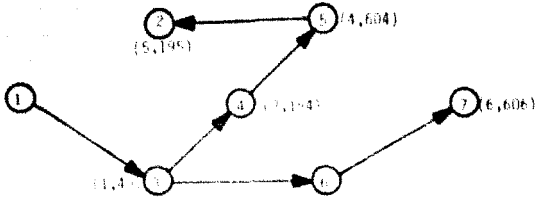


Figura III.20 Arbol de Rutas Cortas

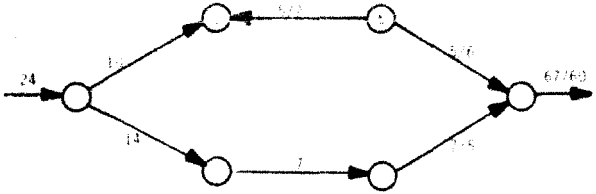


Figura III.21 Nuevo Flujo Aumentante

III.5 ARBOL DE EXPANSION MINIMA

La teoría de gráficas es una de las áreas más elegantes de las matemáticas, muy atractiva para la mente lógica, muchas de las ideas y métodos que maneja parecen obvios; sin embargo un descubrimiento original requiere esfuerzo. La mayor parte de los problemas reales a los que se puede aplicar la teoría de gráficas involucran una buena cantidad de datos por lo que resulta útil auxiliarse de la computadora para poderlos resolver.

La primera contribución importante a las teorías del árbol de expansión mínima fue hecha por KRUSKAL en 1956 que presentó un algoritmo para encontrar un árbol de expansión mínima. En 1957, PRIM propuso un nuevo algoritmo con algunas ventajas sobre el anterior.

El árbol de expansión mínima es un problema simple en teoría de gráficas, pero con una gran aplicación práctica en áreas tan diversas como: redes de intalaciones eléctricas o redes de comunicación y transporte, problemas de confiabilidad, redes de tensión mínima, análisis de flujo de redes multiterminales y la solución de problemas del viajero.

Para cada red pueden conectarse varios árboles de expansión distintos.

Considérese la gráfica de la figura III.23

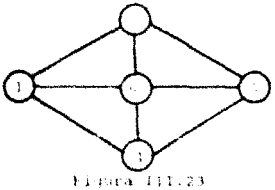
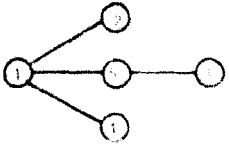
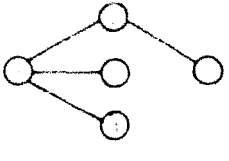


Figura III.23

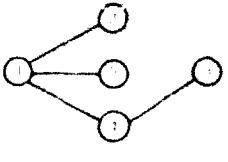
Los árboles de expansión de la red de la figura III.23 son los siguientes:



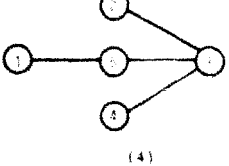
(1)



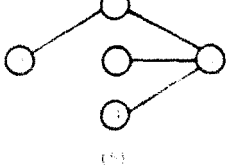
(2)



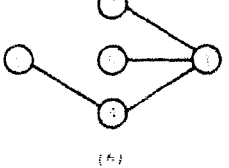
(3)



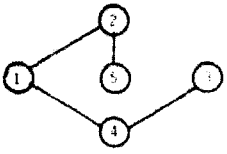
(4)

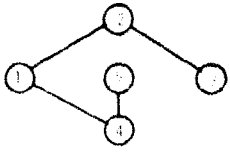


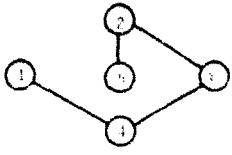
(5)

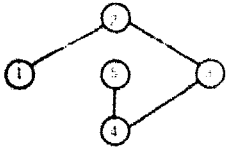


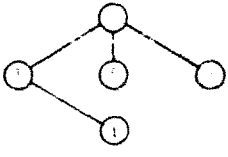
(6)

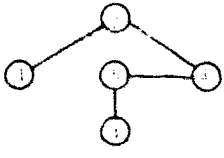

(7)

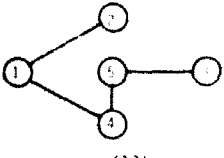

(8)

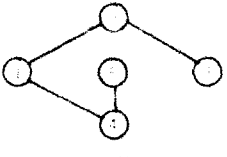

(9)

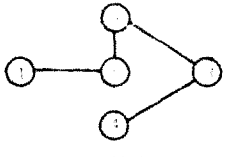

(10)

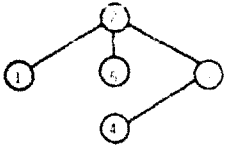

(11)

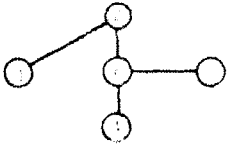

(12)

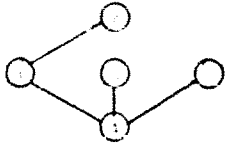

(13)

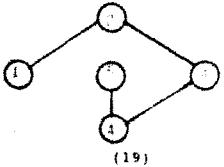

(14)

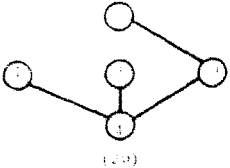

(15)

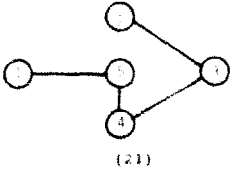

(16)

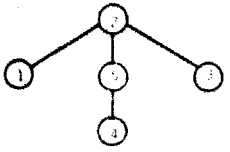

(17)

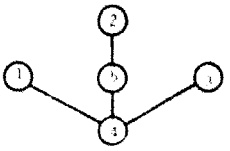

(18)

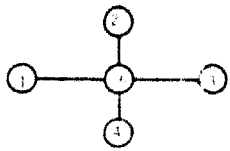

(19)

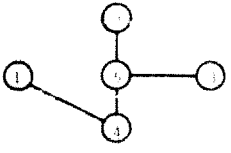

(20)

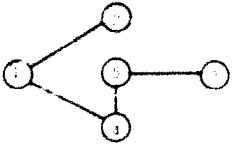

(21)

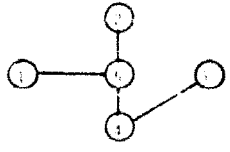

(22)

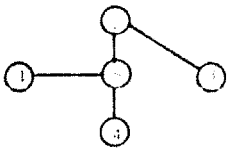

(23)

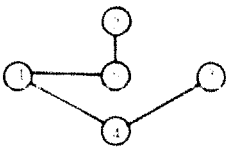

(24)

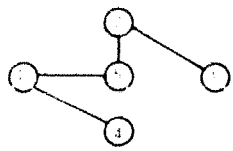

(25)

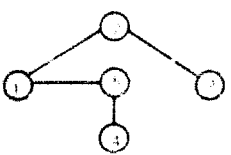

(26)

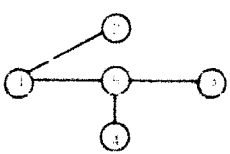

(27)

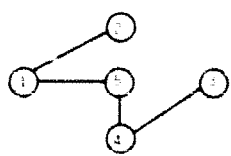

(28)

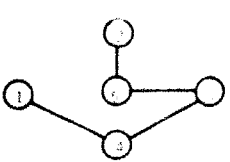

(29)

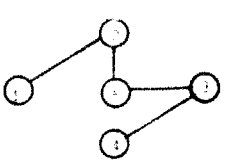

(30)

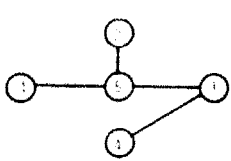

(31)

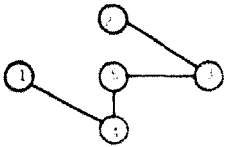

(32)


(33)

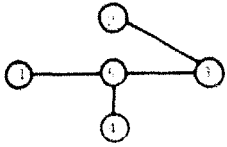

(34)


(35)


(36)



(17)



(18)

Cuando se tiene una función c : A ---> R de los arcos de una gráfica conectada G a los números reales que asigna a cada arco un atributo como su longitud, su costo, su peso, etc.; es posible definir el valor del atributo que corresponde a la gráfica sumando los valores asignados a cada uno de los arcos que lo forman.

En este caso se puede hablar de un árbol de interés especial llamado árbol de expansión mínima, que es el árbol con menor atributo.

Este tipo de problemas de optimización tiene aplicación, fundamentalmente, en las redes de comunicación eléctrica, telefónica, telegráfica, carretera, ferrocarrilera, aérea, marítima, etc.,en las cuales los nodos representan, por ejemplo, puntos de consumo o terminales y los arcos, líneas de alta tensión, líneas telefónicas o telegráficas, carreteras, vías aéreas, de ferrocarril, etc.

Considerando la gráfica de la figura III 24, a la que se le ha agregado la anotación de los valores de la función longitud sobre cada uno de sus arcos.

Las longitudes correspondientes a los distintos árboles de expansión se pueden obtener en la tabla 1.1

Como puede observarse para este caso, el árbol de expansión mínima es el árbol numero 12.

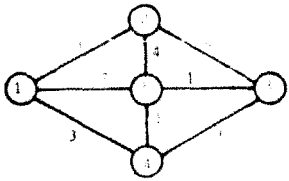


Figura III.24

Arbol	Longitud	Arbol	Longitud
1	9	20	12
2	13	21	11
3	11	22	13
4	11	23	11
5	12	24	8
6	12	25	9
7	13	26	8
8	12	27	10
9	15	28	12
10	17	29	12
11	12	30	14
12	10	31	11
13	8	32	7
14	12	33	9
15	14	34	11
16	14	35	11
17	9	36	10
18	10	37	10
19	12	38	9

Tabla 3.1

III.5.1. DESCRIPCION DEL PROBLEMA.

Considérese el siguiente problema: En un lago hay n islas, denotadas x_1, x_2, ..., x_n, y se desea construir puentes para comunicarlas. La construcción del puente (x_i, x_j) cuesta c_{ij} pesos.

El problema consiste en determinar donde construir los puentes de tal manera que cada par de islas queden conectadas por medio de éstos y que el costo total de construcción sea mínimo.

Sea G = (X, A) una gráfica, no-dirigida, donde el conjunto de vértices de X representa al conjunto de islas y cada elemento (x_i,x_j) del conjunto de aristas A, representa la posible construcción de un puente entre las islas x_i y x_j.

Sea c una función que asocia, a cada elemento de A, el costo de construcción del puente respectivo. Obsérvese que una solución para este problema es una gráfica parcial T = (X, A') de G.

Esta gráfica parcial deberá cumplir los tres puntos siguientes:

a) T es conexa, puesto que se desea que exista una cadena que una a todo par de vértices.

b) T no deberá tener ciclos puesto que, de ser así, se incurriría en un costo innecesario.

c) El costo de T deberá ser mínimo.

En base a lo anterior, se define lo siguiente:

1) Un árbol es una gráfica T = (X, A) conexa y acíclica. La gráfica de la figura III.25 es un árbol.

2) Sea G = (X, A) una gráfica no dirigida. Un árbol expandido de G es una gráfica parcial T = (X,A') de G, que es un árbol. Como ejemplo considérese la gráfica de la figura III.26(a); un árbol expandido de ella, figura III.26(b); y un árbol no expandido de ella, figura III.26(c).

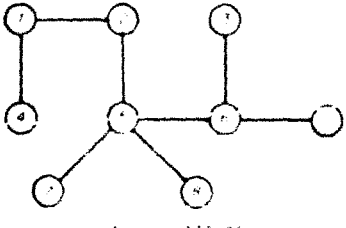


Figura III.25

(a) (b)

Figura III.26

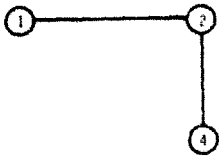


Figura III.26 (c)

Se observa, entonces que, la gráfica T que será una solución para el problema de las islas debe ser un árbol expandido de G. Por otro lado, se observa que una gráfica puede tener varios árboles *expandidos*.

Por esta razón, existen diferentes alternativas de solución para el problema de las islas; sin embargo, se tiene interés en la mejor de todas ellas, es decir, aquella con costo total de construcción mínimo. Se definirá, entonces el costo de un árbol.

Considérese una función p que asocia un real a cada arista de una gráfica. A este real se le llama peso de la arista. La función puede representar costo, distancia, tiempo, etc. En el caso del problema de las islas la función p fué denotada con la letra c y representa el costo de la construcción de un puente.

El peso de un árbol es la suma de los pesos de las aristas que lo forman.

De acuerdo a los conceptos anteriores, es sencillo concluir que la solución óptima al problema de las islas está dada por el árbol expandido de peso mínimo asociado a la gráfica G.

III.5.2 CONCEPTOS UTILES.

Existen distintas definiciones de Árboles todas ellas equivalentes entre sí. Con el propósito de demostrar tales equivalencias se empezará a analizar ciertas propiedades elementales de una gráfica con varias componentes conexas.

Sea G = (X, A) una gráfica. Supóngase que se agrega la arista a = (x,y) a la gráfica G. Entonces:

a) El número de componentes conexas de G disminuye una unidad si los extremos de a pertenecen a dos componentes conexas distintas. En este caso, la arista a no pertenece a ningún ciclo de la gráfica G' = (X, A\U a).

b) El número de componentes conexas de G permanece igual si los extremos de a pertenecen a la misma componente conexa. En este caso, a pertenece a un ciclo de la gráfica G'=(X,A U a)

Ahora, sea G = (X, A) una gráfica con n vértices y m aristas. Entonces:

a) Si G es conexa entonces m ≥ n-1.

b) Si G es acíclica entonces m ≤ n-1.

Teorema. Sea G = (X, A) una gráfica con n vértices. Supóngase que n ≥ 2. Los postulados siguientes son equivalentes y caracterizan un árbol:

a) G es conexa y acíclica

b) G es acíclica y tiene n-1 aristas.

c) G es acíclica y si se agrega una arista se forma exactamente un ciclo.

d) G es conexa y tiene n-1 aristas.

e) G es conexa pero deja de serlo si se elimina una arista.

f) Existe, en G, una única cadena entre todo par de vértices.

III.5.3 METODOS DE SOLUCION.

Existen varios métodos para encontrar el árbol de expansión mínima; el más obvio consiste en comparar el valor del atributo asociado con todos y cada uno de los árboles de expansión de la red con que se está trabajando y elegir de entre ellos el menor, pero este método requiere demasiado esfuerzo cuando el número de árboles es grande.

El método de J.B. KRUSKAL, consistente en partir el arco con el atributo de menor valor e ir agregando arcos sucesivos hasta formar el árbol de expansión mínima. Estos arcos son seleccionados, tomando el de menor atributo de entre los restantes que no formen circuitos con los ya incluidos. Este método da lugar a la formación de varios subárboles que van creciendo sumultáneamente hasta que lleguen a unirse para formar el árbol de expansión.

Este algoritmo es también conocido como el algoritmo "glotón". Este método consiste en ordenar las aristas en orden ascendente de peso. Las aristas se irán examinando en el orden establecido y serán consideradas en el árbol si no forman ciclo con las anteriormente consideradas; de este modo se obtendrá una gráfica acíclica.

El algoritmo termina cuando el número de aristas consideradas sea igual al número de aristas consideradas sea igual al número de vértices menos uno garantizado, de esta manera, la generación de un árbol expandido de G. Obsérvese que, en cada iteración, la gráfica formada por las aristas consideradas y sus extremos no necesariamente es conexa, excepto en la última iteración.

Debe también notarse que el número de iteraciones será, al menos, el número de vertices menos uno (en el caso de que las primeras n-1 aristas en el orden establecido no formen ciclo) y, a lo más, el número de aristas.

ALGORITMO DE KRUSKAL.

Este algoritmo determina el árbol de expansión cuyo "costo o peso" es mínimo en una gráfica conectada G = [X, A] con n vértices y función de costos c : A $\longrightarrow$ R conocida.

Paso 1.

(Iniciar)
Ordene el conjunto de aristas en forma creciente respecto a la función de costos. Sean $a_1, a_2, \ldots, a_m$ las aristas ordenadas. Hacer k = 0, j = 1, A' = $\emptyset$.

Paso 2.

(Añadir aristas)
Si la arista a_j no forma ciclo con el conjunto de aristas de A' entonces A' = A' U a_j y hacer k:= k+1 e ir al paso 3. De otra manera continuar con el paso 3.

Paso 3.
(Criterio de terminación).
Si k < n-1 hacer j:=j+1 y regresar al paso 2.
En caso contrario T = [X, A'] es el árbol de expansión mínima.

Ejemplo:

Considérese la red de la figura III.27 y determínese el árbol de peso mínimo.

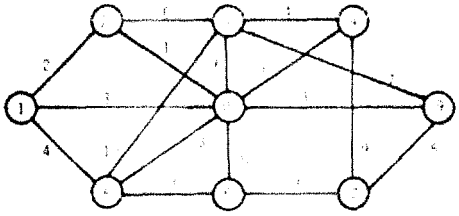


Figura III.27

Iteracion 1. Se ordenan las aristas:

$a_1=(2,3)$, $a_2=(2,8)$, $a_3=(1,2)$, $a_4=(8,4)$, $a_5=(1,8)$,

$a_6=(5,8)$, $a_7=(8,9)$, $a_8=(1,5)$, $a_9=(3,4)$, $a_{10}=(5,6)$,

$a_{11}=(7,9)$, $a_{12}=(3,8)$, $a_{13}=(6,7)$, $a_{14}=(3,9)$, $a_{15}=(8,6)$,

$a_{16}=(4,7)$, $a_{17}=(3,5)$

$A' = a_1$; k=1. Puesto que $k < n-1 = 8$, aún no se tiene el árbol.

En la tabla 3.2 se presenta un resumen de las operaciones realizadas en cada iteración.

En la primera columna aparece el numero de iteración; en la segunda, la arista que se agrega al conjunto A' para ir formando el árbol; por último, en la tercera columna se tiene el número de aristas que ya se han incluido en el árbol hasta esa iteración.

Número de iteración (j)	Arista agregada a A'	Valor de k
1	a_1 = (2,3)	1
2	a_2 = (2,8)	2
3	a_3 = (1,2)	3
4	a_4 = (8,4)	4
5	ninguna (a_5 forma un ciclo)	4
6	a_6 = (5,8)	5
7	a_7 = (8,9)	6
8	ninguna (a_8 forma un ciclo)	6
9	ninguna (a_9 forma un ciclo)	6
10	a_{10} = (5,6)	7
11	a_{11} = (7,9)	8

Tabla 3.2

En la iteración 11 se tiene que k = 8 = n-1, se ha obtenido la solución óptima; ésta está dada por la gráfica T = [X, A'], donde A' = $a_1, a_2, a_3, a_4, a_6, a_7, a_{10}, a_{11}$, el cual se observa en la figura III.28.

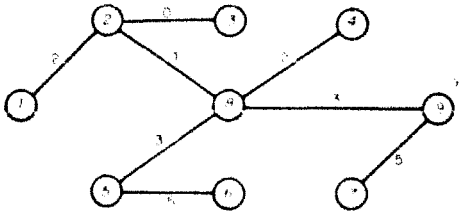


Figura III.28

Este árbol expandido tiene un peso de:

0 + 1 + 2 + 2 + 3 + 3 + 5 + 5 = 21

METODO ALTERNATIVO DE SOLUCION

El método alternativo de solución para el problema del árbol de peso mínimo de una red conexa con n vértices, es el algoritmo de Prim. El método propuesto por R.C. Prim (1957), construye el árbol de expansión mínima a partir de un nodo arbitrario para formar un árbol parcial al que se le agrega el arco con menor atributo para formar un árbol.

Este algoritmo consiste en considerar, inicialmente, una gráfica formada por cualquier vértice de la gráfica; después de agregará la arista de menor peso adyacente a él y su extremo. Luego se aumenta la arista más pequeña, que tenga exactamente un extremo en la gráfica formada, junto con su otro extremo. Se procede de esta manera, sucesivamente, hasta tener n-1 aristas en la gráfica generada.

A diferencia del algoritmo de Kruskal, la gráfica construida en cada iteración es conexa. En particular, la última gráfica es un árbol expandido de la gráfica original. Debe notarse también, que el algoritmo termina en n-1 iteraciones exactamente; esto constituye otra diferencia con el algoritmo de Kruskal.

ALGORITMO DE PRIM

Este algoritmo determina el árbol de expansión cuyo "peso o costo" es mínimo en una gráfica conectada $G = (X, A)$ con n vértices y función de peso o costo $C : A \rightarrow R$ conocida.

Paso 1.

(Iniciar)

Sea x_0 (arbitrario) elemento de X y k=0. Sea

$$X_0 = x_0 \quad ; \quad A_0 = \emptyset$$

Paso 2.

(Añadir arista).

Sea F_k el conjunto de aristas de A que tienen exactamente un extremo en A_k. Sea a_k la arista de costo mínimo en F_k y denote por x_k el extremo de a_k que no pertenece a F_k. Hacer:

$$X_{k+1} := X_k \cup x_k \quad ; \quad A_{k+1} := X_k \cup a_k$$

Paso 3.

(Criterio de terminación).
Hacer k:=k+1. Si k < n-1 regresar al paso 2. En caso contrario terminar. La gráfica $T_{n-1} = [X_{n-1}, A_{n-1}]$ representa el árbol de expansión mínima de G.

Ejemplo:

Considérese la red de la figura III.29. Determínese el árbol expandido de peso mínimo mediante el algoritmo de Prim. Se empezará a aplicar el algoritmo definiendo $X_0 = 7$.

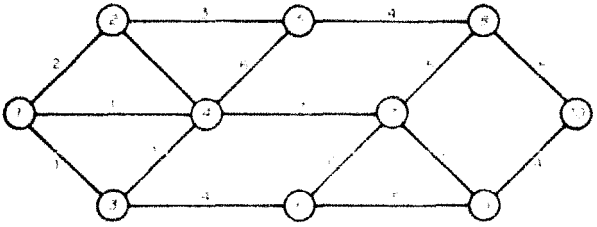


Figura III.29

En el cuadro 3.1 se presentan, resumidas, las iteraciones delalgoritmo.

Del cuadro 3.1, en la primera columna aparece el número de iteración (k), en la segunda el conjunto C_k resultante de la k-ésima iteración, en la tercera la arista de peso mínimo de C_k (a_k), y, por último, en las columnas cuarta y quinta respectivamente, los vértices y las aristas que irán formando el árbol de expandido de peso mínimo de la gráfica.

En la iteración k = n - 1 = 9 puede observarse el árbol expandido de peso mínimo generado por el algoritmo , el cual se muestra en la figura III.30.

Iteración (k)	C_k	a_k	X_k	A_k
1	(4,7),(6,7), (7,8),(7,9)	(6,7)	7,6	(6,7)
2	(3,6),(6,9), (4,7),(7,8) (7,9)	(3,6)	7,6,3	(6,7),(3,6)
3	(1,3),(3,4) (6,9),(4,7), (7,8),(7,9)	(1,3)	7,6, 3,1	(6,7),(3,6), (1,3)
4	(1,2),(1,4) (3,4),(6,9), (4,7),(7,8) (7,9)	(1,4)	7,6, 3,1, 4	(6,7),(3,6), (1,3),(1,4)
5	(1,2),(2,4) (4,5),(6,9), (7,8),(7,9)	(1,2)	7,6, 3,1, 4,2	(6,7),(3,6), (1,3),(1,4), (1,2)
6	(2,5),(4,5) (6,9),(7,8), (7,9	(2,5)	7,6, 3,1, 4,2 5	(6,7),(3,6), (1,3),(1,4), (1,2),(2,5)
7	(5,8),(6,9), (7,8),(7,9)	(5,8)	7,6, 3,1, 4,2, 5,8	(6,7),(3,6), (1,3),(1,4) (1,2),(2,5), (5,8)
8	(6,9),(7,9), (8,10)	(6,9)	7,6, 3,1, 4,2, 5,8, 9	(6,7),(3,6), (1,3),(1,4) (1,2),(2,5), (5,8),(6,9)
9	(9,10),(9,10(	(9,19)	7,6, 3,1, 4,2, 5,8, 9,10	(6,7),(3,6), (1,3),(1,4) (1,2),(2,5), (5,8),(6,9), (9,10)

Tabla 3.3

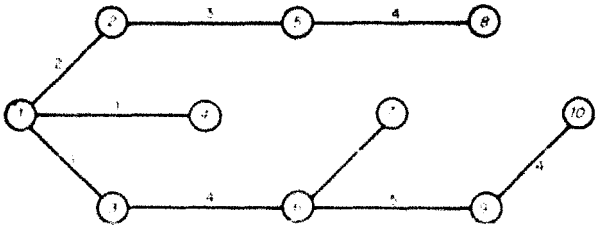


Figura III.30

cuyo peso es: 2 + 4 + 1 + 1 + 2 + 3 + 4 + 5 + 4 = 26

CAPITULO

CUATRO

SISTEMA DE SOFTWARE

IV.1 CONCEPTOS GENERALES.

El propósito de esta sección es mostrar un panorama general de lo que es la Ingeniería de Software, además de dar una definición y descripción de sus características. La definición que se establece aquí es el resultado de muchos estudios del desarrollo del software de diferentes metodologías y problemas.

IV.1.1 LA CRISIS DEL SOFTWARE.

El manejo de la información por medio de computadoras se hace día con día más frecuente, sin embargo el diseño de sistemas eficientes y adecuados para la manipulación de información no resulta fácil. En ocasiones el analista se enfrenta a problemas de diseño, el programador a problemas de codificación y el usuario se enfrenta con un sistema deficiente, esto se debe en gran medida a la falta de uso de una técnica bien definida para el diseño de sistemas.

Las técnicas empleadas son diversas, pero todas pretenden llegar a lo mismo: desarrollar sistemas que sean eficientes, fácilmente implementables y bien documentados. Dichas técnicas se agrupan en la que se conoce como Ingeniería de Software o Ingeniería de Programación.

La ingeniería de programación es una disciplina que hasta la década de los 60's aún no se había establecido, surgió como una solución a lo que se conoce como Crisis del Software.

La problemática principal que se detectó en dicha crisis se pueden definir en los siguientes puntos:

a) El costo del software se incrementa en forma exponencial.

b) Los proyectos no se terminaban a tiempo, si aún con el presupuesto programado.

c) El mantenimiento de los sistemas absorvía la mayor parte de los recursos de la gente de desarrollo.

Los problemas anteriores a su vez, generan otros problemas, como:

1) Insatisfacción del usuario con el sistema.
2) Calidad dudosa del software.
3) Dificultad para la realización del mantenimiento.

IV.1.2 INGENIERIA DE SOFTWARE

Han existido una diversidad de definiciones de la ingeniería de software y la más aceptada es la siguiente:

> *"La Ingeniería de Software es el establecimiento y uso de metodos de ingeniería con el fin de obtener software rentable y funcional, y a su vez generar sistemas de computo eficaces, confiables, transportables y rentables".*

APLICACIONES DEL SOFTWARE.

Las aplicaciones del software van de acuerdo al área en que van a ser utilizadas.

Sistema Software. El sistema sofware es un conjunto de programas escritos pra servicio de otros programas por ejemplo compiladores, editores y manejo de archivos, que son procesos complejos que determinana las estructuras de la información. Este tipo de sistemas se caracterizan por la gran interacción que tiene con el hardware, por el uso de múltiples usuarios y por la necesidad de un proceso sofisticado en su manejo.

Software en los Negocios. El software empleado en esta rama involucra "sistema discretos" como lo son inventarios, cuentas por pagar, ahorro, cuentas por cobrar, etc.; los cuales se caracterizan po el manejo de sistemas de información (MIS), los cuales facilitan la reestructuración de los datos existentes, el manejo de las decisiones y otros aspectos.

Software en la Ciencia y la Ingeniería. El software usado en las ciencias y en la ingeniería comprende un gran rango de aplicaciones caracterizándose por la innovación de algoritmos.

Software Combinacional. Hace uso de los algoritmos no numéricos para la solución de problemas complejos que requieren de Inteligencia Artificial. El reconocimiento de patrones de imágenes y de voz, los juegos, los cuestionarios, etc.; son algunos de los problemas que están implícitos en estas técnicas.

IV.1.3 CICLO DE VIDA DE UN SISTEMA DE PROGRAMACION.

Existe un número considerable de metodologías de desarrollo pero en todas se pueden distinguir las siguientes fases:

Estudio del Sistema.

Se refiere a una revisión general del sistema actual, de la que se deriva la detección y definición de necesidades, se plantean diferentes alternativas para satisfacer las necesidades. Se realiza un estudio de factibilidad que incluye una revisión de los recursos para determinar si es factible su utilización y un análisis beneficio/costo. El estudio de factibilidad se hace generalmente para todas las alternativas propuestas, ya que de este estudio de factibilidad se hace generalmente para todas las alternativas propuestas, ya que la realización del sistema no es factible, y ya no se continuaría con las siguientes etapas.

Planeación.

Una vez que se definió que se necesita el nuevo sistema y que es factible llevarlo a cabo, se hace una planeación de como se desarrollará el mismo. Primero se definen los alcances del proyecto. Se determinan también la disposición de los recursos cronológicamente y se establecen mecanismos de supervisión y control de avance.

Con respecto a las herramientas de control del proyecto, se tienen las opciones del gráfico de Gantt, los cuales consisten de un reporte en forma de tabla donde se incluye la información del avance,actividades, etc.; la otra opción es la realización de la Ruta Crítica, la cual consiste en una red donde los nodos denotan las actividades a realizarse, la ruta crítica suele utilizarse en proyectos muy grandes donde se requiere gran coordinación. En la planeación también se define la estructura organizacional para la realización del sistema, es decir, se asignan responsables.

Análisis y Diseño.

El análisis y el diseño son las etapas más determinantes dentro del ciclo, ya que de estas depende la vida del sistema, es aquí donde se define el QUE, el COMO y CUANDO . En estas etapas se determinan los datos que se va a manejar el sistema, la localización de los mismos (archivos que se van a localizar), los procesos que se requieren y la forma en que se comunicarán.

Con los resultados de esta fase, se puede definir el lenguaje más óptimo para que se realize el desarrollo del sistema.

Programacion y Pruebas.

Se refiere a la codificación de los programas en un lenguaje de computación, y a su depuración hasta dejarlos en buen funcionamiento.

Liberacion, Instalacion y Documentacion.

En esta etapa se presenta el sistema terminado al usuario con la documentación necesaria para su operación y mantenimiento.

Mantenimiento.

Son los cambios, reducciones o ampliaciones que el sistema vaya requiriendo durante su vida operable.

Deceso del Sistema.

Ocurre cuando se requiere un nuevo sistema que mejore el anterior. Esto sucede cuando la organización donde operaba el sistema cambia de políticas, cuando el sistema se vuelve obsoleto por equipo de hardware o porque se requiere otro tipo de procesos que no son compatibles con lo que se plantearon al principio del sistema en deceso. El nuevo sistema pasará por todas las etapas antes citadas.

Las etapas anteriores son las correspondientes al ciclo de vida de un sistema computarizado.

IV.1.4. TECNICAS ESTRUCTURADAS.

Anteriormente no se empleaban técnicas bien definidas para el desarrollo de sistemas o si se utilizaba alguna metodología no era general. La metodología estructurada incluye el uso de técnicas generales, ya que se enfocan a analizar el problema lógico y no físico. Además dichas técnicas están fundamentadas matemáticamente.

La Técnica Estructurada incluye básicamente los siguientes pasos

A) Análisis Estructurado.

B) Diseño Estructurado.

C) Programación o Codificación Estructurada.

A) ANALISIS ESTRUCTURADO.

El análisis estructurado, es la primera fase en el proyecto del desarrollo de un sistema EDP (Electronic Data Process).

Para entender el análisis estructurado, es importante examinar los pasos que usualmente se siguen en el análisis del sistema clásico. En este, el analista prepara un documento que describe el propósito del sistema y que el usuario revisa. Este documento está formado por una gran cantidad de páginas escritas en términos técnicos, con las siguientes características:

1) Es monolítico, es decir, debe hacerse de principio a fin.

2) Es redundante, se da la misma información en diferentes lugares.

3) Es difícil de modificar y mantener.

4) Se preocupa por la parte física no por la lógica.

De las características anteriores, se puede deducir que existen problemas que el análisis estructurado trata de resolver.

El análisis estructurado es básicamente el uso de herramientas gráficas en la documentación, para producir un nuevo tipo de especificaciones funcionales.

Las herramientas gráficas para lograr las metas del análisis estructurado son las siguientes:

a) Diagrama de Flujo de Datos (DFD).

b) Diccionario de Datos.

c) Especificaciones de proceso o Miniespecificaciones.

d) Diagrama de Relación de Entidades (E-R).

e) Diagrama de Transición de Estados (STD).

a) Diagrama de Flujo de Datos.

Es el modelo de datos a través del sistema en forma gráfica, constituido por los siguientes elementos:

- Archivos.
- Fuente o destino de los datos.
- Proceso de datos.
- Flujo de datos

b) Diccionario de Datos.

Es la colección organizada de definiciones lógicas de los nombres de los datos que se encuentran en el diagrama de flujo de datos. La estructura para la definición de los datos es la siguiente:

< Nombre del Dato > = < Definición >

c) Especificaciones de Proceso o Minieespecificaciones.

Permite la descripción rigurosa y precisa de las políticas (pero no la implementación de las tácticas) representadas en los procesos de los niveles más bajos del diagrama de flujo de datos.

La especificación del proceso puede realizarse de diversas formas como fórmulas gráficas, tablas de decisión o en pseudocódigo.

d) Diagrama de Relación de Entidades (E-R).

Se refiere a la relación existente dentre los datos que se almacenan. Cada dato en el diagrama de flujo de datos (DFD) corresponde a un objeto en el diagrama de relación de entidades.

e) Diagrama de Transición de Estados (STD).

Estos diagramas son muy útiles en sistemas de tiempo real, es una modificación del DFD.

B) DISEÑO ESTRUCTURADO.

En el diseño estructurado del sistema se trata de lograr una especificación clara y completa de los requerimientos de software de un sistema.

Módulo y Modularidad.

Un aspecto importante a considerar en el diseño estructurado es la modelaridad. Un módulo se puede definir como un conjunto de instrucciones a las que se les puede asignar un nombre (módulo lógico).

Existen algunas formas principales de clasificar los módulos:

- Según su función, hay módulos de control para establecer relaciones, tomar decisiones y llamar a otros módulos.
- De proceso, para realizar alguna tarea operativa.
- De rutinas auxiliares, para usos como validación, inicialización e impresión.
- De interfases E/S, los cuales involucran a algún dispositivo periférico.
- De errores y excepciones los cuales se encargan de recuperar errores del sistema y reestablecerlo para su ejecución normal.
- Según su ejecución, hay módulos secuenciales, es decir uno después de otro; incrementales, cuando se ejecutan procesos intermedios; y paralelos si se ejecutan en forma simultánea.

Una característica de los módulos es que son independientes, lo cual es la base de la modularidad. Un módulo debe tener la mayor cohesión posible y el menor acoplamiento. La estructura del sistema no nos la da el flujo de datos, sino la dependencia entre módulos.

En general la metodología del diseño estructurado sigue los siguientes pasos:

+ Revisión y evaluación del DFD del análisis
+ División en subsistemas.
+ Evaluación para cada subsistema de los recursos y del tipo de sistema.
+ Para cada subsistema se realiza una revisión de los archivos definiéndolos físicamente.
+ Se evalúa el tipo de diseño: transformación y transacciones.

C) CODIFICACION O PROGRAMACION ESTRUCTURADA.

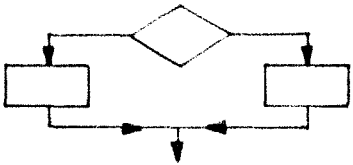
El término fue concebido por el profesor Edsger Dijkstra a mediados de la década de los 60's. Fue la primera de la técnicas estructuradas. Esta técnica se basa en la teoría de que si el código del programa se escribe cuando solamente las tres

estructuras, secuencia, selección e iteración; se tendrían pocos errores y podría modificarse con mayor velocidad y seguridad.

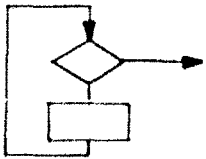
Las bases teóricas de todos los procedimientos lógicos tradicionalmente se describen con diagramas de flujo. A mediados de los 60's. Conrado Bohm y Guisseppe Jacopini, concluyeron matemáticamente que cualquier procedimiento lógico, esto es, cualquier diagrama de flujo, se puede derivar de las siguientes tres condiciones:

Secuencia

Selección



Iteración



Otras Metodologías.

Para el diseño de sistemas, en particular de software, existen diversas técnicas, las cuales nos guían a través de diversos pasos. Entre las más empleadas y las más conocidas podemos citar las siguientes:

a) Diseño Top-Down.
b) Diseño Bottom-Up.
c) Diseño por Prototipos.

IV.2 DESCRIPCION DEL PROYECTO.

El proyecto descrito a continuación se desarrollo en la División de Estudios de Posgrado de la Facultad de Ingeniería (DEPFI).

La teoría de redes es una clase de modelos matemáticos que involucra la representación gráfica de ciertos problemas de optimización. Las redes tienen una aplicación extensa en diversos campos como son: planeación, administración, ingeniería, química, educación y otros.

En estos campos innumerables situaciones pueden formularse como modelos matemáticos de redes. Algunos ejemplos son: sistemas de producción-distribución, tráfico urbano, transporte colectiv comunicación, redes eléctricas, reemplazo de equipo, inventario prensas, flujo de dinero, tuberías, oleoductos, asignación recursos y otras aplicaciones más.

A causa de la vasta aplicación de este tipo de modelos y a la valiosa ayuda que proporcionan para el entendimiento de los sistemas, ha habido gran en su estudio. Gracias a esto, y a la escritura especial que presentan los modelos de redes, surgió el desarrollo de este proyecto.

El objetivo del proyecto es diseñar e implementar una herramienta de trabajo que ayude a resolver problemas de teoría de redes aplicados a varios campos, de una forma más rápida y eficaz.

La aplicación de la teoría de redes a este tipo de situaciones hace que la solución que se obtenga sea óptima, esto es, obtener la mejor solución al problema.

Esta herramienta presenta una serie de métodos para la solución de problemas aplicando la teoría de redes . Los métodos que se trabajan con esta herramienta son: *Conectividad de una red, Busqueda en anchura, Busqueda en profundidad, Arbol de Expansion Minima, Ruta corta y Flujo maximo*

Algunos de estos metodos tienen diferentes formas de aplicación dependiendo del tipo de datos que manejan y al tipo de problema al cual se aplican.

Uno de los propósitos de esta herramienta es la obtención de resultados rapidamente en pantalla o en impresora, de acuerdo a la selección del usuario. Esto puede lograrse de acuerdo al equipo con que cuente el usuario para utilizar el sistema, ya que la velocidad de procesamiento de la computadora es un factor importante en la obtención de dichos resultados.

IV.3 ANALISIS DEL SISTEMA.

El análisis es la primera etapa dentro del desarrollo de un sistema. El aspecto más importante que se puede obtener del análisis es saber lo que se va a hacer.

El análisis consiste en un planteamiento inicial de los objetivos con los cuales el sistema deberá cumplir, siempre y cuando no tenga antecesor, de lo contrario, el análisis consiste en el planteamiento de un nuevo sistema que cumpla con las necesidades que se tengan actualmente, procurando mejorar su tiempo respuesta usando un lenguaje más apropiado.

El análisis se llevará a cabo mediante el uso de técnic estructuradas.

IV.3.1 PLANTEAMIENTO DE LOS OBJETIVOS.

En está sección se inicia la elaboración de un sistema que se forma por el planteamiento adecuado de los objetivos acordes a los requerimientos establecidos. Es importante agregar que los sitemas no pueden ser creados sin una documentación que describa claramente las funciones que contenga el sistema a desarrollar.

Para el desarrollo de este sistema, los objetivos que se persiguen son:

a) Mostrar al usuario los métodos de solución que cuenta para la solución de redes de flujo.

b) Capturar y validar la información referente a cada método de solución de redes de flujo.

c) Una vez validada la información, proceder a resolver el problema.

d) Mejorar el tiempo de respuesta.

e) Implementar ayuda y autopresentación al sistema.

f) El sistema dará los resultados obtenidos en pantalla y en impresora.

g) El sistema alcanzará un alto grado de funcionalidad.

IV.3.2 RECURSOS.

Para ubicar los requerimientos de los programas a desarrollar, a continuación se describen los componentes de *hardware* utilizados en el desarrollo del sistema "Métodos de Solución de Problemas de Redes de Flujo (IO)". Se describirán algunas de las características de las microcomputadoras personales IBM PC y compatibles, bajo las cuales ha sido desarrollado este sistema.

IV.3.2.1 COMPONENTES DE LA IBM PC Y COMPATIBLES.

Los cuatro miembros de la familia IBM PC son: la PC, la PC XT, la PC portátil y la PC AT. Las tres primeras son similares entre sí, todas usan el microprocesador 8088 y pueden direccionar hasta un megabyte de memoria. Sin embargo, la PC AT usa el microprocesador 80286, el cual emplea un bus de datos de 16 bits y un bus de direcciones de 24 bits. Como resultado, es más rápida y puede direccionar hata 16 megabytes de memoria.

En la figura IV.1 se muestra un diagrama del sistema.

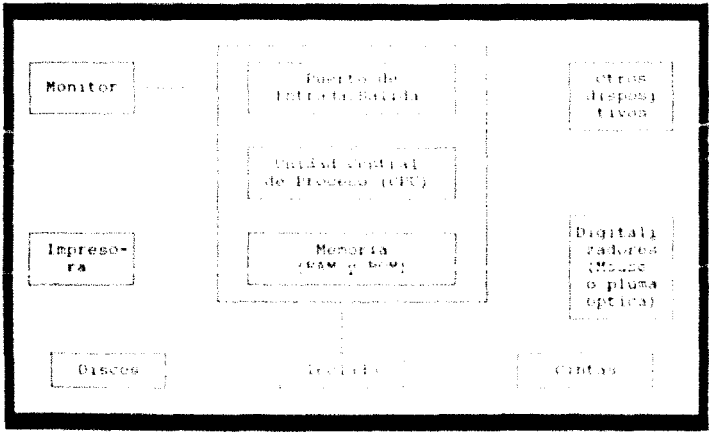


Figura IV.1 Diagrama de bloques del sistema de la PC

El sistema de la PC está formado de una gran variedad, de dispositivos opcionales, incluyendo una o dos unidades de *Diskette*, unidad de despliegue, impresora y expansiones de almacenamiento en unidades de 256 Kbytes.

El procesador central (CPU).

Los microprocesadores de INTEL de la familia que incluye el 8086, 8088, 80186 y 80286 son similares en muchos aspectos. Están organizados como dispositivos de 16 bits que pueden ejecutar un conjunto de instrucciones comun. Mientras todos estos microprocesadores operan internamente con datos de 16 bits, el 8088 solamente tiene un bus de datos de 8 bits hacia el exterior. En contraste, los otros dos son bus de datos externos de 16 bits.

Estos microprocesadores incorporan catorce localidades de memoria, llamdos registros, de un tamaño de 16 bits. Debido a que estos registros se localizan internamente en el microprocesador, pueden ser accesados más rapidamente.

Los registros pueden ser divididos en varios grupos:

Los registros de propósito general

Cuatro de los registros del microprocesador pueden ser usados para manipular datos. Estos cuatro registros pueden ser usados ya sea como registro de 16 bits (AX, BX, CX, DX) o como registros de 8 bits (AH, AL, BH, BL, CH, CL, DH, DL).

Los registro apuntadores.

El microprocesador cuenta con otros registros (SI, DI, SP, BP o IP) que sirven como apuntadores a direcciones de datos en la memoria principal.

Los registros de segmento.

El microprocesador usa 20 bits para formar una dirección, por lo que puede manejar hasta un megabyte de memoria, sin embargo, el microprocesador no incorpora ningún registro de 20 bits por lo que es necesario combinar dos registros para direccionar una localidad.

Los procesadores matemáticos 8087 y 80287.

Los microprocesadores de la familia 8088 pueden realizar operaciones aritméticas y tambien pueden realizar operaciones lógicas, sin embargo, estas operaciones se realizan solamente en enteros de uno y dos bytes. Si se quieren realizar operaciones con números de punto flotante es necesario escribir rutinas que las

realicen, las cuales son más lentas que las operaciones realizadas por *hardware*.

Debido a la importancia de las operaciones de punto flotante, en aplicaciones científicas y de ingeniería, INTEL ha fabricado una familia de procesadores auxiliares para este propósito. El coprocesador matemático 8087 trabaja con el 8088 o 8086; el 80287 trabaja con el 80286. Estos dispositivos son llamados coprocesadores porque pueden realizar sus cálculos al mismo tiempo que está operando el microprocesador central.

Los coprocesadores 8087 y 80287 realizan todos sus cálculos con una precisión de 80 bits. Hay siete tipo de datos distintos se tienen operaciones matemáticas como el seno, coseno, logaritmo raíz cuadrada, etc. Los cálculos se obtienen más rápidamente y con una precisión mayor que si se implementaran por *software*.

Organización de la Memoria.

IBM ha asignado varias actividades para diferentes partes del espacio de memoria de un megabyte. Una porción contiene instrucciones generales permanentes para la operación de la computadora y sus accesorios, otra guarda los programas y datos de los usuarios y una tercera está dedicada a la memoria de video.

Antes de ver la organización de la memoria, revisaremos brevemente los dos tipos de memoria. Algunas de las celdas de memoria de una computadora pueden ser leídas pero no modificadas; este tipo es llamado memoria de solo lectura (ROM). Un segundo tipo de memoria puede ser modificada, esto es, se puede leer y escribir; este tipo de memoria es llamada de acceso aleatori (RAM).

La ROM es memoria permanente disponible en el instante en que la computadora es encendida y no puede ser modificada por un programa de computadora. En contraste, la información contenida en RAM se pierde cada vez que la computadora es apagada. Esta memoria debe ser reprogramada cada vez que la computadora es encendida, sin embargo, la ventaja de la RAM es que puede ser modificada de acuerdo a las necesidades del usuario. El mapa de memoria de la PC se muestra en la tabla IV.1.

Los últimos 64 kbytes son ROM. La primera parte de esta memoria contiene el BIOS (Basic Input Output System) que es una porción de la memoria que contiene instrucciones para el programa de autoprueba. La otra parte de esta memoria ROM contiene las instrucciones del intérprete de BASIC. Los siguientes segmentos normalmente no son utilizados pero pueden contener RAM o ROM. El segmento C800 tiene ROM para la operación del disco duro si éste está instalado, de otra forma esta región estará vacía. Los segmentos B000 y B800 contienen RAM para la memoria de video. El resto de la memoria es utilizada como memoria de trabajo.

Puertos de Entrada/Salida.

El microprocesador se comunica con un megabyte de memoria principal a través de un bus de direcciones de 20 bits. Además hay un área de 1024 bytes para 256 vectores de interrupción. Estos vectores con los puntos de entrada a las subrutinas de servicio localizadas en algún lugar de la memoria. Las rutinas de servicio se comunican directamente con los periféricos a través de registros conocidos como puertos de entrada/salida. Usualmente más de un puerto es asignado a cada periférico; uno puede ser para status y otro para datos. Ya que las direcciones de los puertos son *numeros* de 16 bits, puede haber en un principio hasta 64k puertos diferentes. Sin embargo la PC por diseño solo es capaz de direccionar 1k puertos.

Los puertos seriales.

La PC tiene dos puertos seriales. El primer puerto es llamado AUX o COM1; el segundo puerto es llamado COM2. Estos puertos pueden ser usados para un módem telefónico, una impresora serial, un mouse o una conexión directa a otra computadora. Las direcciones de los puertos series están dadas en el área de datos del BIOS en las localidades de puertos 400 y 402.

Los puertos paralelos

La PC tiene tres puertos paralelos. Estos puertos son llamados PRN o LPT1, LPT2 y LPT3. Las direcciones de estos puertos están almacenadas en las localidades de puertos 408, 40A y 40C.

Dirección (hex)	Descripción
0000	Vectores de interrupción del BIOS en ROM
0008	Vectores de interrupción del DOS en RAM
0040	Area de datos del BIOS en RAM
0050	Area de datos del DOS y de BASIC en RAM
A800	Memoria de video del EGA en RAM
B000	Adaptador monocromático en RAM
B800	Adaptador gráfico (CGA) en RAM
C800	Puntos en ROM para manejo del disco duro
F600	BASIC en ROM
FE00	BIOS en ROM

Tabla IV.1 Mapa de memoria de la PC.

El monitor monocromatico.

El monitor monocromático despliega información escrita en los primeros 4 kbytes de memoria en el segmento B000. El monitor despliega los caracteres Ascii normales y los caracteres gráficos. Los caracteres están formados por arreglos de 14 x 19 puntos.

El monitor monocromático puede desplegar 25 líneas de texto de 2000 caracteres, sin embargo, la pantalla requiere 4000 bytes ya que se requieren dos bytes para cada caracter. La tarjeta adaptadora monocromática contiene 4 kbytes de memoria (4096 bytes), es decir hay 96 bytes extras de memoria que no se usan para desplegar información.

Cada localidad de memoria de video es de 16 bits, el primer byte (la dirección par) es el caracter a ser desplegado y el segundo byte (la dirección impar) es el atributo que describe la apariencia del caracter.

Hay varias formas en que un caracter monocromático puede ser desplegado: normal, brillante, video inverso y brillante en video inverso, además se puede hacer que cada caracter parpadee y los caracteres normales y brillantes pueden ser subrayados ya sea parpadeando o no.

Los monitores de colores

El segmento A000 se usa cuando está instalado un adaptador gráfico extendido (EGA,VGA, etc). El monitor monocromático estándar utiliza 4 kbytes de memori empezando en la dirección B000:0000; el monitor gráfico utiliza 16 kbytes de memoria empezando en la dirección B800:0000. Cualquier información escrita en estas áreas de memoria será desplegada inmediatament en pantalla.

El monitor grafico

El monitor gráfico es manejado por el adaptador gráfico de color (CGA), el cuual tiene 16 kbytes de memoria empezando en la dirección B800:0000. Mientras el monitor monocromático tiene un solo modo texto, el monitor gráfico de color tiene siete modos diferentes.

El CGA puede ser inicializado a alguno de estos siete modos de los cuales cuatro son modos para texto y los tres restantes son modos para gráficos. Los modos texto despliegan 40 u 80 caracteres en cada línea en colores o en blanco y negro.

Los modos gráficos tienen una resolución de 320 columnas por 200 renglones en cuatro colores o 640 x 200 en dos colores.

El teclado.

El teclado de la PC incorpora un microprocesador 8048 para procesar la información. Tiene un *buffer* de 16 bytes por lo que se puede seguir escribiendo en el teclado mientras la computadora se encuentra realizando algun otro proceso. La señal de interrupción generada por el teclado interrumpe temporalmente a la computadora para que la tecla que ha sido presionada pueda ser identificada y almacenada en un *buffer*.

Cuando la computadora se encuentra lista para recibir otro caracter, el procesador del teclado le envia el siguiente caracter que se encuentre en su *buffer*. El procesador del teclado también es capaz de detectar cuando una tecla se mantiene presionada por un tiempo prolongado, para enviar mas rápidamente el caracter al procesador central de la computadora.

El teclado de la PC es diferente de otros teclados los cuales envían el caracter en su formato ASCII. El teclado de la PC envia un código propio de identificación conocido como *scan code*. Todas las teclas están numeradas de 1 a 83 y cuando se presionada alguna tecla, el procesador del teclado envia su numero (el codigo de la tecla más 128).

Las impresoras

La impresora de la PC normalmente se conecta a través del puerto paralelo al cual se hace referencia como LPT1 y es usualmente una unidad de matriz de puntos de impacto con una cabeza de impresión móvil, aunque existen otros tipos.

IV.3.3 DEFINICION DE REQUERIMIENTO DE LOS PROGRAMAS.

El propósito de esta sección es definir formalmente los requerimientos que deben cumplir los programas para que el sistema funcione adecuadamente. Una vez realizada la investigación sw los métodos de solución de redes de flujo, fueron planteados los requerimientos fundamentales que los programas deben cumplir para el fin del proyecto.

Una característica que deben presentar los programas, es que debe de ser de fácil manejo, es decir, el programa debe mostrar continuamente las opciones y el procedimiento a seguir, verificándose los posibles errores e indicando el motivo de estos, además de contar en todos los programas con una misma tecla de ayuda que al presionarla muestre información más completa del funcionamiento del programa.

Cabe mencionar que se requiere que los programas tomen control total de la operación del sistema, pata que el usuario tenga completo control del funcionamiento del sistema a través del teclado de la computadora.

IV.3.4 DIAGRAMA DE FLUJO DE DATOS.

Una vez que se han determinado los requerimientos que deben cumplir los programas se procede a establecer un diagrama de flujo de datos del sistema, el diagrama de flujo de datos del se muestra en la figura IV.3

El proceso Atención a Usuario está organizado en forma de *"menus"*, donde el usuario puede elegir con simples movimientos del cursor, la opción deseada. La entrada que recibe este proceso es el comando que el usuario desea ejecutar y la salida que genera es la opción seleccionada que corresponda a dicho comando.

El establecimiento de la comunicación entre el usuario y los procesos la realiza el proceso Manejador. las entradas que recibe este proceso son: la indicación de parte del usuario del uso de los procesos de Edición, Manejo-de-Archivos, Métodos y Dibuja-Red; la indicación de verificación de existencia de ratón, indicación de regreso a este proceso por parte de los procesos de Edición, Manejo-de-Archivos, Métodos y Dibuja-Red. La salida que genera este proceso son: activación del proceso indicado

El proceso ratón realiza la verificación de la conexión del ratón a la microcomputadora para activarlo en caso de que este conectado. Este proceso servirá como auxiliar para que los *movimientos del cursor hechos con el ratón sean más ágiles para el usuario.* La entrada que recibe este proceso es la indicación de activación por parte del proceso Manejador. Las salidas que genera este proceso son: indicación si el ratón encuentra instalado o no, la indicación al proceso Error, si no encuentra conectado el ratón a la microcomputadora, petición de regreso al proceso Manejador.

En el proceso Dibuja-Red el usuario podrá realizar el dibujo de la red o redes que desee, las cuales solo podrán crearse y editarse con este sistema, este proceso cuenta con las siguientes opciones: cargar, desplegar, salvar, editar y renombrar los archivos creados. Las entradas que recibe este proceso son la indicación de activación del proceso por parte del proceso Manejador, el resultado obtenido al ejecutarse el proceso Archivo-a-Memoria . Las salidas que genera este proceso son: indicación al proceso Archivo-a-Memoria de cargar el archivo seleccionado a memoria, la indicación al proceso Error en caso de que haya ocurrido un error en el proceso Archivo-a-Memoria o en este mismo proceso, petición de regreso al proceso Manejador.

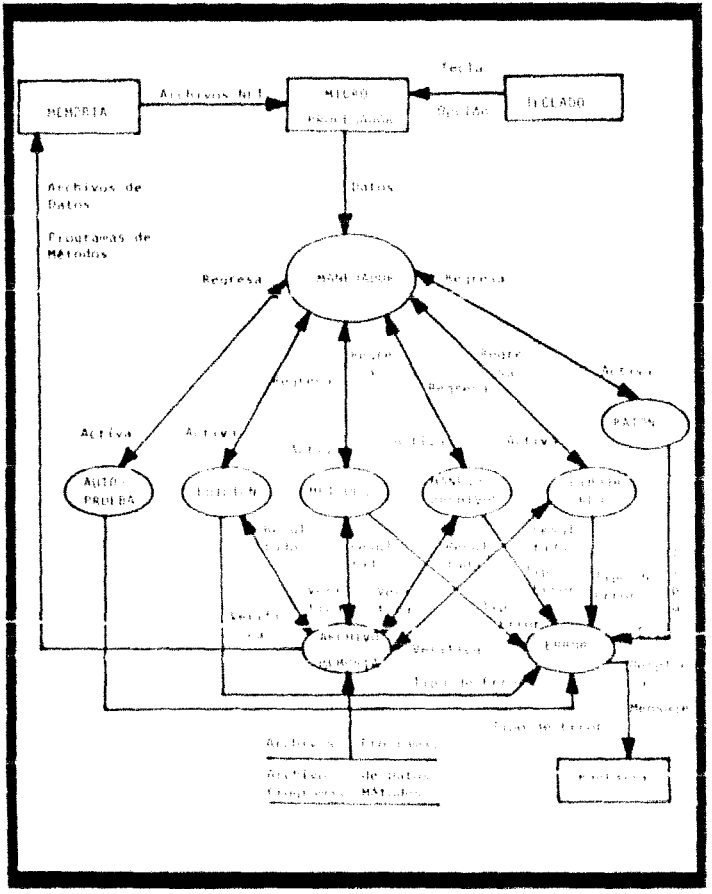


Figura IV.1 Diagrama de flujo de Datos

El proceso Manejo-de-Archivos es el encargado de realizar las funciones de copiar, borrar y renombrar archivos, así como observar el contenido del directorio seleccionado, cambio de directorio, salida Shell del sistema y salida del sistema. Las entradas que recibe este proceso son: la indicación de activación por parte del proceso Manejador, el resultado obtenido al ejecutarse el proceso Archivo-a-Memoria. Las salidas que genera este proceso son: indicación al proceso Archivo-a-Memoria de cargar el archivo seleccionado a memoria, la indicación al proceso Error en caso de que haya ocurrido un error en el proceso Archivo-a-Memoria o en este mismo proceso, petición de regreso al proceso Manejador.

El proceso Edición es el encargado de realizar las funciones de observación y creación/modificación de archivos de datos que utiliza el proceso Métodos para la ejecución de los métodos de solución que contiene. Las entradas que recibe este proceso son: la indicación de activación por parte del proceso Manejador, el resultado obtenido al ejecutarse el proceso Archivo-a-Memoria. Las salidas que genera este proceso son: indicación al proceso Archivo-a-Memoria de cargar el archivo seleccionado a memoria, la indicación al proceso Error en caso de que haya ocurrido un error en el proceso Archivo-a-Memoria o en este mismo proceso, petición de regreso al proceso Manejador.

El proceso Métodos es el encargado del manejo de los métodos de solución aplicados a problemas de teoría de redes, este proceso emplea los siguientes métodos: conectividad, búsqueda en anchura, búsqueda en profundidad, ruta corta (Kruskal), ruta corta (Prim), flujo máximo (Fordfulker son), árbol de expansión mínima (Dijkstra) y árbol de expansión mínima (Floyd). Las entradas que recibe este proceso son: la indicación de activación por parte del proceso Manejador, el resultado obtenido al ejecutarse el proceso Archivo-a-Memoria. Las salidas que genera este proceso son: indicación al proceso Archivo-a-Memoria de cargar el archivo seleccionado a memoria, la indicación al proceso Error en caso de que haya ocurrido un error en el proceso Archivo-a-Memoria o en este mismo proceso, petición de regreso al proceso manejador.

El encargado de cargar los archivos de datos y los programas a la memoria es el proceso Archivo-a-Memoria. Las entradas que recibe este proceso son: el nombre del archivo proveniente de los procesos Edición, Manejo-de-Archivos, Métodos y Dibuja-Red. Las salidas que genera este proceso son: el resultado de la operación de cargar el archivo a memoria, indicación de error si ocurre alguno y en caso de que no exista ningún error, la información será cargada en memoria.

Finalmente el proceso Error únicamente se encarga de indicarle al usuario, a través de la pantalla, que ha ocurrido algún error en alguno de los procesos anteriores. La entrada que recibe este proceso es la indicación de que hubo error en alguno de los procesos anteriores y la salida que genera es el despliegue del mensaje correspondiente a ese error.

IV.3.5 DESCRIPCION DE ENTRADA/SALIDA.

Las pantallas del sistema son básicamente de tres tipos: menus, captura de información (datos) y despliegue de información (resultados).

Las pantallas que muestran el menu tienen el formato de la figura IV.4. Las pantallas de captura tienen el formato que se muestra en la figura IV.5.

Las pantallas de despliegue de información varían de acuerdo al método de solución seleccionado, el formato general de estas pantallas se muestran en la figura IV.6.

IV.3.6 PSEUDOCODIGO.

En esta sección se presenta una descripción de los procesos que componen este sistema en forma de pseudocódigo, considerando una especificación completa del sistema.

```
1. ATENCION A USUARIOS
     Despliegue de Menu.
     Repite
          Lee Comando
     Hasta que haya selección
     Opción

2. ERROR
     Selecciona tipo de error
          1: Despliega mensaje 1
          2: Despliega mensaje 2
          3: Despliega mensaje 3
          .      .      .      .
          .      .      .      .
          .      .      .      .
          n: Despliega mensaje n
     Fin Selecciona

3. MANEJO DE ARCHIVOS
     Repite
          Despliega Menu
          Repite
               Lee Comando
          Hasta que haya selección
```

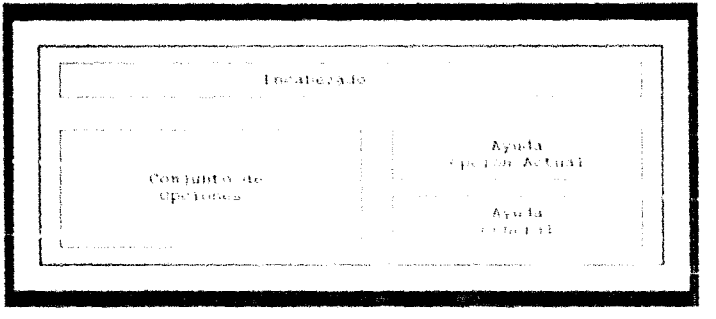


Figura [illegible]

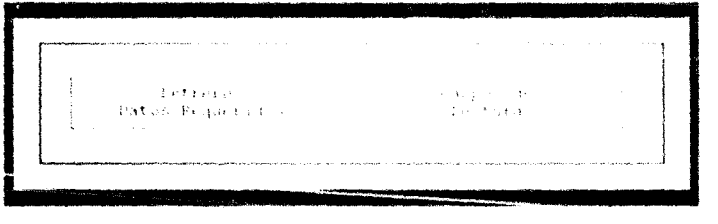


Figura [illegible]

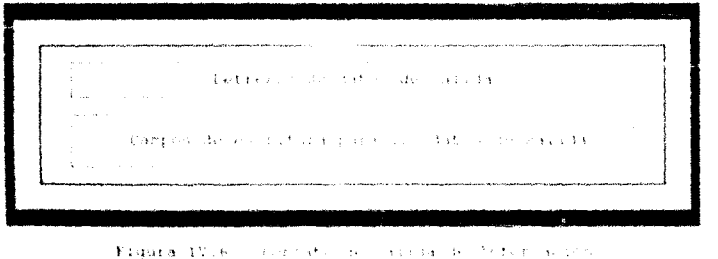


Figura [illegible]

```
Selecciona Comando Menú
     Copia: Mientras no termine proceso Haz
               Despliega Nombre de Archivos
               Repite
                    Lee selección
               Hasta que haya selección
               Si existe archivo Entonces
                    copia archivo seleccionado a archivo
                    nuevo

                    Resultado = No Error
               O Bien
                    Resultado = Error
               Fin Si
          Fin Mientras
     Borra: Mientras no termine proceso Haz
               Despliega Nombre de Archivos
               Repite
                    Lee selección
               Hasta que haya selección
               Si existe archivo Entonces
                    Borra archivo seleccionado
                    Resultado = No Error
               O Bien
                    Resultado = Error
               Fin Si
          Fin Mientras
     Renombra: Mientras no termine proceso Haz
               Despliega Nombre de Archivos
               Repite
                    Lee selección
               Hasta que haya selección
               Si existe archivo Entonces
                    Nombra archivo anterior con
                    nombre archivo nuevo

                    Resultado = No Error
               O Bien
                    Resultado = Error
               Fin Si
          Fin Mientras
     Directorio: Mientras no termine proceso Haz
               Lee Nombre Directorio
               Si existe directorio Entonces
                    Despliega archivos contenidos
                    Resultado = No Error
               O Bien
                    Resultado = Error
               Fin Si
          Fin Mientras
     Cambia_Dir: Mientras no termine proceso Haz
               Lee Nombre Nuevo Directorio
               Si existe directorio Entonces
                    Cambia nuevo directorio
                    Despliega nombres archivos
                    Resultado = No Error
```

```
                        O Bien
                            Resultado = Error
                        Fin Si
                    Fin Mientras
        S.O. Shell: Mientras no termine proceso Haz
                        Salida a sistema operativo
                        Repite
                            Espera
                        Hasta que el usuario regrese
                    Fin Mientras
        Salir: Mientras el usuario no seleccione salir Haz
                    Manejo sistema
                Fin Mientras
    Fin Selecciona
Hasta que el usuario solicite terminar.
```

4. EDICION DE ARCHIVOS

```
Repite
    Despliega Menú
    Repite
        Lee Comando
    Hasta que haya selección
    Selecciona Comando Menú
        Observar: Mientras no termine proceso Haz
                    Despliega nombre de archivos
                    Repite
                        Lee selección
                    Hasta que haya selección
                    Si existe archivo Entonces
                        Repeat
                            Despliega información
                        Hasta que el usuario solicite
                            terminar
                        Resultado = No Error
                    O bien
                        Resultado = Error
                    Fin Si
                Fin Mientras
        Editar :Mientras no termine proceso Haz
                    Despliega nombres de archivos
                    Repite
                        Lee selección
                    Hasta que haya selección
                    Si existe archivo Entonces
                        Repeat
                            Despliega información
                        Hasta que el usuario solicite
                            terminar
                        Resultado = No Error
                    O bien
                        Resultado = Error
                    Fin Si
                Fin Mientras
```

```
                Salvar: Mientras no termine proceso Haz
                            Escribe el archivo a disco
                            Si escribe archivo Entonces
                                Resultado = No Error
                            O bien
                                Resultado = Error
                            Fin Si
                        Fin Mientras
        Fin Seleccion
Hasta que el usuario solicite terminar
```

5. METODOS

```
Repite
        Despliega Menu
        Repite
                Lee Comando
        Hasta que haya seleccion
        Selecciona Comando Menu
                Conectividad: Mientras no termine proceso Haz
                                        Despliega Opciones
                                        Repeat
                                                Lee opción
                                        Hasta que haya selección
                                        Si opcion = archivo Entonces
                                                Despliega Archivos .Net
                                                Repeat
                                                        Selección archivo
                                                Hasta que haya selección
                                                Si existe archivo Entonces
                                                        Lee archivo
                                                        Resultado = No Error
                                                O bien
                                                        Resultado = Error
                                                Fin Si
                                        O bien
                                                Lee datos de teclado
                                        Fin Si
                                        Procesa datos
                                        Despliega Resultados
                                Fin Mientras
                Anchura : Mientras no termine proceso Haz
                                        Despliega Opciones
                                        Repeat
                                                Lee opcion
                                        Hasta que haya selección
                                        Si opcion = archivo Entonces
                                                Despliega Archivos .Net
                                                Repeat
                                                        Selección archivo
                                                Hasta que haya selección
                                                Si existe archivo Entonces
                                                        Lee archivo
                                                        Resultado = No Error
```

```
                    O bien
                        Resultado = Error
                    Fin Si
                O bien
                    Lee datos de teclado
                Fin si
                Procesa datos
                Despliega Resultados
            Fin Mientras
Profundidad : Mientras no termine proceso Haz
                Despliega Opciones
                Repeat
                    Lee opción
                Hasta que haya selección
                Si opcion = archivo Entonces
                    Despliega Archivos .Net
                    Repeat
                        Selección archivo
                    Hasta que haya selección
                    Si existe archivo Entonces
                        Lee archivo
                        Resultado = No Error
                    O bien
                        Resultado = Error
                    Fin Si
                O bien
                    Lee datos de teclado
                Fin si
                Procesa datos
                Despliega Resultados
            Fin Mientras
AEMDijkstra1: Mientras no termine proceso Haz
                Despliega Opciones
                Repeat
                    Lee opción
                Hasta que haya selección
                Si opcion = archivo Entonces
                    Despliega Archivos .Net
                    Repeat
                        Selección archivo
                    Hasta que haya selección
                    Si existe archivo Entonces
                        Lee archivo
                        Resultado = No Error
                    O bien
                        Resultado = Error
                    Fin Si
                O bien
                    Lee datos de teclado
                Fin si
                Procesa datos
                Despliega Resultados
            Fin Mientras
AEMDijkstra2: Mientras no termine proceso Haz
                Despliega Opciones
```

```
                        Repeat
                              Lee opción
                        Hasta que haya selección
                        Si opcion = archivo Entonces
                              Despliega Archivos .Net
                              Repeat
                                    Selección archivo
                              Hasta que haya selección
                              Si existe archivo Entonces
                                    Lee archivo
                                    Resultado = No Error
                              O bien
                                    Resultado = Error
                              Fin Si
                        O bien
                              Lee datos de teclado
                        Fin Si
                        Procesa datos
                        Despliega Resultados
                  Fin Mientras
ArExMinFloyd:     Mientras no termine proceso Haz
                        Despliega Opciones
                        Repeat
                              Lee opción
                        Hasta que haya seleccion
                        Si opcion = archivo Entonces
                              Despliega Archivos .Net
                              Repeat
                                    Selección archivo
                              Hasta que haya selección
                              Si existe archivo Entonces
                                    Lee archivo
                                    Resultado = No Error
                              O bien
                                    Resultado = Error
                              Fin Si
                        O bien
                              Lee datos de teclado
                        Fin Si
                        Procesa datos
                        Despliega Resultados
                  Fin Mientras
Flujo-Maximo:     Mientras no termine proceso Haz
                        Despliega Opciones
                        Repeat
                              Lee opcion
                        Hasta que haya selección
                        Si opcion = archivo Entonces
                              Despliega Archivos .Net
                              Repeat
                                    Selección archivo
                              Hasta que haya selección
                              Si existe archivo Entonces
                                    Lee archivo
                                    Resultado = No Error
```

```
                    O bien
                        Resultado = Error
                    Fin Si
                O bien
                    Lee datos de teclado
                Fin si
                Procesa datos
                Despliega Resultados
            Fin Mientras
RucorKruskal: Mientras no termine proceso Haz
                Despliega Opciones
                Repeat
                    Lee opción
                Hasta que haya selección
                Si opcion = archivo Entonces
                    Despliega Archivos .Net
                    Repeat
                        Selección archivo
                    Hasta que haya selección
                    Si existe archivo Entonces
                        Lee archivo
                        Resultado = No Error
                    O bien
                        Resultado = Error
                    Fin Si
                O bien
                    Lee datos de teclado
                Fin si
                Procesa datos
                Despliega Resultados
            Fin Mientras
RutaCortaPri: Mientras no termine proceso Haz
                Despliega Opciones
                Repeat
                    Lee opción
                Hasta que haya selección
                Si opcion = archivo Entonces
                    Despliega Archivos .Net
                    Repeat
                        Selección archivo
                    Hasta que haya selección
                    Si existe archivo Entonces
                        Lee archivo
                        Resultado = No Error
                    O bien
                        Resultado = Error
                    Fin Si
                O bien
                    Lee datos de teclado
                Fin si
                Procesa datos
                Despliega Resultados
            Fin Mientras
    Fin Selecciona
Hasta que usuario solicite terminar
```

6. RATON

```
Mientras no termine proceso Haz
      Verifica conexion de raton
      Si existe raton conectado Entonces
            Inicializa cursor de raton
            Resultado = No error
      O bien
            Resultado = Error
      Fin Si
Fin Mientras
```

7. Dibuja-Red

```
Repite
      Despliega Menu
      Repite
            Lee Comando
      Hasta que haya selección
      Selecciona Comando Menu
            Cargar: Mientras no termine proceso Haz
                        Despliega Nombre de Archivos
                        Repite
                              Lee selección
                        Hasta que haya selección
                        Si existe archivo Entonces
                              Carga archivo seleccionado a memoria
                              Resultado = No Error
                        O Bien
                              Resultado = Error
                        Fin Si
                     Fin Mientras
            Desplegar: Repite
                              Si existe archivo en memoria Entonces
                                    Despliega archivo en pantalla
                                    Resultado = No Error
                              O Bien
                                    Resultado = Error
                        Hasta que usuario solicite terminar
            Salvar: Mientras no termine proceso Haz
                        Escribe el archivo a disco
                        Si escribe archivo Entonces
                              Resultado = No Error
                        O bien
                              Resultado = Error
                        Fin Si
                     Fin Mientras
            Editar: Mientras no termine proceso Haz
                              Despliega nombre de archivos
                              Repite
                                    Lee selección
                              Hasta que haya selección
                              Si existe archivo Entonces
                                    Repeat
                                          Despliega información
```

```
                                    Hasta que el usuario solicite
                                        terminar
                                    Resultado = No Error
                                O bien
                                    Resultado = Error
                                Fin Si
                            Fin Mientras
                Renombra: Mientras no termine proceso Hag
                                Despliega Nombre de Archivos
                                Repita
                                    Lee selección
                                Hasta que haya selección
                                Si existe archivo Entonces
                                    Nombra archivo anterior con
                                    nombre archivo nuevo
                                    Resultado = No Error
                                O Bien
                                    Resultado = Error
                                Fin Si
                            Fin Mientras
            Fin Selecciona
        Hasta que el usuario solicite terminar
```

IV.3.7 ANALISIS DE LA INFORMACION.

Una vez establecido del flujo de datos entre los procesos y la lógica de cada uno de estos, se procede a realizar el análisis de los datos utilizados en cada uno de los procesos descritos. Este análisis se muestra en las tablas IV.2-IV.9.

Proceso: Ratón			
Datos	Descripción	I/O	Comentarios
Activa	Bandera que indica si se activó el ratón	Interna	Es inicializada por el Procedimiento
Tipo de Error	Indica el tipo de error que se halla en el proceso	Salida	Se utilizará en el proceso Error

Tabla IV.2 Descripción de los datos del proceso Ratón

Proceso: Error			
Datos	Descripción	E/S	Comentarios
Tipo de Error	Señala el [illegible] que será desplegado de acuerdo al error que haya ocurrido.	Entrada	Es generado por los procesos de Edición, Manejo de Archivos, Métodos, Dibuja-Red y Datos.
Resultado	Indica si existe error en algún proceso.	Salida	Es utilizado en los procesos: Edición, Manejo de Archivos, Métodos, Dibuja-Red, Datos y [illegible]

Tabla IV.3 [illegible]

Proceso: Archivos-Manejo [illegible]			
Datos	Descripción	E/S	Comentarios
Resultado	Indica si [illegible] en algún proceso.	Salida	[illegible]
Nombre del Archivo	Nombre del Archivo que es leído y almacenado en la [illegible] nombre del [illegible] a [illegible] para [illegible]	Entrada	[illegible]

Tabla IV.4 [illegible]
[illegible]

Proceso: Manejo de Archivos			
Datos	Descripción	E/S	Comentarios
encontrado	Indica si existe error al verificar si existe el archivo seleccionado.	Entrada	Edición, Manejo de Archivos, Metodos, Dibuja-Red, Ratón y Archivo-Memoria
Verificar	Bandera que indica que puede iniciarse la lectura del archivo seleccionado.	Salida	Bandera que es activada dado el nombre del archivo a través del teclado.
Terminar	Se activa para que el usuario pueda interrumpir el proceso en cualquier momento.	Entrada	Bandera local activada a través del teclado por el usuario
Activa	Bandera que indica la activación del proceso.	Entrada	Es generada por el proceso atención
Proceso	Bandera que indica el reproceso del evento al proceso Manejador	Salida	Bandera que es activada a través del teclado por usuario.
Tipo de Error	Indica el tipo de error que se activo en el proceso.	Salida	Es utilizada en el proceso Error

Tabla IV.1 Descripción de los datos del proceso Manejo de Archivos

Proceso: Edición

Datos	Descripción	Flujo	Comentarios
Resultado	Indica si existe error al verificar si existe el archivo seleccionado.	Entrada	Edición, Manejo de Archivos, Métodos, Caja-Red, Ratón y Archivo-Memoria
Verifica	Bandera que indica que puede iniciarse la búsqueda del archivo seleccionado.	Salida	Bandera que es activada desde el menú del archivo a través del teclado.
Terminar	Se utiliza para que el usuario pueda interrumpir el proceso en cualquier momento.	Entrada	Bandera local activada a través del teclado por el usuario
Activa	Bandera que indica la activación del proceso.	Entrada	Es generada por el proceso principal
Respuesta	Bandera que indica el término del proceso.	Salida	Bandera que es activada a través del teclado por el usuario.
Tipo de Error	Indica el tipo de error que se detecta en el proceso.	Salida	Es utilizada en el proceso principal

Tabla IV.3 [illegible]

Proceso: Metodos			
Datos	Descripcion	[illegible]	Comentarios
Resultado	[illegible] al verificar [illegible] el archivo [illegible]do.	[illegible]	[illegible], Manejo de Archivos, Metodos, [illegible], Raton y Archivo-Memoria
Verifica	Bandera que [illegible] puede [illegible] la [illegible] queda del archivo [illegible] leccionado.	[illegible]	Bandera que [illegible] activada [illegible] el nombre del archivo a traves del tecla[illegible]do.
Terminar	Se utiliza para que el usuario pueda [illegible] pir el proceso [illegible] quir [illegible]	[illegible]	Bandera local activada a traves de [illegible] tecla [illegible] usuario [illegible]
Activa	Bandera que [illegible] la activacion del [illegible]so.	[illegible]	[illegible] por el [illegible]
Regresa	Bandera que [illegible] el regreso del [illegible] al proceso Manej[illegible]	[illegible]	Bandera que [illegible] del teclado por usuario[illegible].
Tipo de Error	Indica el tipo de error que [illegible] en el proceso [illegible]	[illegible]	[illegible]

Tabla IV.[illegible]

Proceso: Dibuja bol			
Datos	Descripción	E/S	Comentarios
Resultado	Indica si existe error al verificar si existe el archivo seleccionado.	Entrada	Edición, Manejo de Archivos, Métodos, Dibuja-bol, Ratón y Archivo-Memoria
Verifica	Bandera que indica que puede iniciarse la busqueda del archivo seleccionado.	Salida	Bandera que es activada dado el nombre del archivo a través del teclado.
Terminar	Se utiliza para que el usuario pueda interrumpir el proceso en cualquier momento.	Entrada	Bandera local activada a través del teclado por el usuario.
Activa	Bandera que indica la activación del proceso.	Entrada	Es demorada por el proceso atención
Regresa	Bandera que indica el regreso del control al proceso Manejador	Salida	Bandera que es activada a través del teclado por usuario.
Tipo de Error	Indica el tipo de error que se activo en el proceso.	Salida	Es utilizada en el proceso Error

Tabla IV.8 [illegible]

Proceso: Atención a Cuarto			
Datos	Descripción	I/O	Comentarios
Acti va	Bandera que indica que se activo el proceso	Entrada	Es generada por el proceso atención
Regre so	Bandera que indica el regreso del cuarto al piso Recepción	Salida	Esta bandera es ac tivada a través del teclado por el usuario.

Tabla IV.4 Descripción de datos del proceso Atención a Cuarto

IV.4 DISEÑO DEL SISTEMA.

El objetivo de esta sección es hacer una especificación clara y completa de los requerimientos de *software* de los programas de aplicación.

El diseño es la etapa anterior al desarrollo y es donde los requerimientos son transladados a su representación de software. La primera parte del diseño consiste en realizar la carta estructurada; en esta carta se especifican los procesos que se definieron en el Diagrama de flujo de Datos realizado en el análisis.

En esta etapa se definen en forma explícita las entradas, salidas y archivos a utilizar en el sistema.

IV.4.1 CARTA ESTRUCTURADA.

La estructura de software es definida como la representación jerárquica. Representa la relación entre módulos o elementos del software empleados, para la solución del problema.

Existen diversas metodologías dentro del diseño de la estructura de software, cada una de ellas es empleada de acuerdo al tipo de sistema que se está desarrollando.

Cabe mencionar la importancia de la simplificación de una estructura de software ya que de ello depende la complejidad o facilidad del desarrollo del sistema.

La carta estructurada para los programas de aplicación se muestra desglosada en las figuras IV.7-IV.10.

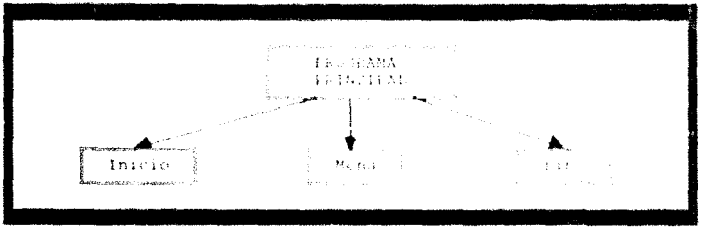


Figura IV.7 Carta Estructurada del Módulo principal

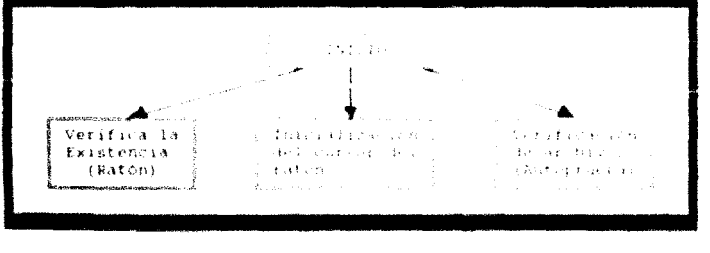


Figura IV.8 Carta Estructurada del [illegible]

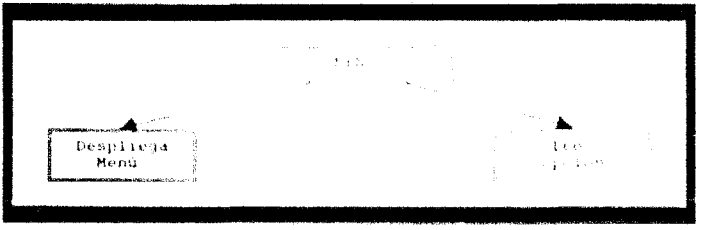


Figura IV.9

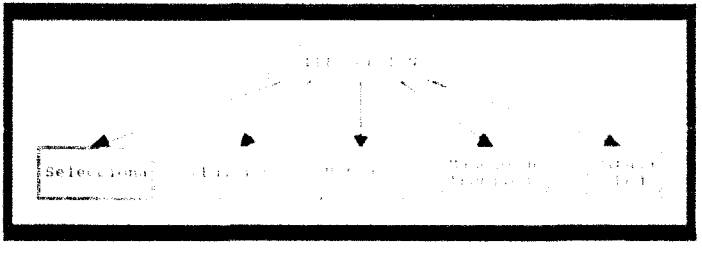


Figura IV.10

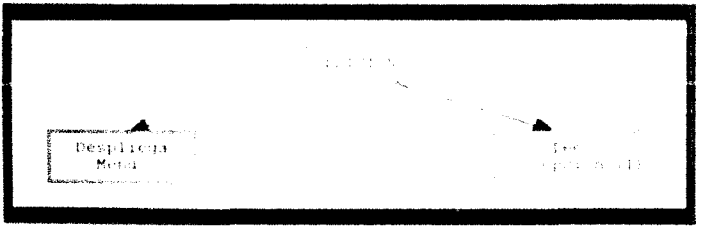


Figura IV.11

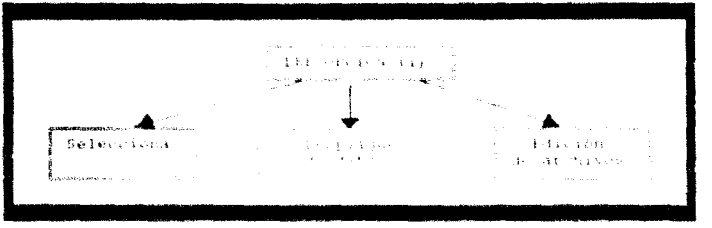


Figura IV.12

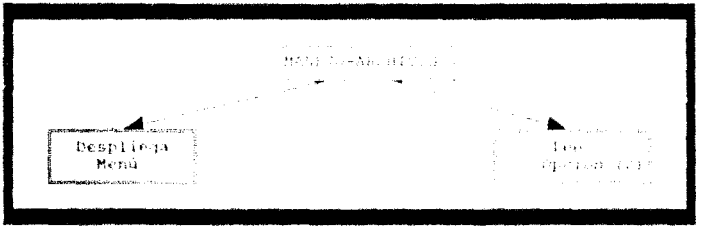


Figura IV.13

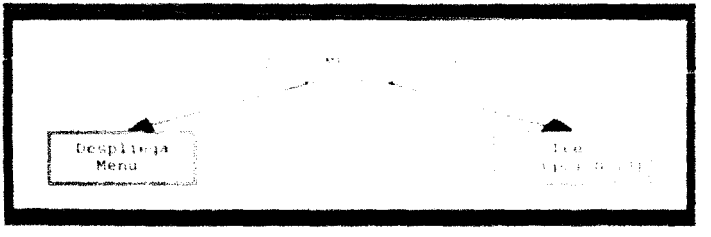


Figura IV.14

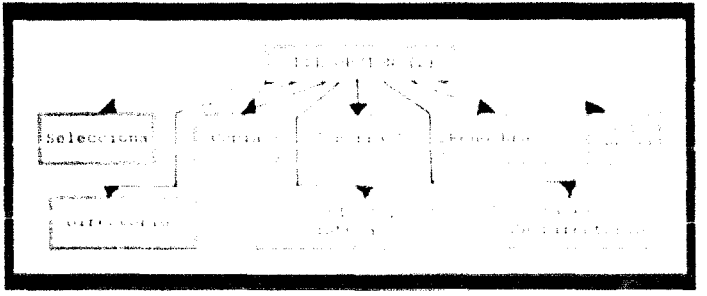


Figura IV.15

Figura IV.16

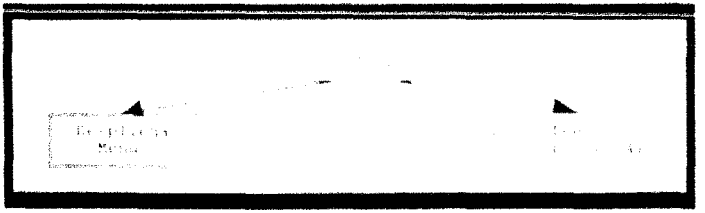


Figura IV.17

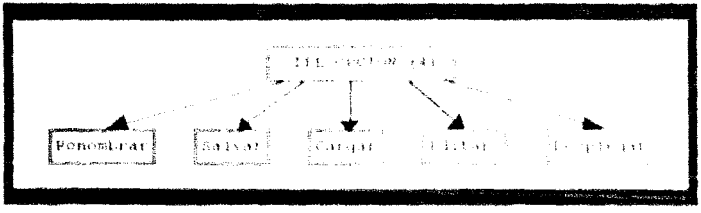


Figura IV.12 [illegible]

IV.4.2 DESCRIPCION DE PROCESOS.

Las tablas IV.10 a la IV.21 muestran en forma detallada la función que realiza cada uno de los procesos propuestos en la carta estructurada además de indicar los parámetros que recibe y que genera, los módulos que utiliza y la función que realiza cada uno de los subprocesos.

IV.5 DESARROLLO E IMPLEMENTACION DE LOS PROGRAMAS

En esta parte se describe como fue que los programas orientados al sistema Solución de Problemas de Redes de Flujo (IO)" fueron diseñados para su implementación en la computadora.

En esta sección se toman todas las ideas surgidas en los puntos anteriores, obedeciendo también a las técnicas estructuradas, optimizando siempre, con el fin de obtener excelencia en todo cuanto constituirá el sistema.

IV.5.1 LENGUAJE.

Aunque no existe una metodología para seleccionar un lenguaje para su aplicación determinada, deben tenerse en consideración algunos factores importants para hacer una selección apropiada.

Módulo: Inicio				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Verifica Existencia Ratón	Realiza la verificación de conexión del ratón a la computadora.	Ninguno	Respuesta de conexión	Ninguno
Inicialización del cursor del ratón	Activa el cursor del ratón	Respuesta de conexión	Ninguna	Ninguno
Verificación de Archivos	Revisa la existencia de los archivos necesarios del sistema	Ninguno	Error falta archivo	Ninguno

Tabla IV.10 Descripción del Módulo Inicio

Módulo: Menú Principal				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Despliega Menú	Muestra en la pantalla las opciones principales con las que cuenta el menú.	Opción [illegible]	Ninguno	Ninguno
Lee Opción	Espera a que el usuario seleccione alguna de las opciones mostradas en la pantalla permitiéndole mover el cursor a través de ella y reconociendo la opción seleccionada	Ninguno	Ninguno	Selecciona [illegible]

Tabla IV.11 Descripción del Módulo Menú Principal

Módulo: Lee Opción (Menu Principal)				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Selecciona	Resalta en la panta- lla la opción selec- cionada	Opción Actual	Ninguno	Ninguno
Edición	Módulo que permite el despliegue y elec- ción de archivos de datos del sistema	Ninguno	Ninguno	.Despliega Menu Edicion .Lee Opcion Edicion
Manejo de Archivos	Módulo que permite realizar el manejo de algunas opciones del sistema operati- vo para archivos	Ninguno	Ninguno	.Despliega Menu Manejo Archivos .Lee Opcion Manejo Archivos
Métodos	Módulo que contiene los métodos de solu- ción de teoria de redes	Ninguno	Ninguno	.Despliega Menu Metodos .Lee Opcion Metodos
Dibuja Red	Módulo que permite al usuario dibujar redes	Ninguno	Ninguno	.Despliega Menu Dibuja Red .Lee Opcion Dibuja Red

Tabla IV.12 Descripción del Módulo Lee Opción (Menu Principal)

Módulo: Edición				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Despliega Menú	Muestra en la panta-lla las opciones prin-cipales con que cuen-ta el menú.	Opción Inicial	Ninguno	Ninguno
Lee Opción Edición	Espera a que el usua-rio seleccione alguna de las opciones mos-tradas en la pantalla permitiéndole mover el cursor a través de ella y retornando la opción selecciona-da	Ninguno	Ninguno	Selecciona, Edición de Archivos, Despliegue Información de Archivos

Tabla IV.13 [illegible]

Módulo: [illegible]				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Selecciona	[illegible]	[illegible]	[illegible]	[illegible]
Edición de Archivos	Módulo que permite editar [illegible] archivo [illegible]	[illegible]	[illegible]	Ninguno
Despliegue de Información	Módulo que [illegible]	[illegible]	[illegible]	[illegible]

Tabla IV.14 [illegible]

Módulo: Manejo de Archivos				
Procesos		Parámetros		Módulos que llama
Nombre	Descripción	Entrada	Salida	
Despliega Menú	Muestra en la pantalla las opciones principales con que cuenta el menú	opción Inicial	Ninguna	Ninguno
Lee Opción Edición	Espera a que el usuario seleccione alguna de las opciones mostradas en la pantalla permitiéndole mover el cursor a través de ella y retornando la opción seleccionada	Ninguno	Ninguno	.Selecciona .Copia .Borra .Renombra .Directorio .Cambia Dir. [illegible]

Tabla IV.[illegible]

Módulo: [illegible]				
Procesos		[illegible]		[illegible]
Nombre	[illegible]	[illegible]	[illegible]	
Despliega Menú	Muestra [illegible]	[illegible] Inicial	[illegible]	[illegible]

Tabla IV.[illegible]

Módulo: Dibuja Red				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Lee Opción Edición	Espera a que el usuario seleccione alguna de las opciones [illegible] tradas en la [illegible] permitiéndole mover el cursor a través de ella y [illegible] la opción [illegible] da	Ninguno	Ninguno	[illegible]

Tabla IV.16(a) [illegible]

Módulo: Met[illegible]				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Desplieqa Menu	Muestra en la pantalla las opciones principales con que cuenta el menú.	Opción Inicial	Ninguno	Ninguno
Lee Opción Edición	Espera a que el usuario seleccione alguna de las opciones mostradas en la pantalla permitiéndole mover el cursor a través de ella y reconociendo la opción seleccionada	Ninguno	Ninguno	[illegible]

Tabla IV.1[illegible] Módulo Met[illegible]

Módulo: Interfaces (Manejo de Archivos)				
Procesos		Interfaces		Módulos que utiliza
Nombre	Descripción	Entrada	Salida	
Selecciona	Resalta en la pantalla la opción seleccionada	Opción Actual	Ninguno	Ninguno
Copia	Módulo que permite realizar la copia del archivo seleccionado a un archivo nuevo	Opción Actual	Error de Archivo	Ninguno
Borra	Módulo que permite borrar el archivo seleccionado	Opción Actual	Error de Archivo	Ninguno
Renombra	Módulo que permite cambiar el nombre del archivo seleccionado por un nuevo nombre	Opción Actual	Error de Archivo	Ninguno
Directorio	Módulo que despliega los nombres de los archivos que contiene el directorio actual	Opción Actual	Error de Directorio	Ninguno
Cambia Directorio	Módulo que permite cambiar el nombre y uso del directorio actual por el nuevo	Opción Actual	Error de Directorio	Ninguno
S.O. Shell	Módulo que permite salir del sistema temporalmente y regresar a él	Ninguno	Error de Archivo COMMAND	Ninguno
Salir del Sistema	Módulo que permite salir del sistema	Ninguno	Ninguno	Ninguno

Tabla IV.4.- Interfaces del Módulo Interfaces (Manejo de Archivos)

Módulo: Lee Opción (Métodos)				
Procesos		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Selecciona	Resalta en la pantalla la opción seleccionada	Opción Actual	Ninguno	Ninguno
Conectividad	Módulo que contiene el método de solución de conectividad de una red	Ninguno	Ninguno	.Entrada de Datos
Búsqueda	Módulo que contiene los métodos de solución de búsqueda en anchura y búsqueda en profundidad de una red	Ninguno	Ninguno	.Entrada de Datos
Ruta Corta	Módulo que contiene los métodos de solución de Dijkstra y Floyd de ruta corta de una red	Ninguno	Ninguno	.Entrada de Datos
Flujo Máximo	Módulo que contiene el método de solución de Ford y Fulkerson para flujo máximo de una red	Ninguno	Ninguno	.Entrada de Datos
Arbol de Expansión Mínima	Módulo que contiene los métodos de solución de Prim y Prim para el Arbol de expansión mínima de una red	Ninguno	Ninguno	.Entrada de Datos

Tabla IV.9 Descripción del Módulo Lee Opción (Menú Métodos)

Módulo: Entrada Datos				
Procesos		Parámetros		Módulo que Utiliza
Nombre	Descripción	Entrada	Salida	
Selecciona	Resalta en la pantalla la opción previamente elegida	Opción Actual	Ninguno	Ninguno
Archivo	Módulo que realiza la lectura de los datos de entrada de los métodos de solución, almacenados por el usuario en un archivo	Ninguno	Ninguno	Ninguno
Manual	Módulo que realiza la lectura de los datos de entrada de los métodos de solución, dados por el usuario desde el teclado	Ninguno	Ninguno	Ninguno
Crea Archivo	Módulo que crea un archivo con los datos de entrada proporcionados por el usuario desde el teclado			

Tabla IV.20 Descripción del Módulo Entrada Datos

Módulo: Descripción del Módulo [illegible]				
Procesos		Parámetros		Módulo que Utiliza
Nombre	Descripción	Entrada	Salida	
Selecciona	Resalta en la pantalla la [illegible] elegida	[illegible] Actual	Ninguno	Ninguno

Tabla IV.21 Descripción del Módulo [illegible] para [illegible]

Módulo: Ver opción (Utilería-Red)				
Proceso		Parámetros		Módulos que Utiliza
Nombre	Descripción	Entrada	Salida	
Cargar	Módulo que permite cargar el archivo seleccionado a la memoria	Opción Actual	Error de Archivo	Ninguno
Desplegar	Módulo que permite la interacción de memoria en pantalla	Ninguno	Ninguno	Ninguno
Renombra	Módulo que permite cambiar el nombre del archivo seleccionado por un nombre nuevo	Opción Actual	Error de Archivo	Ninguno
Edición de Archivos	Módulo que permite editar el archivo archivo seleccionado	Opción Actual	Error de Archivo	Ninguno
Salvar	Módulo que escribe la información contenida en memoria a un archivo en disco	Opción Actual	Error de escritura	Ninguno

Tabla IV.2(f) Descripción del Módulo Ver Opción (Utilería-Red)

Los lenguajes que pueden ser considerados para la implementación de estos programas de aplicación, deben cumplir con las siguientes características básicas:

- El lenguaje debe ser estructurado para permitir la implementación del diseño descrito anteriormente.
- El lenguaje debe ofrecer la capacidad de acceso a puertos, a memoria y a equipo periférico.
- El lenguaje debe poseer rutinas gráficas básicas que permitan escribir los programas de graficación eficientemente.

- Puesto que el proyecto se desarrolla en ambientes de microcomputadoras PC, debe existir un compildador que permita manejar dicho lenguaje en este equipo.

Existen varios lenguajes que cumplen con las características mencionadas anteriormente, sin embargo, se optó por el lenguaje PASCAL, ya que además de cumplir con estos requisitos, es un lenguaje muy difundido en el ambiente académico universitario y fácil de entender para el programador, los cuales son características muy importantes para futuras modificaciones, correcciones y adiciones a los programas desarrollados.

Particularmente se usa el compilador Turbo Pascal, que es el compilador del lenguaje Pascal más eficiente y difundido actualmente en el mercado para ambientes de microcomputadoras PC. Por conveniencia se usa la versión más reciente a la fecha del desarrollo de los programas, es decir, la versión Turbo Pascal 6.0.

IV.5.2 DESCRIPCION DE LOS PROGRAMAS.

A causa de la vasta aplicación de modelos formulados como modelos matemáticos de redes y a la valiosa ayuda que proporcionan, los métodos de solución descritos en esta sección son fundamentales para obtener la solución óptima a estos modelos.

Este programa cuenta con un programa de manejo de archivos, un programa de gráficos, un programa de edición y ocho programas de métodos de solución para problemas formulados como modelos matemáticos de redes.

El programa de manejo de archivos realiza algunas de las funciones que realiza el sistema operativo como son: copiar un archivo, borrar un archivo, renombrar un archivo, desplegar el nombre de los archivos contenidos en un directorio, realizar un cambio de directorio, salir temporalmente del sistema y salir del sistema. Se cuenta con la ventaja de poder realizar estas funciones sin tener que salir del sistema para verificación y respaldo de los archivos de datos del usuario , los cuales son necesarios para obtener la solución de los modelos planteados por el usuario; los archivos de gráficos creados por el usuario o de cualquier otro archivo que necesite.

El programa de edición de archivos realiza el despliegue, modificación y creación de archivos de datos que fueron creados por el sistema y los cuales tienen una extensión **.NET** .

El programa de gráficos realiza el despliegue y creación de gráficos de redes que proporcionan una mejor visualización del problema formulado como modelo de redes por el usuario. El programa realiza el despliegue del gráfico una vez que este se encuentre cargado en memoria, si el usuario necesita realizar alguna modificación, está puede ser realizada con el comando editar y después salvar la nueva gráfica, también cuenta con la opción de renombrar algun archivo gráfico.

A continuación se detallan los métodos de solución de modelos de redes.

a) Conectividad de una Red

El propósito de este método es el de verificar si la red modelada es conexa o no conexa, esto es, verifica si existen trayectorias en toda pareja de nodos que los une. En pantalla aparece la opción de entrada de datos (datos de nodos y trayectorias entre ellos), la cual puede realizarse por medio de un archivo o del teclado, una vez introducidos los datos se realiza el proceso de verificación de conectividad y se despliega el resultado obtenido en pantalla o directamente en impresora de acuerdo a la elección del usuario.

b) Busquedas.

En está parte se cuenta con dos opciones de busquedas en una red, esto es, realiza la examinación de sus nodos y trayectorias. El primero de estos programas es la *Busqueda en Anchura* y el segundo es la *Busqueda en Profundidad*. En pantalla aparece la opción de selección de alguno de estos métodos, una vez realizada la selección aparece en pantalla la opción de entrada de datos (datos de nodos y trayectorias entre ellos), la cual puede realizarse por medio de un archivo del teclado, una vez introducidos los datos se realiza el proceso de busqueda y se despliega la trayectoria obtenida en pantalla o directamente en impresora de acuerdo a la elección del usuario.

c) Ruta Corta.

En está parte se cuenta con tres opciones de encontrar la ruta más corta una red, esto es, encuentra el camino mínima en la red de un nodo a otro. El primero de estos programas es encontrar la *Ruta mas Corta* de un nodo inicial a un nodo final, el segundo es encontrar la *Ruta mas Corta* de un nodo inicial a todo nodo por el método de Dijkstra, finalmente el tercer programa es encontrar la *Ruta mas Corta* entre todo par

de vértices por medio del método de Floyd. En pantalla aparece la opción de selección de alguno de estos métodos, una vez realizada la selección aparece en pantalla la opción de entrada de datos (datos de nodos y trayectorias entre ellos), la cual puede realizarse por medio de un archivo o del teclado, una vez introducidos los datos se realiza el proceso de encontrar la ruta mas corta y se despliega la trayectoria obtenida en pantalla o directamente en impresora de acuerdo a la elección del usuario.

d) Flujo Máximo.

En está parte se cuenta con el método de flujo máximo, el cual proporciona la maximización de una red. La maximización de una red se llevará a cabo por medio del método de Ford Fulkerson. En pantalla aparéce la opción de entrada de datos (datos de nodos y trayectorias entre ellos), la cual puede realizarse por medio de un archivo o del teclado, una vez introducidos los datos se realiza el proceso de encontrar el flujo máximo y se despliega la trayectoria obtenida en pantalla o directamente en impresora de acuerdo a la elección del usuario.

e) Arbol de Expansion Minima.

En esta parte se cuenta con dos opciones de encontrar el árbol de expansión mínima una red, esto es, encuentra el camino en forma de arbol mínimo en la red de un nodo inicil a un nodo final. El primero de estos programas es encontrar el *Arbol de Expansion Minima* medio del método de Fruskal, el segundo es encontrar el*Arbol de Expansion Minima* medio del método de Prim. En pantalla aparecera opción de entrada de datos (datos de nodos y trayectorias entre ellos), la cual puede realizarse por medio de un archivo o del de encontrar la ruta mas corta y se despliega la trayectoria teclado, una vez introducidos los datos se realiza el proceso obtenida en pantalla o directamente en impresora de acuerdo a la elección del usuario.

IV.6 PRUEBAS

Una vez concluido el proyecto se hicieron algunas modificaciones en los módulos de gráficos y métodos, sin que los demás módulos fueran afectados debido al diseño modular del diseño.

Las modificaciones se vieron reflejadas en los programas de los módulos, ya que se obtuvo una mayor velocidad de proceso en los resultados obtenidos en el módulo Métodos.

Cuando se concluyó el desarrollo de los primeros prototipos de cada uno de los métodos de solución se fueron realizando algunas pruebas preliminares con cada uno de ellos. Se realizó una prueba preliminar para los métodos con algunos modelos de redes obtenidos de algunos libros de teoría de redes. Antes de realizar la prueba se solucionaron manualmente estos modelos para verificar la exactitud de los resultados obtenidos. Dichas pruebas fueron realizadas con los programas preliminares del sistema, luego de realizar estas pruebas se estimó que los resultados obtenidos no eran exactos ya que algunos de las trayectorias de la solución no eran correctas en tres de los métodos.

Posteriormente se realizó otra prueba, una vez modificados los programas, con modelos de redes proporcionados por personas que trabajan con datos reales de modelos de redes eléctricas, modelos de tendido de tuberías de agua, modelos de transporte y algunos otros modelos más. Despues de realizadas estas pruebas se estimó que los resultados obtenidos fueron satisfactorios y los métodos de aplicación habían sido aplicados correctamente para el fin propuesto.

Una vez verificados los programas de los métodos de solución se procedió a realizar una prueba con un usuario, la cual arrojo las siguientes modificaciones al sistema:

- Debido a que el sistema maneja datos de entrada de los modelos de redes, en algunas ocasiones los modelos implican un gran manejo de datos, los cuales al momento de introducirlos manualmente una sola vez no es tedioso, pero si se quiere solucionar otra vez este modelo es tedioso volver a introducir todos los datos que necesita el modelo. Para corregir esto se implemento el uso de archivos de datos para los modelos de redes, los cuales se crean una sola vez y pueden utilizarse cuantas veces se quiera.

- Como el sistema maneja archivos se introduce el manejo de archivos al sistema, para que el usuario no tuviera el problema de salir del sistema para verificar la existencia de sus archivos, crear sus archivos o ver el contenido de estos, para saber cual es el archivo que debe utilizar.

- Por sugerencia de algunos usuarios que se prestaron a las pruebas, se introduje el módulo de graficación, el cual permite al usuario realizar el dibujo de las gráficas de su modelo y las gráficas de resultados, el cual favorece a una mejor conclusión sobre los resultados obtenidos.

IV.7 PROYECTOS DE APLICACION

Como se menciono el la sección VI.2, la teoría de redes tiene una aplicción extensa en diversos campos. En estos campos se pueden modelar innumerables situaciones como modelos matemáticos de redes.

Algunos de estos de estos proyectos son: sistemas de transporte, sistemas de tráfico urbano, sistema de transporte colectivo, sistemas de [illegible], redes eléctricas, sistemas de reemplazo de equipo, sistemas de inventarios, sistemas de flujo de dinero, sistemas de presas, sistemas de tuberías, sistemas de oleoductos, sistemas de asignación de recursos y otros modelos más que se pueden implementar por medio de este tipo de modelos.

Como es visto la aplicación de este tipo de modelos es muy amplia y puede aplicarse a diversas áreas de trabajo, de una manera sencillo y obtener los resultados óptimos de forma rápida.

CAPITULO

CINCO

CONCLUSIONES

V.1 INTRODUCCION

Cada día cobra mayor importancia el uso de una computadora y es común encontrarse con una gran variedad de aplicaciones en la vida diaria, tal es el caso del pago de impuestos, control de nóminas, control ferroviario, control aéreo, etc.; con los que estamos relacionados de alguna manera.

Las aplicaciones que se le han dado a la computadora abarcan diversos aspectos que pueden ir desde resolver problemas simples has problemas con un alto grado de dificultad.

Algunas aplicaciones de ello pueden ser de los siguientes tipos:

ADMINISTRATIVOS: los problemas de este tipo se caracterizan principalmente por el manejo de múltiples datos y pocos cálculos. Dicha aplicación agiliza las operaciones en los negocios proporcionando mayor confiabilidad y ayuda en la toma de decisiones.

La importancia de la computación en está ha ido creciendo debido a que cada vez se manejan mayores volúmenes de información que debe ser registrada rápida y eficazmente. La computadora puede llevar controles de la contabilidad, de los inventarios, facturar , controlar las ventas, procesar la nómina y los cheques de pago d los empleados, controlar presupuestos , la producción y e transporte de artículos.

A continuación se describirán algunos ejemplos de este tipo de aplicaciones:

Sistemas Bancarios: las transacciones que se realizan en las instituciones bancarios deben mantenerse al día, por lo que que necesario actualizar rápidamente los saldos de los cuenta-habientes. La automatización en los bancos es indispensable y exige un sistema de cómputo conectado con todas sus sucursales.

Nómina: La puntualidad de los pagos en una empresa con un número considerable de empleados y la exactitud en el cálculo de su percepción neta, hacen necesaria la ayuda de un sistema de nómina computarizado, que considerando la percepción bruta hace las deducciones exigidas y adiciona las comisiones, horas extras, etc.

Control de inventarios: En las empresas en la que existe constante flujo de artículos, tal es el caso de las tiendas comerciales, es necesario llevar un control estricto de las entradas y salidas de mercancía, devoluciones, registro de nuevos artículos, etc. La

realización de estas operaciones en forma manual resulta costoso e implica retrasos, por lo que el uso de la computadora agiliza la actualización del inventario permitiendo pronósticos de ventas y requisiciones en forma eficinte u oportuna.

Contabilidad. Todo negocio requiere de información que refleje su situación, pero para que ésta sea verdaderamente útil es necesario que sea oportuna, veraz y confiable. Tal es el objetivo, de una empresa que se maneja un gran número de transacciones contables, es difícil alcanzarlo llevando controles manuales, por lo que cada vez es mayor el número de negocios que lo adeptan un sistema de contabilidad por computadora permitira procesar y almacenar grandes volúmenes de información y obtener oportunamente una gran variedad de reportes.

TECNICOS Y CIENTIFICOS. En el área técnica o científica se manejan múltipes operaciones de calculo y relativamente poca información. Actualmente los usos de la computadora en este ámbito son diversos, dentro de ellos puedes mencionarse:

- Simulación de componentes electricos y electrónicos.
- Diseño de puentes, carreteras y edificios.
- Control de trafico.
- Diagnósticos medicos.
- Pronósticos meteorologicos.
- Aplicaciones militares.
- Control de trayectorias de satélites.
- Simulación de bacterias.
- Simulación de vuelos aereos.
- Y otros más.

OTRAS APLICACIONES. Conforme la computación se ha ido desarrollando, su aplicación se ha extendido a otras áreas participando en actividades artisticas, recreativas y deportivas.

Como ejemplo de la participación de la computación en el arte se puede mencionar las melodias tocadas por una computadora, imitando instrumentos musicales.

En lo recreativo nos encontramos con computadoras programadas con diversos juegos, que proporcionan momentos de distracción y reto para los jugadores.

En las actividades deportivas un ejemplo del uso de la computadora en este campo es la evaluación de la condición física de los deportistas.

V.1. ALCANCES

Debido a que las aplicaciones de una computadora se ha ido incrementando considerablemente en diversas áreas, ahora la utilizamos en la aplicación de problemas de redes, los cuales por medio de la computadora se podrán realizar de una manera más ágil

Como el área de las redes es aplicable a diversos campos como planeación y administración entre otros, entonces las *perspectivas* que hay para este sistema en el futuro son muy buenas ya que estos dos campos, son campos que no tienen un límite determinado en sus alcances.

Debido a que día con día se planean cosas nuevas, se va teniendo la inquietud de mejorar cosas ya hechas, para lo cual se necesita planear su mejoramiento, en tal caso se podría utilizar la teoría de redes aplicada a ese trabajo, por lo que el sistema les servirá para realizar este planeamiento de mejoración de una manera más ágil.

De igual modo en el área de administración se pueden aplicar la teoría de redes por lo que con ayuda de este sistema se podrán realizar los trabajos de administración enfocados a la teoría de redes de una forma más rápida.

Otros problemas que se pueden resolver por medio de la teoría de redes son:

- Tráfico urbano, problema que existirá durante un largo tiempo por lo que este sistema que ayuda a resolver este problema podrá aplicarse durante un largo tiempo.

- Redes Eléctricas, el tendido de redes eléctricas tiene un ciclo de vida por el momento duradero, ya que la necesidad de la luz hoy en día es indispensable, con este sistema se puede encontrar una solución óptima para saber cual es la ruta que conviene para llevar la energía eléctrica a un determinado lugar o a determinados lugares de una forma menos costosa.

Como los ejemplos anteriores podríamos numerar muchos más en los cuales la aplicación de redes es fundamental para tener la solución más eficaz y óptima de algún problema.

La utilización de computadoras es cada vez más común en todas las áreas. Actualmente, debido a la disminución de su costo y al aumento de marcas en el mercado, es más fácil adquirirlas, por lo que la mayoría de las empresas medianas y pequeñas cuentan con una, y tienden a automatizar sus procesos.

Por esta razón, es importante que las empresas cuenten con una computadora y un sistema que les ayude en sus trabajos, considerando que esta herramienta es un dispositivo que nos ayuda a obtener informes de una forma más rápida y oportuna para la correcta toma de decisiones.

V.2 LIMITACIONES

Este sistema tiene algunas limitaciones, como la mayoría de los sistemas, pero estas limitaciones no impiden que se aproveche a toda su capacidad.

Algunas de sus limitaciones son que las redes que se vayan a trabajar sean mayores a 100 nodos, una limitación que hasta el momento no se ve como tal ya que hasta hoy en día no se ha trabajado con una red tan grande, que dado el caso en que se llegue a utilizar una red de este tamaño, se cambiarían algunos parámetros en el sistema mientras la memoria de la computadora alcance.

Esta es la mayor que tiene el sistema, ya que se puede resolver cualquier problema enfocado a la teoría de redes por medio de este sistema, ya que con lo que cuenta se pueden resolver planteándolos de acuerdo a los problemas que se solucionan en este sistema, lo cual es factible sin perder la esencia del problema original.

V.3 CONCLUSIONES

La computadora se ha convertido en una herramienta prácticamente indispensable para el avance en muchas áreas de conocimiento. Además de servir como herramienta de apoyo, ha dado una nueva visión del mundo. Se ha logrado un avance intentando que una computadora realice funciones hasta ahora consideradas exclusivas del ser humano como oír, hablar o pensar. Estos intentos han logrado que conozcamos ahora, mas de nosotros mismos.

En este trabajo se ha planteado la solución a modelos de sistemas en forma de modelos matemáticos de redes a través de la computadora. Los programas que se presentan en este trabajo fueron desarrollados siguiendo un diseño estructurado por lo que el

mantenimiento de los mismos se puede realizar de una manera rápida y fácil. En la implementación y programación se establecieron algunos parámetros para mejor identificación de variables, constantes, funciones y procedimientos además de documentar su función y uso.

El sistema de programas resultante se puede clasificar como un sistema con acoplamiento por datos, pues los módulos (como se indica en el diagrama de flujo de datos y en las cartas estructuradas) se comunican a tavés de datos además de que posee una cohesión funcional pues en cada módulo sus elementos están dedicados a una sola función.

Los programas han sido terminados para cumplir con los objetivos y requerimientos que fueron planteados al inicio del proyecto. Si consideramos que un sistema está terminado cuando ya no tiene más adiciones o modificaciones que realizarle, el sistema aquí presentado no está terminado totalmente, sino parcialmente ya que a este sistema todavía puede adicionársele otros métodos de soluciones de redes. De hecho ningún sistema, viéndolo así, podría considerarse terminado cuando cumple con los requerimientos mínimos necesarios para entrar en operación y se requiera un tiempo razonable para irle realizando modificaciones constantemente, entonces el sistema presentado está terminado y cumplió con los objetivos y requerimientos planteados.

APENDICE A

TEORIA DE REDES

Aunque a primera vista podría parecernos que no existe relación alguna entre la investigación de operaciones y las actividades bélicas, la verdad es que el surgimiento de la investigación de operaciones se da precisamente durante la Segunda Guerra Mundial por necesidades eminentemente militares. Desde luego, no fueron militares propiamente quienes desarrollaron este campo o Area de las matemáticas: fueron grupos de científicos dedicados a estudiar y determinar el mejor uso posible de los elementos bélicos quienes hicieron posible el marco conceptual y teórico que hoy denominamos investigación de operaciones.

Es importante destacar que las bases para el desarrollo de la técnicas de investigación de operaciones son el trabajo matemático y lógico realizado por el hombre, a través de una cadena de logros y resultados que desemboca en la elaboración de elementos matemáticos cada vez más poderosos. Es por ello que la creación de la investigación de operaciones debe atribuirse no solo a aquellos que le dan luz a partir de mediados de este siglo, sino a todos aquellos que directa o indirectamente participaron en la estructuración de un mejor lenguaje matemático.

La realidad nos muestra una gran cantidad de hallazgos que confirman este hecho; el transistor, tan usado actualmente en multitud de equipos electrónicos, es producto de la carrera espacial emprendida por las grandes potencias; la computadora electrónica digital, elaborada con la mira de resolver problemas matemáticos que demandaban enorme cantidad de cálculos, que tiene el día de hoy una infinidad de aplicaciones diferentes.

Parece ser que el establecimiento de mejores controles en el manejo de los diversos problemas se dan en distintos contextos y se facilita notablemente a través de las técnicas de investigación de operaciones manejadas por computadora, de ahí que cada vez sea mas importante el estudio de una y otra para atacar aquellas situaciones que requieren de un mejor control de optimización de recursos.

La investigación de operaciones es un conjunto de técnicas matemáticas destinadas a optimizar el manejo de recursos en un contexto dado. Por optimizar debemos entender: encontrar la mejor posibilidad o alternativa; sin embargo ello depende de gran medida de la realidad en que operemos.

La investigación de operaciones tiene una seria limitante en la práctica; a través de sus diferentes métodos es posible enfocar problemas para darles una solución que tienda a ser óptima, aunque en algunos casos no sera posible llegar a tal solución.

Sin embargo, la investigación de operaciones podrá acercarnos a ella mediante soluciones que sean factibles y muestren ser realmente ventajosas.

La investigación de operaciones esta dirigida a la realización de análisis adecuados para tomar buenas decisiones; esto es importante porque lo único que podemos controlar son las decisiones que previamente tomemos para la obtención de un cierto resultado desconocido.

Cuando aplicamos la investigación de operaciones describimos algun sistema o conjunto de elementos por medio de un modelo, que será sometido a una serie de operaciones para determinar su comportamiento; de esta manera el modelo nos permitirá observar su funcionamiento bajo diversos estados, con el propósito de seleccionar aquel tipo de comportamiento que a nuestro juicio sea el óptimo o al menos el mejor de entre los obtenidos. Esto es, nos permitirá seleccionar la mejor forma de operación del sistema de estudio.

El que lleguemos a una solución óptima dependerá tanto de la técnica que hay que emplear, como del problema que se desee resolver; esto ultimo sera generalmente lo que delimite la posibilidad de obtener una solución insuperable.

Por su propia estructura en cierto tipo de problemas, algunas técnicas de investigación de operaciones difícilmente nos llevan a una solución única; entonces el buen juicio y el acervo de conocimientos del diseñador son los que entrarán en juego para lograr el mejor resultado.

La teoría de redes, puede aportar una ayuda muy eficaz en el tratamiento de ciertos problemas de carácter combinatorio que aparecen en diversos dominios económicos, sociológicos o tecnológicos; de hecho, notamos que la "teoría de redes" tiene un contenido de tal riqueza que amerita un sitio muy importante en la enseñanza.

En entre las ramas de la teoría de los conjuntos la que promete dar mas frutos tanto para el matemático puro como para el ingeniero, el organizador, el biólogo, el psicólogo y muchos otros.

Se pueden representar estructuras diversas mediante una red, por ejemplo:

1.- Un sistema de caminos o de calles.

2.- Un sistema eléctrico.

3.- Presas.

4.- Tuberías.

Algunos conceptos importantes utilizados en la teoría de redes son los siguientes:

Camino. Es una sucesión de arcos adyacentes que permiten pasar de un nodo a otro siguiendo los arcos.

Circuito. Es un camino en el cual el nodo inicial coincide con el nodo final.

Longitud de un camino o circuito. Es el número de arcos en un camino o circuito.

Lazo. Un lazo es un circuito de longitud 1.

Red Simetrica. Si un nodo X esta conectado a un vértice Y, entonces Y debe estar conectado con X. Si esta propiedad se verifica entre todos los vértices entre los que existe una correspondencia, entonces la red es simétrica.

APENDICE B

ELEMENTOS DEL ANALISIS Y DISEÑO ESTRUCTURADO.

I. ANALISIS ESTRUCTURADO.

El análisis estructurado tiene como prósito fundamental especificar en la forma más precisa posible, los requerimientos para un programa o conjunto de programas.

El diagrama de flujo de datos (DFD) es la principal herramienta gráfica del análisis estructurado y tiene como objeto mostrar las transformaciones de los datos a medida que éstos fluyen a través de los procesos, es decir, ayuda a analizar los cambios que ocurren a los datos de entrada a fin de lograr la salida deseada.

El diagrama de flujo de datos, debido a su sencillez y a que es una herramienta gráfica, resulta comprensible tanto para el usuario como para el analista del sistema.

El diagrama de flujo de datos.

Un diagrama de flujo de datos es un instrumento de modelación que permite enfocar a un sistema como una red de subsistemas conectados unos a otros mediante flujos de datos están relacionados entre sí.

Un diagrama de flujos de datos consta de los siguientes elementos:

a) Un círculo, con un nombre inscrito, para indicar un proceso. El nombre indica la función o el proceso, el cual actua sobre los datos para transformarlos o generar nueva información.

b) Una flecha, con un nombre asociado, para indicar la entada y salida de datos a un proceso. La dirección de la flecha indica la dirección del flujo de datos.

c) Dos líneas paralelas, con un nombre entre ellas para indicar un contenedor de datos, es decir, un archivo en disco o en cinta, o un archivo de tarjetas, etc.

Se muestra un archivo cuando se tiene acceso más de una vez a una pieza de información (mismo dato) y cuando la información se utiliza en un orden diferente al que fue registrada.

Un archivo con una flecha saliente indica que se extrae información de él. Con una flecha entrante indica escritura de información.

d) Un rectangulo que indica donde se origina o se destina la información (sentido de la flecha). Un mismo rectángulo puede ser fuente y destino. El rectángulo tiene un nombre que identifica la fuente o el destino.

Obsérvese que con un diagrama de flujo de datos se obtiene una estructura del tipo entrada-proceso-salida, es decir, datos de entada a un proceso que actua sobre ellos y datos de salida transformados.

Características de un diagrama de flujo de datos.

Las características de un diagrama de flujo de datos son las siguientes:

a) Es gráfico: La ventaja de una herramienta gráfica consiste en su impacto visual, es decir, que a simple vista se perciben rápidamente las funciones principales del sistema. El diagrama de flujo de datos debe usarse de una manera concisa para evitar que el usuario pierda interés en el.

b) Es modular: Esto significa que el diagrama de flujo de datos muestra la partición de un sistema de funciones tan independientes entre si como sea posible, lo cual permite, tant al usuario como al diseñador, revisar cada función del sistema de una manera aislada.

c) Enfatiza el flujo de datos: El diagrama de flujo de datos muestra solamente el flujo de datos que se transforma a medida que pasa a los procesos (funciones) desde la entrada a la salida.

d) Desenfatiza el flujo de control: El diagrama de flujo de datos no muestra información de control (banderas) ni secuencia de acciones en el tiempo.

e) Es modificable: Esto significa que se pueden reconsiderar algunas partes del diagrama de flujo de datos con las cuales no se haya quedado satisfecho y volver a trabajarlas.

f) No es redundante: Esto quiere decir que una función debe registrarse sólo una vez para que el sistema, al cual dará origen el diagrama de flujo de datos, sea consistente y de fácil actualización.

La construcción del diagrma de flujo de datos se hace teniendo en cuenta aspecto funcional del problema, así como la secuencia que se da entre estas funciones.

La forma de proceder de las funciones mayores a las funciones más básicas en el diagrama de flujo de datos es una característica que lo convierte en un instrumento de gran flexibilidad en el modelado de sistemas más complejos.

II. DISEÑO ESTRUCTURADO

La herramienta principal del diseño estructurado es la carta estructurada, la cual muestra la partición del sistema en módulos y la relación jerárquica entre módulos y la relación jerárquica entre estos. Además muestra los flujos de datos y control entre módulos.

Elementos de una carta estructurada.

Una carta estructurada cuenta con los siguientes elementos:

a) Un rectángulo con un nombre inscrito para indicar un módulo. El nombre indica la función del mismo.

b) Líneas que indican la liga entre módulos (llamadas a módulos).

c) Flechas que indican el flujo de datos y el control respectivamente (comunicación ente módulos). Un ejemplo de flujo de control lo constituye una bandera.

d) Un módulo representado como la siguente figura:

significa que es un módulo predefinido. Un ejemplo de un módulo predefinido lo constituyen los subprogramas de biblioteca.

e) El nombre del módulo debe resumir los nombres de sus subordinados inmediatos o resumir su función y las funcines de sus subordinados inmediatos.

Atributos basicos de los modulos.

Un módulo tiene cuatro atributos básicos:

a) Entrada. Los datos que le pasa quién lo invoca.
Salida. Los datos que regresa a quién lo invoca.

b) Funcion. Lo que hace a sus datos de entrada para producir sus datos de salida.

c) Mecanica. Como realiza su función, es decir, su lógica.

d) Datos internos. Su propio espacio de trabajo, es decir, las variables locales.

La carta estructurada se deriva del diagrama de flujo de datos. Como regla general, una carta estructurada muestra a su izquierda los módulos de entrda, al centro los módulos de procesos y a la derecha los módulos de salida.

Caracteristicas de la carta estructurada

Una carta estructurada muestra:

a) La particion del programa, es decir, los modulos de que consta.

b) La estructura jerárquica, es decir, la relación entre módulos.

c) los nombres de modulos y por consiguiente su función.

d) El grado de acoplamiento entre módulos.

e) Las decisiones e iteraciones que involucran la llamada a un módulo.

Una carta estructurada no muestra:

a) El número de veces que se llama a un modulo

b) La secuencia en que se llaman los modulos.

c) Cómo realiza su función.

d) Datos internos del módulo.

Acoplamiento y *Cohesion*.

Acoplamiento. Es una medida cualitativa que se refiere a la interrelación entre módulos de un sistema. Los criterios de acoplamiento se refieren a los parámetros, el número y forma en que se llaman, a las áreas comunes de datos, a las variables de control (banderas), etc.

A continuación se describen tres tipos de acoplamiento. El orden en que aparecen corresponde al grado de calidad del tipo de acoplamiento, comenzando con el mejor.

Acoplamiento por datos Dos modulos están acoplados por datos si se comunican por datos que no sean banderas, ni arreglos, ni registros.

Acoplamiento por estampilla Dos módulos están acoplados por estampilla si se comunican mediante registros o arreglos.

Acoplamiento por control Dos modulos estan acoplados por control si se comunican al menos por bandera.

Cohesion. Es la medida del grado de asociación de los elementos de un mismo módulo.

Un elemento puede ser: Una instrucción, un grupo de instrucciones o una llamada a otro módulo.

Existen varios tipos de cohesión, se describen tres tipos de cohesión partiendo el mejor.

Cohesion funcional. Un módulo con cohesión funcional es aquel en que todos los elementos contribuyen a una y sólo una tarea.

Cohesion secuencial. Un módulo con cohesión secuencial es aquel en que los datos de salida de un elemento sirven como datos de entrada a otro elemento.

Cohesión comunicacional. Un módulo con cohesión comunicacional es aquel cuyos elementos contribuyen a tareas diferentes pero cada tarea tiene los mismos parámetros de entrada y salida.

Otros tipos de cohesion son: cohesión concidental, cuando se hace un módulo para no repetir un código o bien un bloque muy largo se fragmenta en módulos pequeños; cohesión lógica, si existe relación en módulos generales como validación; cohesión temporal, que es igual a la lógica pero relacionada con el tiempo.

BIBLIOGRAFIA

- Aho, A., Ullman, J., "Data Structures and Algorithms", Addison-Wesley, 1981.
- Bazaraa, M.S., Jarvis, J.J., " Programación Lineal y Flujo en Redes", Limusa, 1981.
- Bondy, J.A., Murty, U.S.R., "Graph Theory with Applications", The Macmillan Press LTD, 1977.
- Busaker, R.G., Saaty, T.L., "Finite Graphs and Networks: An Introduction with Applications", Mc Graw-Hill, 1965.
- Carré, B., "Graphs and Networks", Oxford University Press., 1979.
- Christofides, N. "Graph Theory: An Algorithmic Approach", Academic Press., 1975.
- Denardo, E., " Dynamic Programming", Prentice-Hall, Inc., 1982.
- Gondran, M., Minoux, M., "Graphs and Algorithms", John Wiley and Sons, 1984.
- Hadley, G., "Linear Programming", Addison-Wesley, 1967.
- Hillier, F.S., Lieberman, G.J., "Introduction to Operations Research", Holden-Day, Inc., 1980.
- Hu, T.C. "Integer Programming and Network Flows", Addison-Wesley, 1969.
- Jensen, P.A., Barnes, J.W., "Network Flow Programming", John Wyley and Sons, 1980.
- Kauffman, A., "Metodos y Modelos de la Investigación de Operaciones", CECSA, 1968.
- Kauffman, A., "Metodos y Modelos de la Programación Dinámica", CECSA, 1968.

- **Kennington, J.L., Helganson, R.V., "Algorithms for Network Programming", John Wyley and Sons, 1980**

- **Lipschutz, S., "Estructura de Datos", Mc Graw-Hill, 1987.**

- **Minieka, E., "Optimization Algorithms for Networks and Graphics", Marcel Dekker, 1978.**

- **Phillips, D.T., Ravindran, A., Solberg, J.J., "Operations Research: Principles an Practice", John Wyley and Sons, 1976.**

- **Prawda, J., "Metodos y Modelos de Investigación de Operaciones", (Vol.1), Limusa, 1981.**

- **Rivvet, P., "Principles of Model Building", John Wyley And Sons, 1972.**

- **Rockafellar, R.T., "Network Flows and Monotropic Optimization" John Wyley and Sons, 1984.**

- **Taha, H., "Investigacion de Operaciones", Representaciones y Servicios de Ingeniería, S.A., 1981.**

- **Tarjan, R.E., "Data Structures and Network Algorithms", Society for Industrial and Applied Mathematics, 1983.**

- **Tenenbaum, A., Augenstein, M., "Estructura de Datos en Pascal", Prentice-Hall, 1986.**

- **Tremblay, J., Sorenson, "An Introduction to Data Structure With Applications", Mc Graw-Hill, 1984.**