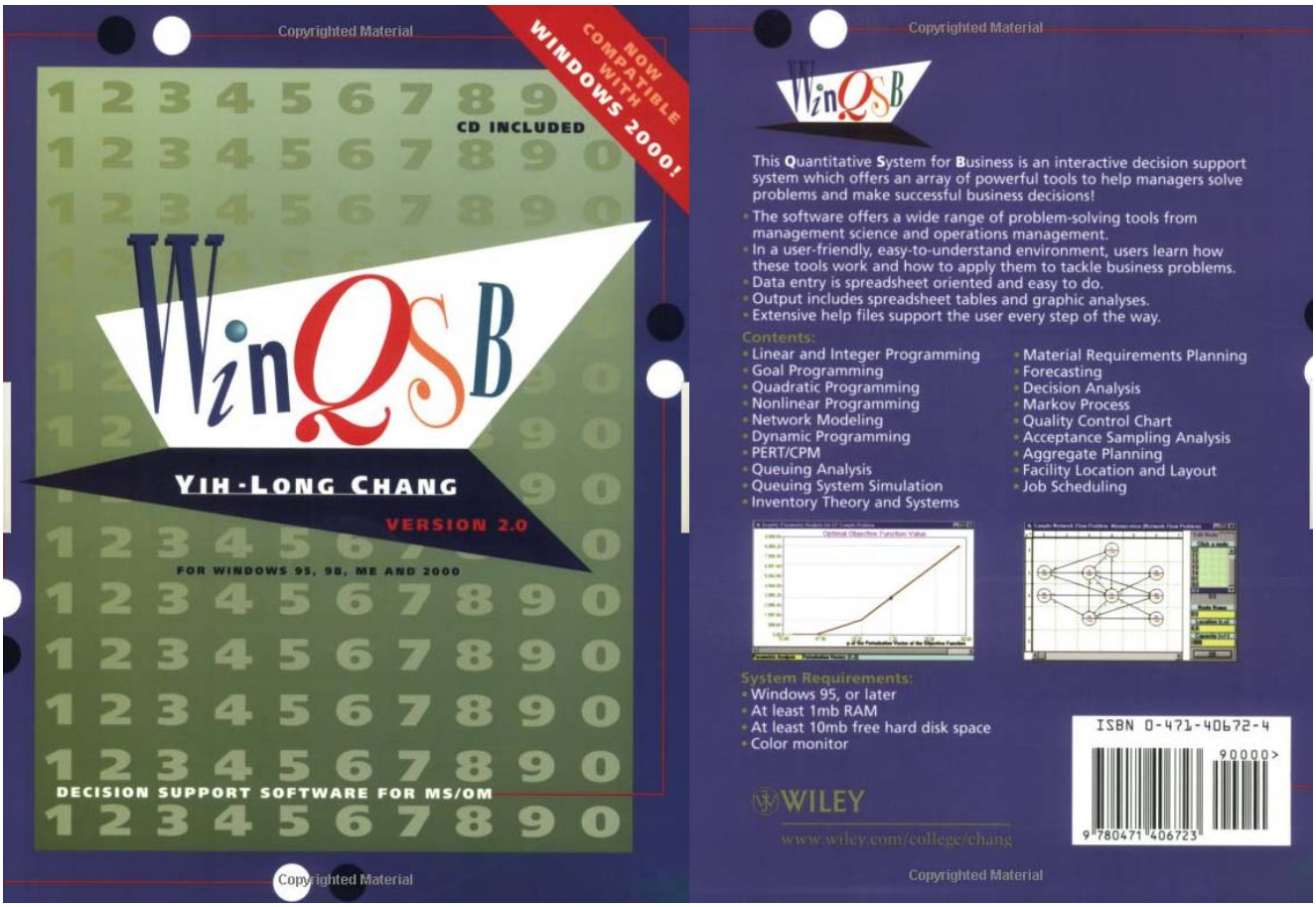
MANUAL DE USO DEL *WinQSB*



PROLOGO

La toma de decisiones en los distintos niveles de las organizaciones cada vez es de mayor complejidad, dadas las crecientes restricciones de disponibilidad de todo tipo de recursos. Los academícos se ha preocupado de investigar y proporcionar herramientas que faciliten a los gerentes el abordaje de estos procesos, teniendo en cuenta que no es recomendable asumir un curso de acción confiados únicamente en la intuición. La llamada administración científica aboga por el uso de los métodos cuantitativos en la toma de decisiones empresariales; de ahí que en los planes de estudio correspondientes a la formación de profesionales de la ingeniería industrial, la administración en sus diferentes matices, las finanzas y muchas más disciplinas, figuren asignaturas que pretendan que los egresados de estas titulaciones se apropien de un cúmulo de herramientas que les facilite el análisis y la toma de decisiones en situaciones complejas.
Con la popularización de los computadores personales (PC's) han surgido programas y aplicaciones muy completas para el tratamiento de los problemas de gestión mediante herramientas cuantitativas, las que en su conjunto constituyen los métodos de la investigación de operaciones.

Se han utilizando diferentes softwares para resolver problemas de investigación de Operaciones como lo han sido MSS (mathematical Science System), STROM, QSB (Quantitative System Business), TORA(que acompaña al libro de Investigación de Operaciones de Hamdy Taha), Mathematical Programming y Probabilistic Programming (que acompañaba al libro de Introducción a la Investigación de Operaciones de Hillier y Lieberman), Lido, Lingo y otros programas en Excel como lo es el Solver, etc. De igual manera se han realizado esfuerzos individuales de quienes relacionados con la Investigación de Operaciones hemos desarrollado programas para la solución de problemas diversos en este campo.

QSB (***Quantitative System Business***), podría decirse que es el software más utilizado en la actualidad por estudiantes de pregrados y postgrados que incluyen en su plan de estudios asignaturas como la investigación de operaciones o temas relacionados. Sin embargo no existe en nuestro medio una guía en español para el docente y el estudiante, que permita el aprovechamiento máximo de los módulos que contempla la aplicación.
Motivados por esta evidencia se ha conformado un manual resultado de una investigación bibliográfica sobre el manejo de este programa, y escrito en un lenguaje sencillo y accesible a toda clase de público, en el que se explican las principales herramientas que contempla el **WINQSB**. Debido a los temas de investigación de operaciones que se tratan en este libro, únicamente cinco temas, serán abordados en este manual.
Advierto, eso sí, que este libro no es una traducción del manual que se distribuye con el software. Los ejemplos utilizados y la forma de explicarlos, así lo evidencian.
Tampoco estoy frente a un libro de investigación de operaciones o similar, por lo que el lector no podrá esperar encontrar una explicación exhaustiva de los fundamentos teóricos de cada tema; sólo en aquellos que lo consideramos necesario nos detenemos en los fundamentos.

Espero que este manual del ***WinQSB*** que se incluye como apoyo en este libro sirva apropiadamente para el fácil y eficiente cálculo de los ejercicios que en este libro se tratan.

A. INTRODUCCIÓN AL MANEJO DEL WinQSB

El objetivo de esta serie es proveer al alumno de pregrado o postgrado de un manual completo en español sobre el software **WINQSB**, para la solución de una gran cantidad de problemas complejos de tipo cuantitativo.
Este Manual lo introducirá en a la solución de problemas complejos mediante el uso de un software de relativo fácil manejo.

WinQSB es una herramienta poderosa para el manejo de métodos cuantitativos, el cual está conformado por 19 módulos: Este programa contiene los más útiles y populares métodos cuantitativos usados en las ciencias administrativas, investigación de operaciones y administración de operaciones.

WinQSB es una aplicación versátil que permite la solución de una gran cantidad de problemas: administrativos, de producción, de recurso humano, dirección de proyectos, etc.
Debido a su facilidad y potencia de manejo, este libro se convierte en una herramienta indispensable para el estudiante de pregrado o postgrado que participa en materias como la investigación de operaciones, los métodos de trabajo, planeación de la producción, evaluación de proyectos, control de calidad, simulación, estadística, entre otras.

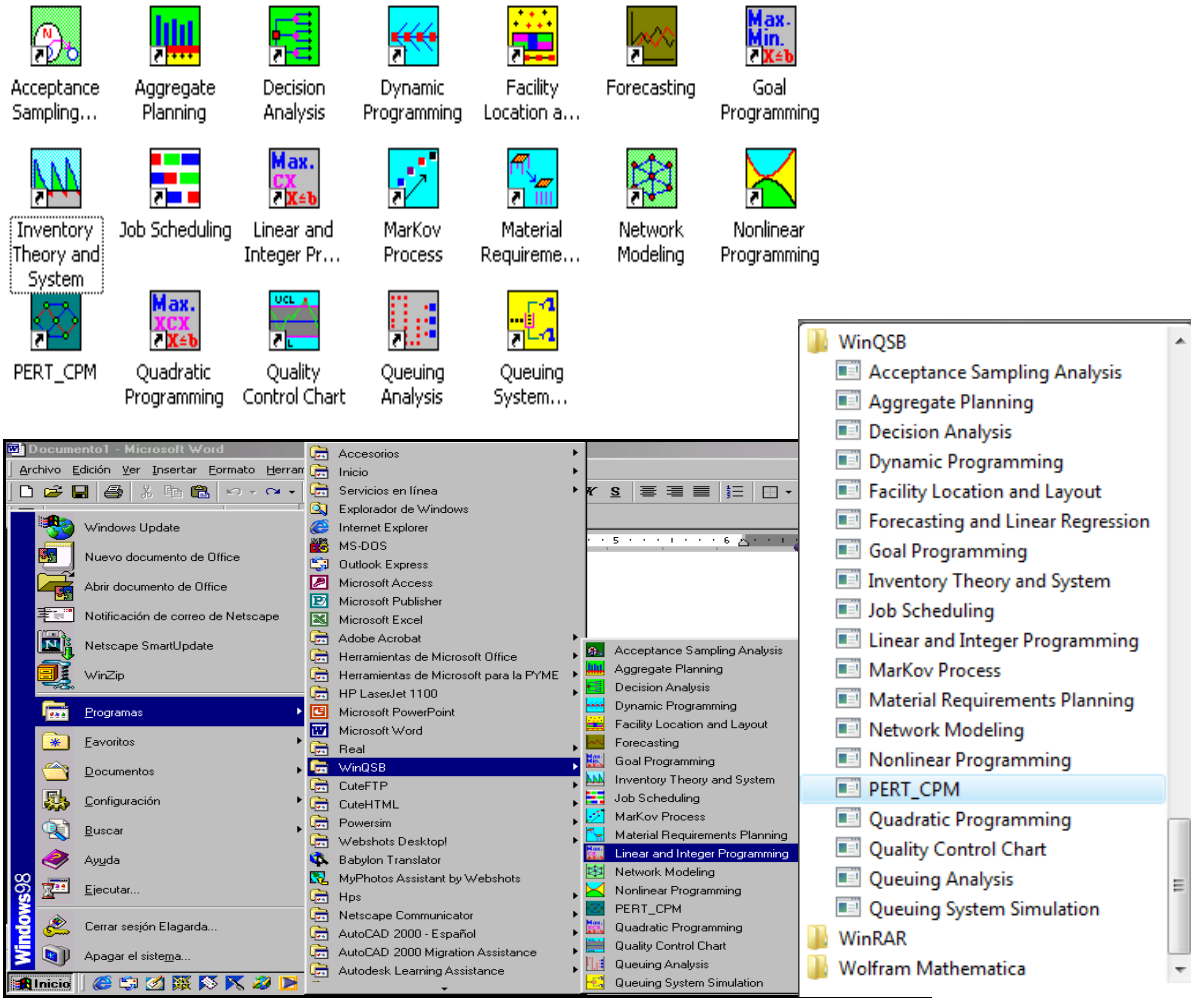
El paquete WinQSB puede usarse con varios objetivos: a) comprobar las soluciones de los problemas de las relaciones; b) resolver problemas grandes y c) realizar experimentos para comprender los conceptos presentados en clase.

Los módulos tratados en este libro son:

- Programación Dinámica
- Teoría de sistemas de colas
- Análisis de Decisiones
- Cadenas de Markov
- Modelos de Redes

No está el lector ante un manual de enseñanza de investigación de Operaciones por lo que supondremos que tendrá las bases teóricas de los módulos aquí referenciados. A diferencia de la versión en ingles que trae el propio programa, este libro conduce el desarrollo de ejemplos completos explicados paso a paso, para que el lector pueda dedicarse más al análisis detallado de la solución de los problemas.

El acceso al **WinQSB** se puede hacer a través del botón ***INICIO*** del sistema operativo WINDOWS, en el menú ***PROGRAMAS*** en la carpeta ***WINQSB***.



Por medio de una interfase interactiva, los profesionales y estudiantes tienen fácil acceso a los diferentes módulos de decisión para resolver una gran variedad de problemas Cada módulo de WinQSB es brevemente descrito a continuación:

Análisis de muestreo de aceptación (*Acceptance Sampling Analysis*)

Acceptance sampling analysis (ASA): Este programa desarrolla y analiza los planes de muestreos de tolerancias para atributos y características de calidad variable.

Planeación agregada (*Aggregate Planning*)

Aggregate planning (AP): Soluciona los problemas de planeamiento agregado a las demandas de satisfacción del consumidor con mínimos o aceptables costos relacionados.

Análisis de decisiones (*Decision Analysis*)

Decision analisys (DA): El programa resuelve 4 típicos problemas de decisión: Análisis Beyesiano, análisis de tablas de rentabilidad, análisis de árbol de decisión y la teoría del juego de cero suma.

Programación dinámica (*Dynamic Programming*)

Dynamic Programming (DP): Resuelve 3 tipos populares de problemas dinámicos: Diligencia, mochila y problemas de planeación de producción e inventarios.

Diseño y localización de plantas (*Facility Location and Layout*)

Facility location and layout (FLL): Este módulo resuelve los problemas de facilidades de localización, disposición funcional y balanceo de línea de producción.

Pronósticos (*Forecasting*)

Forecasting (FC): Este módulo resuelve proyecciones de series de tiempo usando 11 diferentes métodos y además utilizando regresiones lineales de múltiples variables.

Programación por objetivos (*Goal Programming*)

Linear Goal Programming (GP) e Integer Linear Goal Programming (IGP): Este programa resuelve los problemas de GP usando el método simplex modificado o el método gráfico y los problemas de IGP usando el procedimiento branch-and-bound.

Teoría y sistemas de inventarios (*Inventory Theory and System*)

Inventory theory and systems (ITS) : Resuelve problemas de control de inventarios: problemas de cantidades económicas a pedir (EOQ), problemas de descuento de cantidad de la orden, problemas de periodos probabilísticos simples y problemas de tamaño dinámico de lotes; y evalúa y simula 4 sistemas de control de inventarios: (s, Q), (s, S), (R, S) y (R, s, S).

Programación de jornadas de trabajo (*Job Scheduling*)

Job scheduling (JOB): Este programa resuelve los problemas de taller de tareas y programación del flujo de trabajo usando generación heurística y aleatoria.

Programación lineal y entera (*Linear and integer programming*)

Linear Programming (LP) e Integer Linear Programming (ILP): Este programa resuelve los problemas de LP usando el metodo simplex o el método gráfico y los problemas de ILP usando el procedimiento branch-and-bound.

Procesos de Markov

Markov process (MKP): Este programa resuelve y analiza el proceso de Markov.

Planeación de Requerimiento de Materiales

Material requirements planning (MRP): El programa efectúa la planeación de requerimiento de materiales y determina que, cuanto y cuanto cuestan los materiales y componentes que son requeridos para satisfacer un plan de producción de productos finales para un horizonte de planeación**.**

Modelación de redes (*Network Modeling*)

Network Modeling (NET): Este modulo resuelve los problemas de red incluyendo flujo de red (transbordo), transporte, asignación, caminos cortos, máximo flujo, cruces mínimos y problemas de viajes de vendedores.

Programación no lineal (*Nonlinear Programming*)

Nonlinear Programming (NLP): Este programa resuelve los problemas no lineales no forzados usando el método de búsqueda y los problemas no lineales forzados usando el método de la función de castigo.

PERT y CPM (*PERT_CPM*)

PERT/CPM: Este módulo resuelve los problemas de planeación de proyectos usando el método de ruta crítica y la técnica de evaluación y revisión. Así mismo realiza análisis de choque, análisis de costos, análisis de probabilidad y simulación.

Programación cuadrática (*Quadratic Programming*)

Quadratic Programming (QP) e Integer Quadratic Programming (IQP): Este programa resuelve los problemas de QP usando el método simplex modificado o el método gráfico y los problemas de IQP usando el procedimiento branch-and-bound.

Cartas de control de calidad (*Quality Control Chart*)

Quality control charts (QCC): Construye gráficos de control de calidad para variables y datos de atributos y así mismo realiza análisis de gráficas relacionadas.

Sistemas de colas (*Queuing Analysis*)

Queuing analysis (QA): Este programa resuelve el rendimiento de sistemas de colas de etapa simple usando la formula de cercanía, aproximación o simulación.

Simulación de sistemas de cola (*Queuing Analysis Simulation*)

Queuing system simulation (QSS): Este programa modela y simula sistemas de colas simples y multietapas con componentes, incluyendo poblaciones de clientes arribando, servidores, colas y/o colectores de basuras.

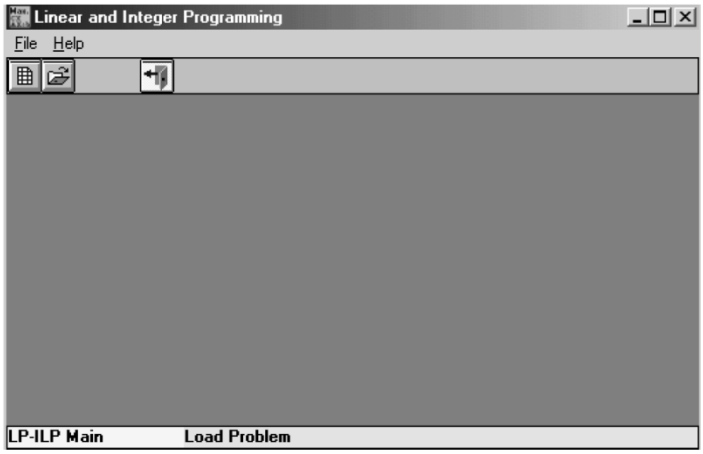
WinQSB utiliza los mecanismos típicos de la interface de Windows, es decir, ventanas, menús desplegables, barras de herramientas, etc. Por lo tanto el manejo del programa es similar a cualquier otro que utilice el entorno Windows.

Una vez seleccionado el módulo con el cual se desee trabajar, aparecerá una ventana cuyas características iníciales serán similares para todos los módulos del **WinQSB**. Al acceder a cualquiera de los módulos se abre una ventana en la que debemos elegir entre crear un nuevo problema (**File > New Problem**) o leer uno ya creado (**File > Load Problem**). Las extensiones de los ficheros con los modelos las pone el programa por defecto, por lo tanto solamente debemos preocuparnos del nombre, que no deberá tener más de 8 caracteres.

Todos los módulos del programa tienen en común los siguientes menús desplegables:

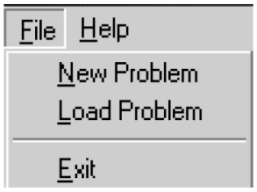
- **File:** incluye las opciones típicas de este tipo de menús en Windows, es decir, permite crear y salvar ficheros con nuevos problemas, leer otros ya existentes o imprimirlos.
- **Edit:** incluye las utilidades típicas para editar problemas, copiar, pegar, cortar o deshacer cambios. También permite cambiar los nombres de los problemas, las variables, y las restricciones. Facilita la eliminación o adición de variables y/o restricciones, y permite cambiar el sentido de la optimización.
- **Format:** incluye las opciones necesarias para cambiar la apariencia de las ventanas, colores, fuentes, alineación, anchura de celdas, etc.
- **Solve and Analyze:** esta opción incluye al menos dos comandos, uno para resolver el problema y otro para resolverlo siguiendo los pasos del algoritmo.

- **Results:** incluye las opciones para ver las soluciones del problema y realizar si procede distintos análisis de la misma.
- **Utilities:** este menú permite acceder a una calculadora, a un reloj y a un editor de gráficas sencillas.
- **Window:** permite navegar por las distintas ventanas que van apareciendo al operar con el programa.
- **WinQSB:** incluye las opciones necesarias para acceder a otro módulo del programa.
- **Help:** permite acceder a la ayuda on-line sobre la utilización del programa o las técnicas utilizadas para resolver los distintos modelos. Proporciona información sobre cada una de las ventanas en la que nos encontremos.



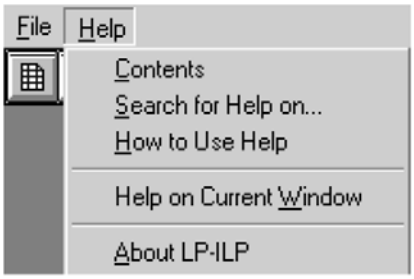
La parte superior de la ventana llamada ***TITULO*** indica el nombre del módulo seleccionado, en este caso se optó por mostrar el módulo de ***Programación Lineal y Entera*** (***Linear and integer programming***).

Debajo encontramos los menú ***Archivo*** (***File***) y ***Ayuda*** (***Help***). El menú archivo comprende las siguientes opciones:



- ***Nuevo problema*** (***New Problem***): Permite introducir un nuevo problema.
- ***Abrir Problema*** (***Load Problem***): Abre un problema que se ha guardado con anterioridad.
- ***Salir*** (***Exit***): Sale del programa.

El menú ***Ayuda*** (***Help***) lo conforman:



- ***Contenido*** (***Contents***): Contenido completo de la ayuda sobre el módulo seleccionado.
- ***Buscar ayuda en...*** (***Search for Help on...***): Búsqueda de ayuda mediante palabras claves.
- ***Cómo usar la ayuda*** (***How to Use Help***): Indicaciones (puede ser en español) de como se utiliza la ayuda para sacarle el máximo provecho.
- ***Ayuda sobre la ventana actual*** (***Help on Current Windows***): Interesante opción que muestra la ayuda sólo sobre los elementos que aparecen actualmente en la ventana.
- ***Acerca de...*** (***About LP-ILP***): Muestra datos sobre la creación del programa e información sobre la licencia.

El programa también cuenta con una barra de herramientas que ayuda de forma significativa la selección de las opciones más usadas.

El primer botón permite la creación de un nuevo problema, el segundo abre un problema existente, mientras que el tercero, permite salir del programa.
En el centro de la venta se encuentra un espacio vacío el cual llamaremos ***ZONA DE TRABAJO***, donde se procederá a alimentar con información al programa.

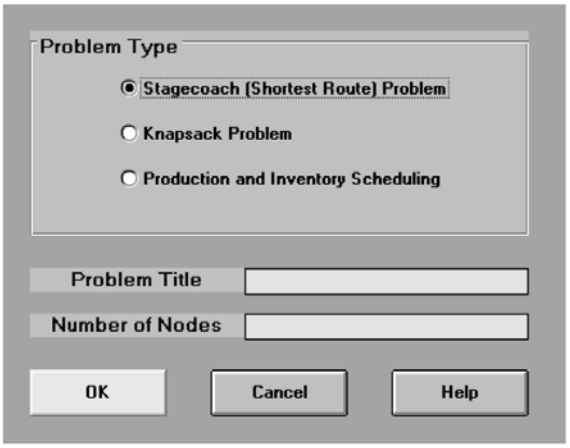
B. PROGRAMACIÓN DINÁMICA

La programación dinámica es un enfoque general para la solución de problemas en los que es necesario tomar decisiones en etapas sucesivas. Las decisiones tomadas en una etapa condicionan la evolución futura del sistema, afectando a las situaciones en las que el sistema se encontrará en el futuro (denominadas estados), y a las decisiones que se plantearán en el futuro.
Conviene resaltar que a diferencia de la programación lineal, el modelado de problemas de programación dinámica no sigue una forma estándar. Así, para cada problema será necesario especificar cada uno de los componentes que caracterizan un problema de programación dinámica.
El procedimiento general de resolución de estas situaciones se divide en el análisis recursivo de cada una de las etapas del problema, en orden inverso, es decir comenzando por la última y pasando en cada iteración a la etapa antecesora. El análisis de la primera etapa finaliza con la obtención del óptimo del problema.

B.1 Modelos de Programación Dinámica

Existen tres modelos diferentes manejados por **WinQSB**.

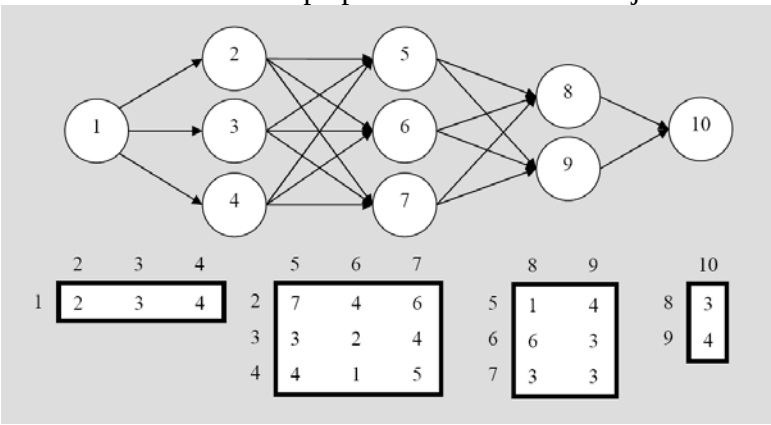
- ***Problema de la diligencia*** (***Stagecoach Problem***)
- ***Problema de la diligencia*** (***Stagecoach Problem***)
- ***Problema de la mochila*** (***Snapsack Problem***)
- ***Programación de producción e inventarios*** (***Production and InventoryScheduling***)



B.2 El Problema de la diligencia

Ejemplo B.1:

Considérese el gráfico que contempla las rutas posibles para ir desde la ciudad 1 hasta la ciudad 10. Cada nodo representa una ciudad y los arcos la infraestructura vial disponible. La tabla recoge el costo asociado al desplazamiento entre cada par de nodos para cada una de las etapas. Supondremos que todos los desplazamientos tienen la misma duración, y que el viaje ha de realizarse en cuatro etapas. Cada una de ellas se corresponde con un único desplazamiento entre un par de nodos del grafo, así al finalizar la primera etapa estaremos en una de las ciudades 2, 3 ó 4. La segunda etapa finalizará en la ciudad 5, la número 6 ó la número7. La tercera jornada nos llevará a la ciudad 8 o a la número 9. La cuarta etapa permite finalizar el viaje en la ciudad 10.



B.3 Terminología y Notación Básica

Períodos o etapas: Sea N= {1, 2,....., n} un conjunto finito de elementos. Mediante el índice $N\ n \in$, representamos cada uno de ellos. N es el conjunto de períodos o etapas del proceso. En la ilustración anterior N= {1, 2, 3, 4}, las cuatro etapas del viaje, cada una de ellas es un período y se representa mediante un valor del índice *n*, así cuando *n* =1 nos estamos refiriendo a la primera etapa del proceso.

Espacio de estados: es una familia de conjuntos, uno para cada período *n*. *S* se denomina espacio de estados en el período *n*. Cada uno de sus elementos, que se representa mediante *S*n, es un estado, que describe una posible situación del proceso en ese período. En nuestro ejemplo, *S*1 = {1}, *S*2= {2, 3, 4}, *S*3= {5, 6, 7}, *S*4= {8, 9}.

La función recursiva: Dados unos nodos y unos arcos que conectan estos nodos, el problema de la diligencia intenta encontrar la ruta más corta que conecta un nodo de arranque con el nodo final (el destino).

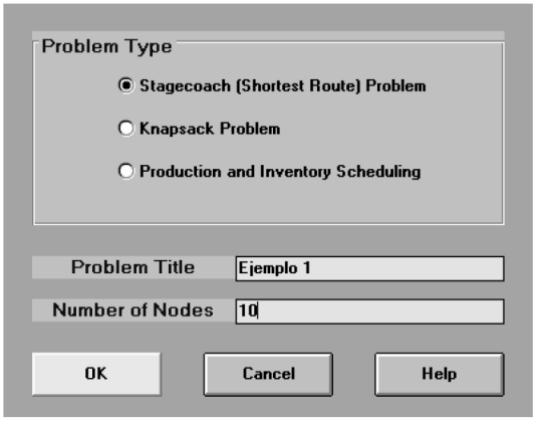
Sea s: el estado de inicio; j: estado destino
- **n**: la fase, normalmente representa el número de arcos hasta el destino.
- **C(s,j)**: costo o distancia de ir desde s hasta j.
- **f(n,s)**: la política de costo mínimo cuando se encuentra en el estado s de la etapa n.

La relación recursiva dinámica se expresa como

f(n,s) = mínimo [C(s,j) + f(n-1,j)] para todos los arcos (s, j) en la red

B.4 Ingresando el problema al WinQSB

El problema contiene 10 nodos claramente identificados:



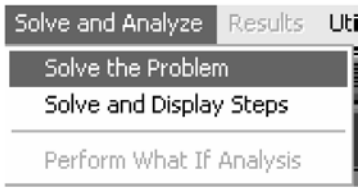
Al pulsar ***OK*** podremos ingresar el resto de información, el cual se basa en las relaciones existentes entre los nodos:

From \ To	Node1	Node2	Node3	Node4	Node5	Node6	Node7	Node8	Node9
Node1									
Node2									
Node3									
Node4									
Node5									
Node6									
Node7									
Node8									
Node9									
Node10									

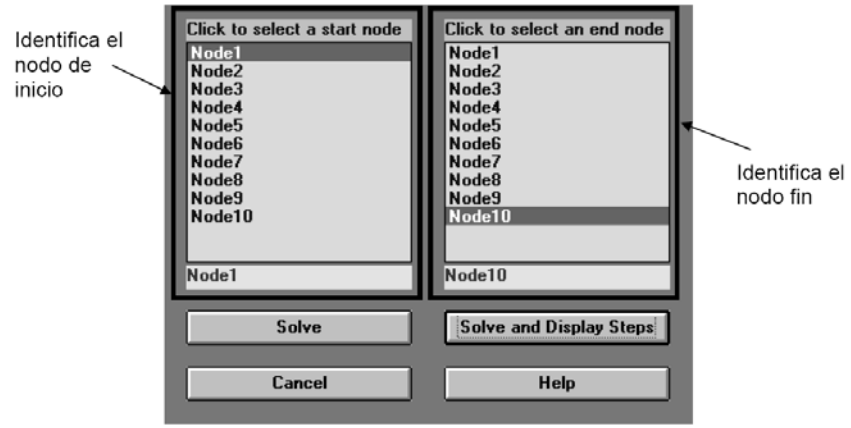
Los valores van de acuerdo a la red establecida en el problema:

From \ To	Node1	Node2	Node3	Node4	Node5	Node6	Node7	Node8	Node9	Node10
Node1		2	4	3						
Node2					7	4	6			
Node3					3	2	4			
Node4					5	6	7			
Node5								1	4	
Node6								6	3	
Node7								3	3	
Node8										3
Node9										4
Node10										

Para resolver el problema pulsamos la opción ***Resolver el problema*** (***Solve the Problem***) del menú ***Resolver y analizar*** (***Solve and Analyze***).



La ventana siguiente permite identificar los nodos de inicio y fin:
Al pulsar ***SOLVE*** generamos la solución al problema:
Si queremos una solución detallada debemos pulsar sobre ***Mostrar solución detallada*** (***Show Solution Detail***) en el menú ***Resultados*** (***Results***):

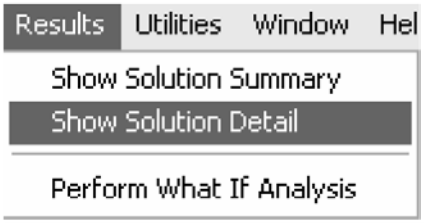


Al pulsar ***SOLVE*** generamos la solución al

11-30-2005 Stage	From Input State	To Output State	Distance	Cumulative Distance	Distance to Node10
1	Node1	Node3	4	4	11
2	Node3	Node5	3	7	7
3	Node5	Node8	1	8	4
4	Node8	Node10	3	11	3
	From Node1	To Node10	Min. Distance	= 11	CPU = 0

problema:

Si queremos una solución detallada debemos pulsar sobre ***Mostrar solución detallada*** (***Show Solution Detail***) en el menú ***Resultados*** (***Results***):



11-30-2005 17:04:56	Stage	From Input State	To Output State	Distance	Distance to Node10	Status
1	1	Node1	Node3	4	11	Optimal
2	2	Node2	Node5	7	11	
3	2	Node3	Node5	3	7	Optimal
4	2	Node4	Node5	5	9	
5	3	Node5	Node8	1	4	Optimal
6	3	Node6	Node9	3	7	
7	3	Node7	Node8	3	6	
8	4	Node8	Node10	3	3	Optimal
9	4	Node9	Node10	4	4	
	From Node1	To Node10	Minimum	Distance =	11	CPU = 0

B.5 Problema de la mochila o canasta de equipaje

La idea básica es que existen **N** tipos distintos de artículos que pueden cargarse en una mochila; cada artículo tiene asociados un peso y un valor. El problema consiste en determinar cuántas unidades de cada artículo se deben colocar en la mochila para maximizar el valor total. Nótese que este enfoque resulta útil para la planificación del transporte de artículos en algún medio, por ejemplo: carga de un buque, avión, camión etc. También es utilizable este modelo en planificación de producción, por ejemplo enrutamiento de la producción a través de varias máquinas.

Ejemplo B.2:

La carga de un avión se distribuye con el propósito de maximizar el ingreso total. Se consideran 5 elementos y sólo se necesita uno de cada uno. La compañía gana 5000 u.m. por elemento más una bonificación por elemento. El avión puede transportar 2000 libras.

Elemento	Peso, lb	Volumen, pies^3	Valor bonificación
1	1000	70	700
2	1100	100	800
3	700	100	1100
4	800	80	1000
5	500	50	700

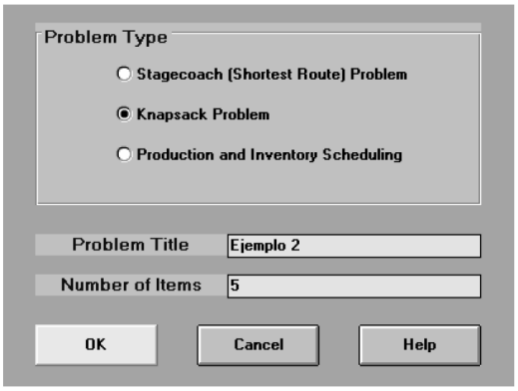
a) ¿Cuáles elementos deben transportarse?
b) Si se considera un volumen máximo de 200 pies cúbicos. ¿Cuáles elementos deben transportarse?

El problema se desarrolla bajo las dos consideraciones, primero teniendo en cuenta el peso y luego el volumen. Como puede apreciarse este es un problema que bien podría resolverse por programación lineal entera teniendo en cuenta la función objetivo y restricciones siguientes:

$$Max.Z = 5700x_1 + 5800x_2 + 6100x_3 + 6000x_4 + 5700x_5$$
$$S.a. \quad 1000x_1 + 1100x_2 + 700x_3 + 800x_4 + 500x_5 \leq 2000$$
$$x_j \leq 1, \; entero$$

Siendo x_j el elemento j a transportar.
Para el caso del volumen se reformaría la primera restricción cambiando los coeficientes por los volúmenes de los ítems.
Sea j: la variable que representa el artículo:
• **x(j)**: el número de unidades el número de unidades cargadas del artículo j
• **w(j)**: el espacio o el peso que demanda cada unidad del artículo j
• **R(j,x(j))**: la función del retorno del artículo j si se llevan x(j) unidades en la mochila, del artículo j
• **g(j,w)**: retorno del total acumulativo dado el espacio w disponible para el artículo j
La relación recursiva dinámica se expresa como:
g(j,w) = máximo {R(j,x(j)) + g[j-1,w-w(j)x(j)]} para todo posible x(j)

Ahora ingresemos los datos al **WinQSB**:



La entrada de datos queda como sigue.

Item (Stage)	Item Identification	Units Available	Unit Capacity Required	Return Function (X: Item ID) (e.g., 50X, 3X+100, 2.15X^2+5)
1	A	1	1000	5700A
2	B	1	1100	5800B
3	C	1	700	6100C
4	D	1	800	6000D
5	E	1	500	5700E
Knapsack	Capacity =	2000		

Al resolver el problema tenemos:

11-30-2005 Stage	Item Name	Decision Quantity (X)	Return Function	Total Item Return Value	Capacity Left
1	A	0	5700A	0	2000
2	B	0	5800B	0	2000
3	C	1	6100C	6100	1300
4	D	1	6000D	6000	500
5	E	1	5700E	5700	0
	Total	Return	Value =	17800	CPU = 0,22

La solución nos indica que se deben transportar los ítems 3, 4 y 5 con un retorno total de 17800 u.m. y utilización plena de la capacidad (en peso), disponible del avión. Teniendo en cuenta sólo el volumen, el nuevo modelo es:

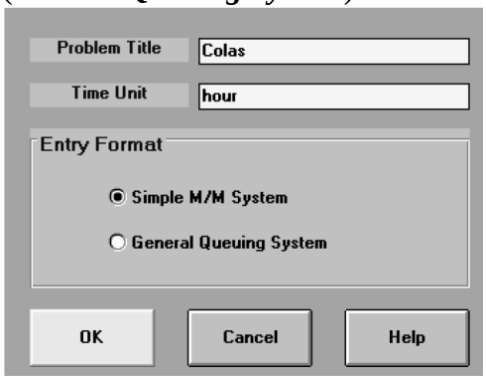
Item (Stage)	Item Identification	Units Available	Unit Capacity Required	Return Function (X: Item ID) (e.g., 50X, 3X+100, 2.15X^2+5)
1	A	1	70	5700A
2	B	1	100	5800B
3	C	1	100	6100C
4	D	1	80	6000D
5	E	1	50	5700E
Knapsack	Capacity =	120		

La solución es:

11-30-2005 Stage	Item Name	Decision Quantity (X)	Return Function	Total Item Return Value	Capacity Left
1	A	1	5700A	5700	50
2	B	0	5800B	0	50
3	C	0	6100C	0	50
4	D	0	6000D	0	50
5	E	1	5700E	5700	0
	Total	Return	Value =	11400	CPU = 0,02

C. TEORÍA DE COLAS

Un primer paso consiste, como en todos los modelos, en la especificación del problema mediante la cual se establecerá si el modelo a tratar es un M/M/S (***Simple M/M System***) o un modelo general (***General Queuing System***).



3.1 Los campos requeridos

Vamos a suponer por ahora un modelo M/M/S. Lo que sigue es el ingreso de los datos de acuerdo con las especificaciones de la ventana.

La ventana anterior consta de:

Data Description	ENTRY
Number of servers	
Service rate (per server per hour)	
Customer arrival rate (per hour)	
Queue capacity (maximum waiting space)	M
Customer population	M
Busy server cost per hour	
Idle server cost per hour	
Customer waiting cost per hour	
Customer being served cost per hour	
Cost of customer being balked	
Unit queue capacity cost	

- ***Numero de servidores*** (***Number of Servers***)
- ***Tasa de servicio*** (***Service Rate***)
- ***Tasa de llegada de clientes*** (***Customer Arrival Rate***)
- ***Capacidad de la cola*** (***Queue Capacity***)
- ***Tamaño de la población de clientes*** (***Customer Population***)
- ***Costo del servidor ocupado*** (***Busy Server Cost per Hour***)
- ***Costo del servidor desocupado*** (***Idle Server Cost per Hour***)
- ***Costo de espera de los clientes*** (***Customer Waiting Cost per Hour***)
- ***Costo de los clientes siendo servidos*** (***Customer Being Served Cost per Hour***)
- ***Costo de los clientes siendo despachados*** (***Cost of Customer Being Balked***)
- ***Costo de la unidad de capacidad de la cola*** (***Unit Queue Capacity Cost***)

Un ejemplo del modelo es el siguiente (recuerde que las letras **M** indican un valor infinito o muy grande):

Data Description	ENTRY
Number of servers	2
Service rate (per server per hour)	15
Customer arrival rate (per hour)	20
Queue capacity (maximum waiting space)	M
Customer population	M
Busy server cost per hour	150
Idle server cost per hour	150
Customer waiting cost per hour	200
Customer being served cost per hour	200
Cost of customer being balked	150
Unit queue capacity cost	Pesos

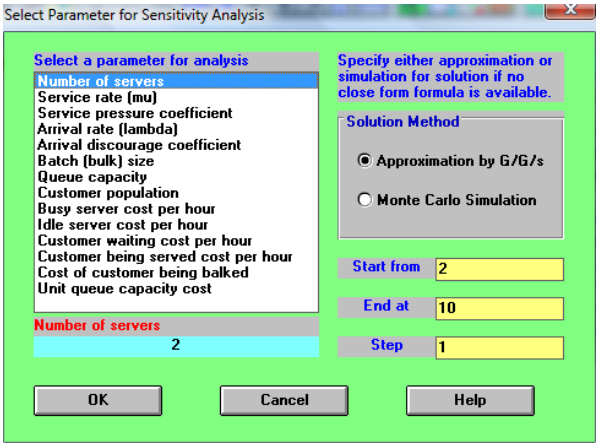
Una de las posibilidades de solución es calcular las tradicionales medidas de desempeño (medidas de efectividad), que nos proporciona el tablero siguiente:

06-09-2009	Performance Measure	Result
1	System: M/M/2	From Formula
2	Customer arrival rate (lambda) per hour =	20.0000
3	Service rate per server (mu) per hour =	15.0000
4	Overall system effective arrival rate per hour =	20.0000
5	Overall system effective service rate per hour =	20.0000
6	Overall system utilization =	66.6667 %
7	Average number of customers in the system (L) =	2.4000
8	Average number of customers in the queue (Lq) =	1.0667
9	Average number of customers in the queue for a busy system (Lb) =	2.0000
10	Average time customer spends in the system (W) =	0.1200 hours
11	Average time customer spends in the queue (Wq) =	0.0533 hours
12	Average time customer spends in the queue for a busy system (Wb) =	0.1000 hours
13	The probability that all servers are idle (Po) =	20.0000 %
14	The probability an arriving customer waits (Pw) or system is busy (Pb) =	53.3333 %
15	Average number of customers being balked per hour =	0
16	Total cost of busy server per hour =	$200.0000
17	Total cost of idle server per hour =	$100.0000
18	Total cost of customer waiting per hour =	$213.3334
19	Total cost of customer being served per hour =	$266.6667
20	Total cost of customer being balked per hour =	$0
21	Total queue space cost per hour =	$0
22	Total system cost per hour =	$780.0000

El resumen de probabilidades de encontrar n clientes en el sistema es:

06-09-2009 12:45:27 n	Estimated Probability of n Customers in the System	Cumulative Probability
0	0.2000	0.2000
1	0.2667	0.4667
2	0.1778	0.6444
3	0.1185	0.7630
4	0.0790	0.8420
5	0.0527	0.8947
6	0.0351	0.9298
7	0.0234	0.9532
8	0.0156	0.9688
9	0.0104	0.9792
10	0.0069	0.9861
11	0.0046	0.9908
12	0.0031	0.9938
13	0.0021	0.9959
14	0.0014	0.9973
15	0.0009	0.9982
16	0.0006	0.9988
17	0.0004	0.9992
18	0.0003	0.9995
19	0.0002	0.9996
20	0.0001	0.9998
21	0.0001	0.9998
22	0.0001	0.9999
23	0.0000	0.9999

Análisis de sensibilidad a cambios en número de servidores iniciando en 2 y terminando en 10.



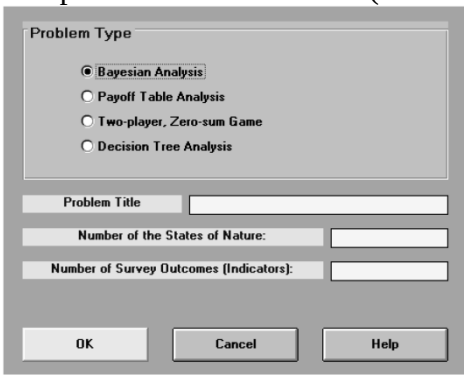
Un análisis parecido puede hacerse tomando como base la capacidad del sistema, que puede ir desde una capacidad específica de x clientes (capacidad limitada) hasta infinita.

Sensitivity Analysis of Number of servers for practica

06-09-2009 Value	Effective Arrival Rate	System Utilization	L	Lq	Lb	W	Wq	Wb	P0	Pw	Average Balked	Busy Server Cost	Idle Server Cost	Waiting Customer Cost	Served Customer Cost	Balked Customer Cost	Queue Capacity Cost	TOTAL COST
2	20.0000	0.6667	2.4000	1.0667	2.0000	0.1200	0.0533	0.1000	0.2000	0.5333	0	200.0000	100.0000	213.3334	266.6667	0	0	780.0000
3	20.0000	0.4444	1.4780	0.1446	0.8000	0.0739	0.0072	0.0400	0.2542	0.1808	0	200.0000	250.0000	28.9266	266.6667	0	0	745.5933
4	20.0000	0.3333	1.3592	0.0259	0.5000	0.0680	0.0013	0.0250	0.2621	0.0518	0	200.0000	400.0000	5.1780	266.6667	0	0	871.8447
5	20.0000	0.2667	1.3379	0.0046	0.3636	0.0669	0.0002	0.0182	0.2634	0.0126	0	200.0000	550.0000	0.9172	266.6667	0	0	1017.5840
6	20.0000	0.2222	1.3341	0.0008	0.2857	0.0667	0.0000	0.0143	0.2636	0.0026	0	200.0000	700.0000	0.1511	266.6667	0	0	1166.8180
7	20.0000	0.1905	1.3334	0.0001	0.2353	0.0667	0.0000	0.0118	0.2636	0.0005	0	200.0000	850.0000	0.0228	266.6667	0	0	1316.6900
8	20.0000	0.1667	1.3333	0.0000	0.2000	0.0667	0.0000	0.0100	0.2636	0.0001	0	200.0000	1000.0000	0.0031	266.6667	0	0	1466.6700
9	20.0000	0.1481	1.3333	0.0000	0.1739	0.0667	0.0000	0.0087	0.2636	0.0000	0	200.0000	1150.0000	0.0004	266.6667	0	0	1616.6670
10	20.0000	0.1333	1.3333	0.0000	0.1538	0.0667	0.0000	0.0077	0.2636	0.0000	0	200.0000	1300.0000	0.0000	266.6667	0	0	1766.6670

D. TEORIA DE DECISIONES

La opción ***Nuevo Problema*** (***New Problem***) muestra una ventana con los siguientes campos:



A continuación se describirán los diferentes tipos de problemas sobre análisis de decisiones disponibles en **WINQSB** a través de la ventana ***Especificaciones del problema*** (***Problem Specification***):

- ***Análisis bayesiano*** (***Bayesian Analysis***)
- ***Análisis de tablas de pago*** (***Payoff Table Analysis***)
- ***Juegos de suma cero para dos jugadores*** (***Two-Player, Zeros-Sum Game***)
- ***Análisis de árboles de decisión*** (***Decision Tree Analysis***)

A continuación explicaremos con un ejemplo algunas de estas opciones:

D.1 Análisis Bayesiano

Mediante un ejemplo demostraremos como se introducen los datos para la creación de una aplicación de análisis bayesiano.

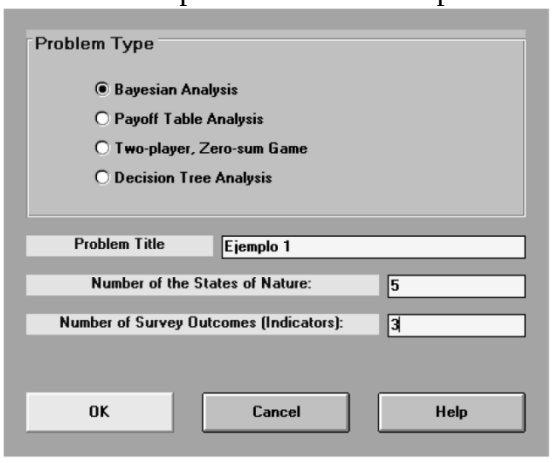
Ejemplo 8-1:

Se tienen cinco urnas con 10 canicas cada una, de colores azul, negra y rojo, según se muestra en la tabla:

Canicas	Urna 1	Urna 2	Urna 3	Urna 4	Urna 5
Azul	1	6	8	1	0
Negra	6	2	1	2	6
Rojo	3	2	1	7	4

Si se elige una urna en forma aleatoria y se extrae una canica y esta resulta ser roja, cuál es la probabilidad de que provenga de la urna 3.

En la ventana ***Especificaciones del problema*** (***Problem Specification***) procedemos a ingresar los datos básicos para la solución del problema:

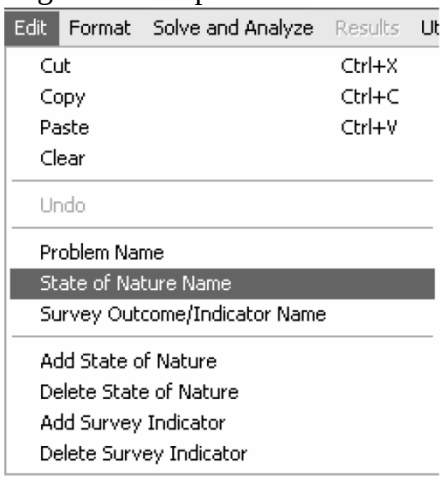


En el apartado ***Número de estados naturales*** (***Number of the States of Nature***) colocaremos la cantidad de urnas existentes, mientras que en el campo ***Númerode resultados*** (***Number of Survey Outcomes***) escribiremos los tipos de canicas (tres en total: azul, negra y roja).

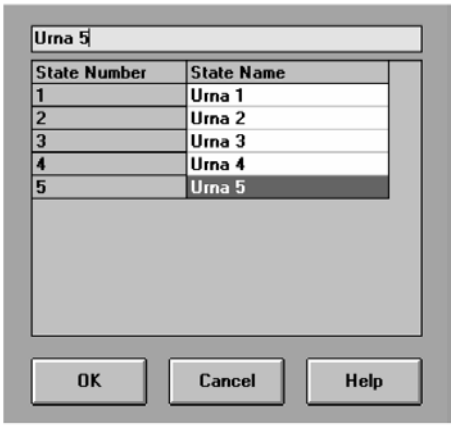
Al pulsar ***OK*** aparecerá una tabla en la cual podremos ingresar las probabilidades individuales, tanto para las urnas como las canicas que tienen dentro.

Outcome \ State	State1	State2	State3	State4	State5
Prior Probability					
Indicator1					
Indicator2					
Indicator3					

Para mejorar el aspecto de la tabla y evitar posibles equivocaciones en la interpretación de los datos, cambiaremos los campos de la tabla por los trabajados en el ejercicio. Empezaremos modificando los ***States*** por los nombre de las urnas correspondientes, para lo cual, en el menú ***Editar*** (***Edit***) elegiremos la opción ***Nombres de los estados naturales*** (***State of Nature Name***).



La ventana con los nombres modificados debe quedar así:



Para cambiar los ***Indicators*** por los correspondientes colores de las canicas haremos el mismo procedimiento solo que esta vez, seleccionaremos la opción ***Nombre del indicador*** (***Survey Outcomes/Indicator Name***)

Al pulsar ***OK*** regresaremos a la ventana inicial, la cual debería quedar como la siguiente:

Outcome \ State	Urna 1	Urna 2	Urna 3	Urna 4	Urna 5
Prior Probability					
Azul					
Negra					
Roja					

Para poder resolver el problema deberemos pasar primero los datos del ejercicio a las probabilidades:
• De elegir una urna de forma aleatoria (probabilidad anterior)
• De seleccionar una canica dentro de la urna

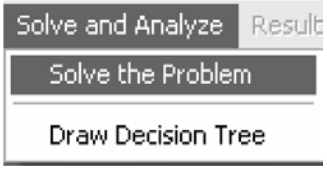
La tabla resumen quedaría:

Canicas	Urna 1	Urna 2	Urna 3	Urna 4	Urna 5
Probabilidad Anterior	0,2	0,2	0,2	0,2	0,2
Azul	0,1	0,6	0,8	0,1	0,0
Negra	0,6	0,2	0,1	0,2	0,6
Roja	0,3	0,2	0,1	0,7	0,4
Total probabilidad canicas	1,0	1,0	1,0	1,0	1,0

Ingresemos ahora los datos a la tabla del **WinQSB**:

Outcome \ State	Urna 1	Urna 2	Urna 3	Urna 4	Urna 5
Prior Probability	0.2	0.2	0.2	0.2	0.2
Azul	0.1	0.6	0.8	0.1	0.0
Negra	0.6	0.2	0.1	0.2	0.6
Roja	0.3	0.2	0.1	0.7	0.4

Para resolver el problema simplemente pulsamos en ***Resolver el problema*** (***Solve the Problem***) en el menú ***Resolver y analizar*** (***Solve and Analyze***).



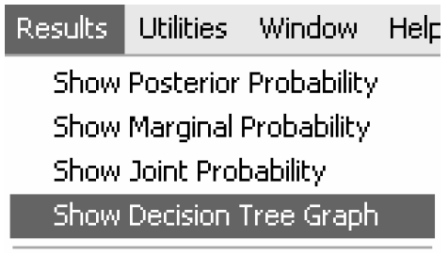
La tabla generada muestra los resultados de las probabilidades condicionales.

Indicator\State	Urna 1	Urna 2	Urna 3	Urna 4	Urna 5
Azul	0,0625	0,375	0,5	0,0625	0
Negra	0,3529	0,1176	0,0588	0,1176	0,3529
Roja	0,1765	0,1176	0,0588	0,4118	0,2353

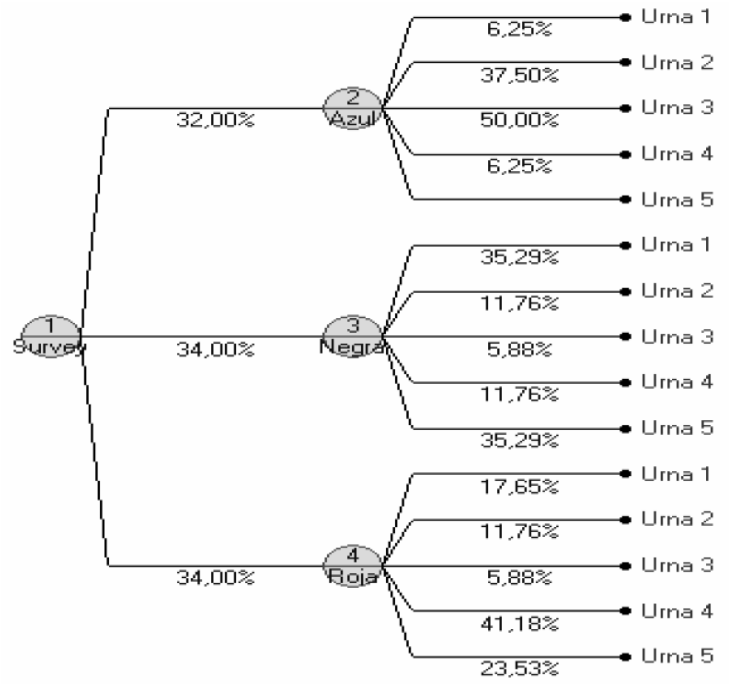
En este caso la probabilidad de que al haber seleccionado la urna 3 se saque una balota roja es de 5,88%.

Indicator\State	Urna 1	Urna 2	Urna 3	Urna 4	Urna 5
Azul	0,0625	0,375	0,5	0,0625	0
Negra	0,3529	0,1176	0,0588	0,1176	0,3529
Roja	0,1765	0,1176	0,0588	0,4118	0,2353

Para activar el modo gráfico pulsamos sobre ***Mostrar gráfico del árbol de decisión*** (***Show Decision Tree Graph***).



Gráficamente tenemos:



E. PROCESO DE MARKOV

La opción ***Nuevo Problema*** (***New Problem***) genera una plantilla llamada ***Especificaciones del problema PMK*** (***MKP Problem Specification***) en la cual, se introducirán las características de nuestro problema:

Para comenzar a armar un problema de este tipo es necesario ingresar los campos:
• ***Titulo del problema*** (***Problem Title***)
• ***Número de estados*** (***Number of States***)

E.1 Un poco de teoría

Un sistema existe en estados diferentes (o condiciones). A través del tiempo, el sistema se moverá de un estado a otro estado. El proceso de Markov normalmente se usa para caracterizar estos movimientos o transiciones. Para describir y analizar un proceso de Markov, definimos las terminologías siguientes:

• ***Estado***: una condición particular del sistema, i = 1, 2,..., n.
• ***Probabilidad de estados s(i)***: la probabilidad de que el sistema se encuentre en el estado i
• ***Probabilidad de transición p(i,j)***: la probabilidad de que el sistema se mueva del estado i al estado j
• ***S(t)***: conjunto de todos s(i) en momento t, $\Sigma s(i) = 1$
• ***P***: matriz de transición p(i,j), dónde i=1,2,…,m yj = 1, 2,... ,n

Dado el sistema en el momento t con las probabilidades de estado S(t), entonces en el momento t+1 , el sistema se expresará por ***S(T+1) = S(T) P***
Y en el t+2 , el sistema se expresará por

$$S(T+2) = S(T)\ P\ P = S(T)\ P^2$$

Y en t+3, el sistema se expresará por

$$S(T+2) = S(T)\ P\ P\ P = S(T)\ P^3$$

Y así sucesivamente.

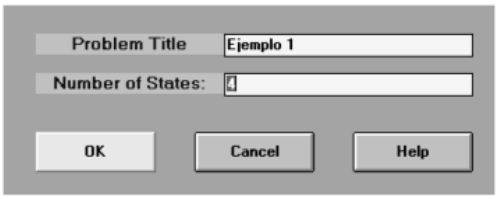
Si las probabilidades de estado no cambian de periodo a periodo, el sistema se encuentra en estado estable. No todo sistema tiene un estado estable. Si el sistema alcanza el estado estable, las probabilidades de estado estable, digamos S, tendrán las propiedades siguientes:
S = S P (1)

La ecuación (1) representa un conjunto de n ecuaciones simultáneas con n variables de probabilidad

de estado. Para obtener las probabilidades de estado estable, reemplace cualquiera de las ecuaciones en (1) con $\Sigma s(i) = 1$ y resuelva las n nuevas ecuaciones simultáneas.

E.2 Analizando un ejemplo

Ingresemos un sistema representado por 4 estados:



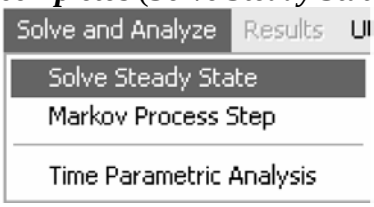
La plantilla vacía representa una matriz con las relaciones entre los estados (***State***), sus probabilidades iniciales (***Initial Prob.)*** y el costo de cada uno de ellos (***State Cost***).

From \ To	State1	State2	State3	State4
State1				
State2				
State3				
State4				
Initial Prob.				
State Cost				

Veamos un ejemplo:

From \ To	State1	State2	State3	State4
State1	0.2	0.3	0.1	0.4
State2	0.25	0.35	0.4	0
State3	0.1	0.2	0.2	0.5
State4	0.5	0.3	0.1	0.1
Initial Prob.	0.2	0.1	0.35	0.35
State Cost	2000	1500	1000	900

En el menú ***Resolver y analizar*** (***Solve and Analyze***) tenemos las opciones de ***Resolver los estados completos*** (***Solve Steady State***) o mostrar el ***Proceso de Markov por pasos*** (***Markov Process Step***).

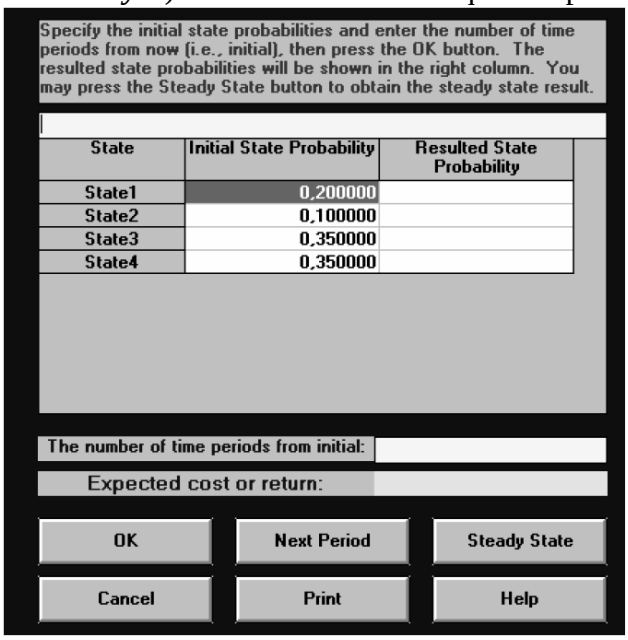


La primera opción da como resultado la siguiente tabla:

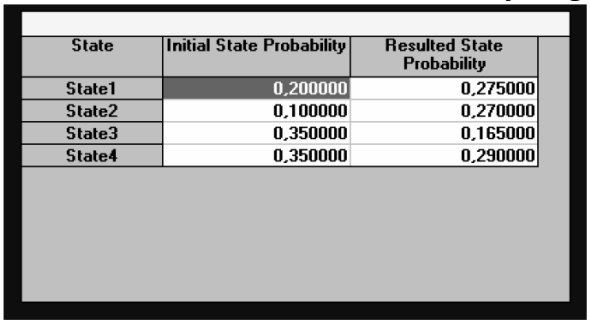
12-02-2005	State Name	State Probability	Recurrence Time
1	State1	0,2638	3,7908
2	State2	0,2938	3,4038
3	State3	0,2090	4,7838
4	State4	0,2334	4,2849
	Expected	Cost/Return =	1387,3530

E.3 Resolviendo el ejercicio paso a paso

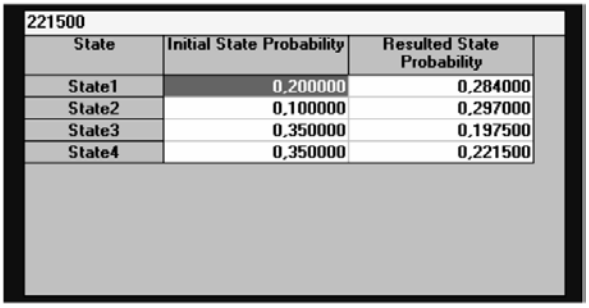
Regresando a la matriz inicial y tomando la segunda opción del menú ***Resolver y analizar*** (***Solve and Analyze***) tenemos una ventana que nos permite controlar las iteraciones del proceso:



Podemos observar el ***Número de periodos procesados*** (***The Number of Time Periods from Initial***). Pulsemos en el botón ***NEXT PERIOD*** y luego en el botón ***OK***:

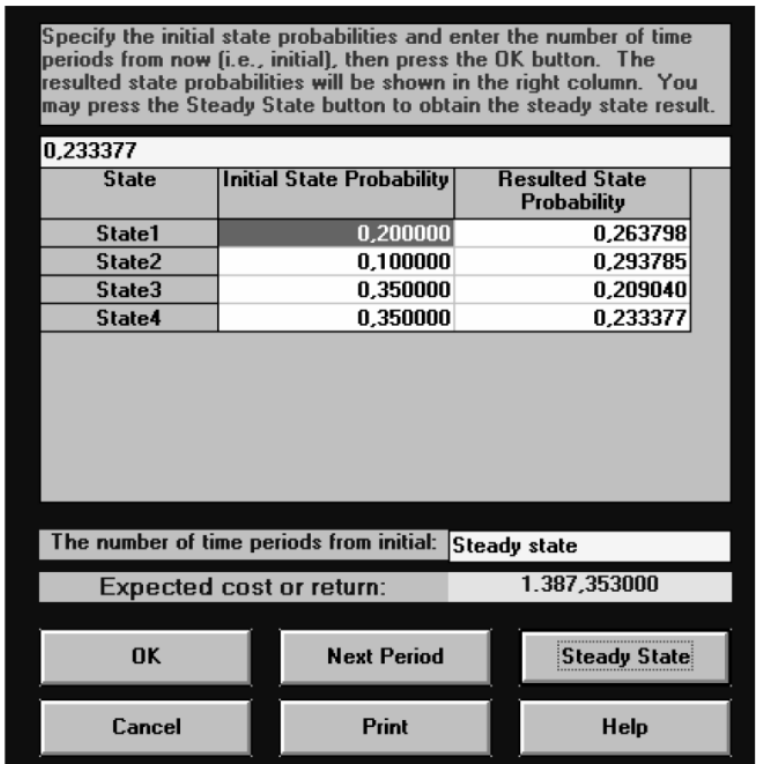


Para el periodo dos (recuerde pulsar en ***NEXT PERIOD*** seguido del botón ***OK***):

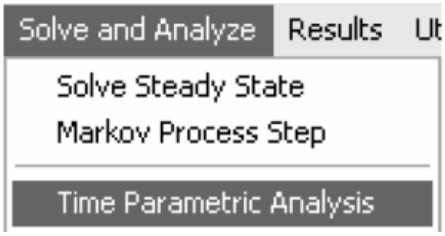


En la columna ***Probabilidad del estado resultante*** (***Resulted State Probability***) se muestran las

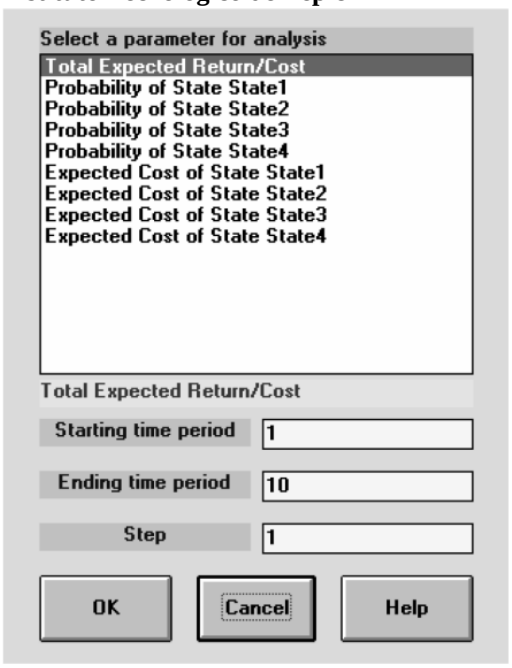
probabilidades para los periodos. Pulsando es el botón ***STEADY STATE*** alcanzamos la matriz estable:



Para ver un Análisis paramétrico en el tiempo de los costos y las probabilidades de los estados seleccionamos la opción



La nueva ventana contiene:



- ***Retorno/Costo total esperado*** (***Total Expected Return/Cost***)
- ***Probabilidad de cada estado*** (***Probability of State State#***)
- ***Costo esperado de cada estado*** (***Expected Cost of State State#***)

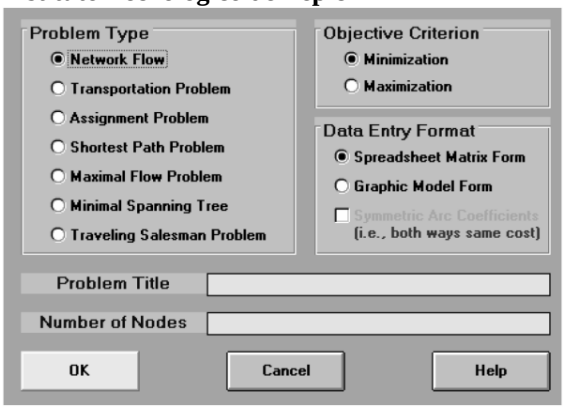
Pulsemos el botón ***OK*** para mostrar el ***Retorno/Costo total esperado*** (***Total Expected Return/Cost***) para 10 periodos (1 por periodo – Step = 1).

12-03-2005	Time Period	Total Expected Return/Cost
1	1	1381
2	2	1410,3500
3	3	1385,6500
4	4	1387,9060
5	5	1387,0120
6	6	1387,4280
7	7	1387,3330
8	8	1387,3610
9	9	1387,3510
10	10	1387,3540

Se puede observar como el costo comienza a estabilizarse para los últimos periodos (recuerde que el costo final es de 1987,3530).

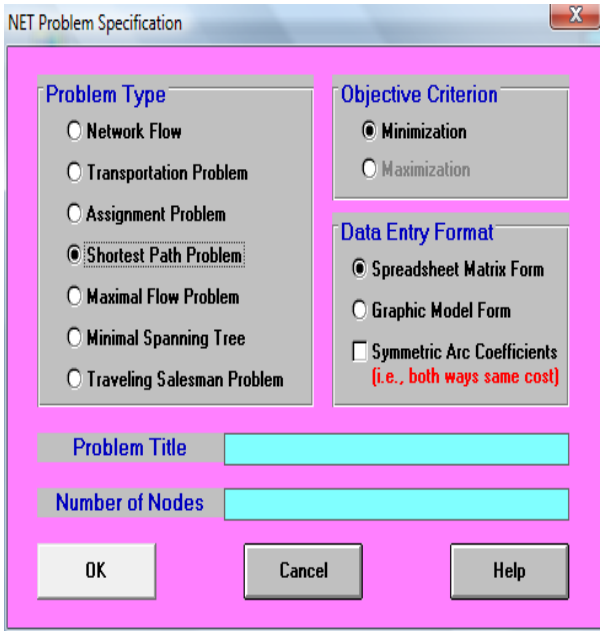
F. MODELO DE REDES

La opción ***Nuevo Problema*** (***New Problem***) generará la siguiente ventana:



Existen 7 modelos fundamentales para el tratamiento de los problemas que involucran redes con el fin de optimizar el uso de algún recurso, generalmente tratándose de la minimización de costos, tiempo o la maximización del flujo a través de una red. Estos modelos son:

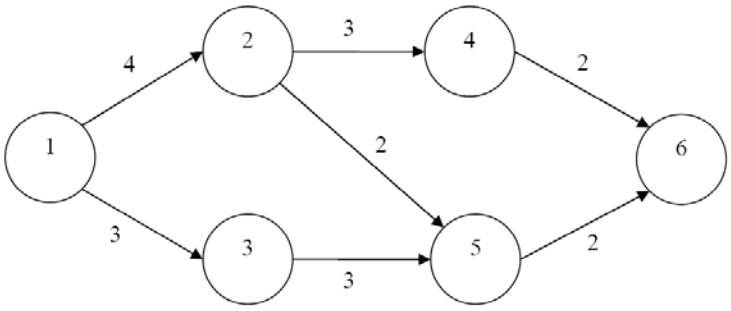
- ***Flujo en redes o modelo de trasbordo*** (**Network Flow**)
- ***Problema de transporte*** (***Transportation Problem***)
- **Problema de asignación** (***Assignment Problem***)
- ***Problema de la ruta más corta*** (***Shortest Path Problem***)
- ***Problema de flujo máximo*** (***Maximal Flow Problem***)
- ***Árbol de mínima expansión*** (***Minimal Spanning Tree***)
- ***Problema del agente viajero*** (***Traveling Salesman Problem***)



Por cuestiones del contenido de temas de de teoría de redes en este libro, se cubrirán únicamente los últimos tres temas.

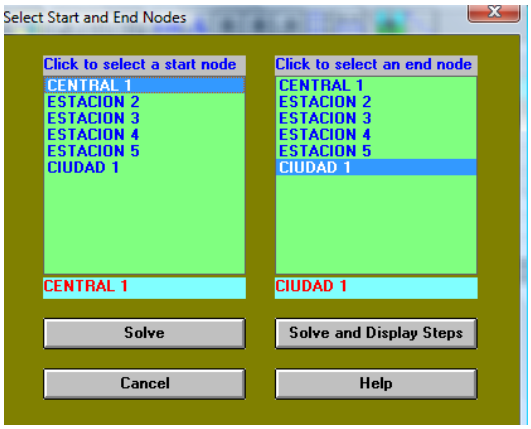
F.1 El Problema de la Ruta Más Corta

El problema de la ruta más corta incluye un juego de nodos conectados donde sólo un nodo es considerado como el origen y sólo un nodo es considerado como el nodo destino. El objetivo es determinar un camino de conexiones que minimizan la distancia total del origen al destino. El problema se resuelve por el "algoritmo de etiquetado".



La tabla siguiente muestra cómo se ingresan los datos para la red de ejemplo.
El nodo 1 representa la central y el nodo 6 la ciudad a donde debe llevarse el cableado procedente de la central, pasando por algunos de los otros nodos que conectan la central con la ciudad. Los números sobre los arcos representan distancias en millas. Se trata de llevar a cabo la interconexión con el menor consumo de cable.

From \ To	CENTRAL 1	ESTACION 2	ESTACION 3	ESTACION 4	ESTACION 5	CIUDAD 1
CENTRAL 1		4	3			
ESTACION 2				3	2	
ESTACION 3					3	
ESTACION 4						2
ESTACION 5						2
CIUDAD 1						



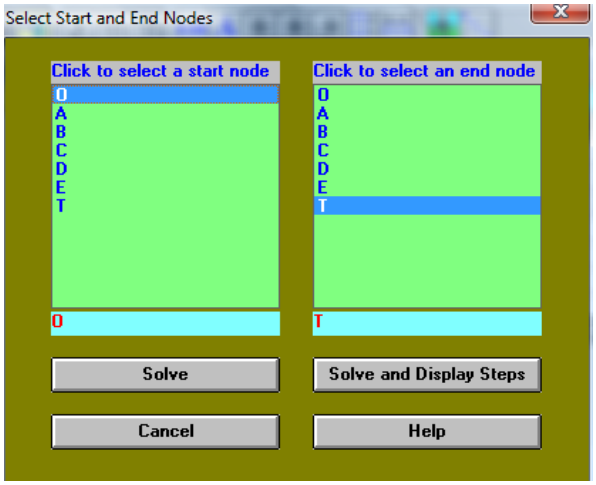
La solución final del problema sería:

06-09-2009	From	To	Distance/Cost	Cumulative Distance/Cost
1	CENTRAL 1	ESTACION 3	3	3
2	ESTACION 3	ESTACION 5	3	6
3	ESTACION 5	CIUDAD 1	2	8
	From CENTRAL 1	To CIUDAD 1	=	8
	From CENTRAL 1	To ESTACION 2	=	4
	From CENTRAL 1	To ESTACION 3	=	3
	From CENTRAL 1	To ESTACION 4	=	7
	From CENTRAL 1	To ESTACION 5	=	6

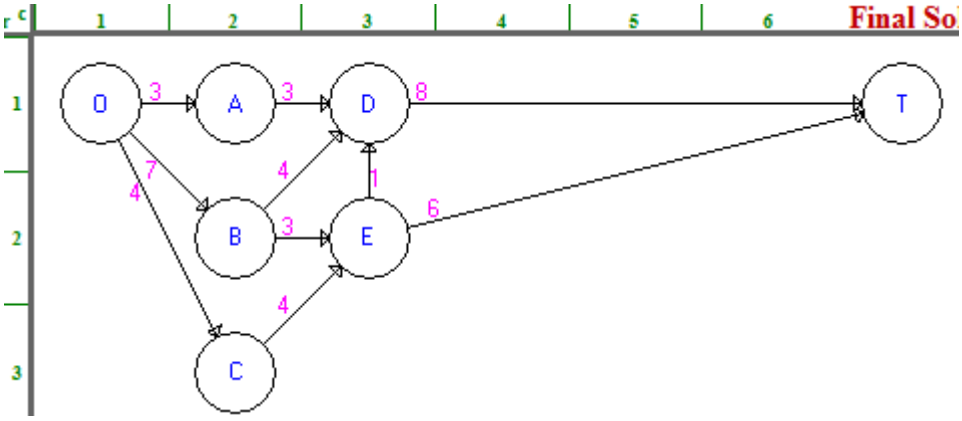
F.2 El Problema de Flujo Máximo

Muchos problemas pueden ser modelados mediante una red en la cual se considera que los arcos tienen la capacidad de limitar la cantidad de un producto que se puede enviar a través del arco. En estas situaciones, frecuentemente se desea transportar la máxima cantidad de flujo desde un punto de partida llamado fuente hacia un punto final denominado pozo. La tabla siguiente muestra un ejemplo de este modelo:

From \ To	O	A	B	C	D	E	T
O		5	7	4			
A			1		3		
B				2	4	5	
C						4	
D							9
E					1		6
T							



Gráficamente tenemos:



La solución del problema es:

06-09-2009	From	To	Net Flow		From	To	Net Flow
1	O	A	3	6	B	E	3
2	O	B	7	7	C	E	4
3	O	C	4	8	D	T	8
4	A	D	3	9	E	D	1
5	B	D	4	10	E	T	6
Total	Net Flow	From	O	To	T	=	14

Obsérvese que este modelo tiene aplicación en la planificación de transporte vehicular, transporte de líquidos mediante tuberías y otros problemas de similar estructura.

F.3 El Árbol de Expansión Mínima

Es un problema clásico de optimización combinatoria, formulado en 1926 por Boruvka quien lo planteó para resolver el problema de hallar la forma más económica de distribuir energía eléctrica en el sur de Moravia. La formulación de este problema ha sido útil para la realización de muchas investigaciones en varios campos como el transporte, electrónica, telecomunicaciones e investigación de operaciones.

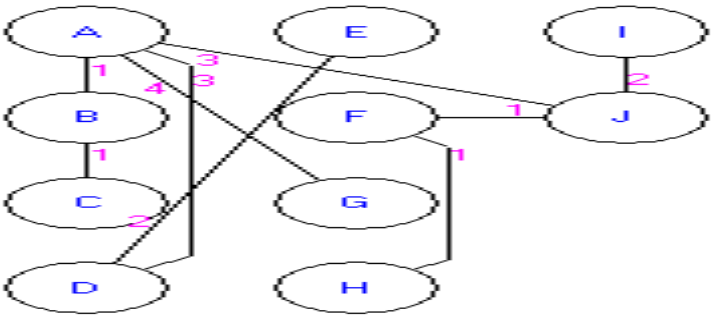
El modelo contempla un conjunto de arcos que conectan todos los nodos de la red sin crear un solo ciclo o vuelta. El problema consiste en determinar el árbol que minimiza la distancia de conexión total; se resuelve por el Algoritmo de Etiquetado. En cuanto a la introducción de datos y el proceso de solución es similar a los modelos anteriores de este módulo.

From \ To	A	B	C	D	E	F	G	H	I	J
A		1	2	3			4			3
B	1		1			3				4
C	2	1			3					
D	3				2		7			
E			3	2						
F		3						1		1
G	4			7					5	
H						1			3	4
I							5	3		2
J	3	4				1		4	2	

La solución para la plantilla anterior es:

06-09-2009	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	A	B	1	6	A	G	4
2	B	C	1	7	F	H	1
3	A	D	3	8	J	I	2
4	D	E	2	9	A	J	3
5	J	F	1				
	Total	Minimal	Connected	Distance	or Cost	=	18

El modelo de la red del ejemplo es:



7. REFERENCIAS BIBLIOGRÁFICAS

WinQSB, Yih-Long Chang, Versión 2.0. Ed. Wiley & Sons, Inc. 2003

ANÁLISIS CUANTITATIVO CON *WINQSB*, *VÍCTOR MANUEL QUESADA IBARGÜEN y JUAN CARLOS VERGARA SCHMALBACH*,http://rapidshare.com/files/.../WinQSB_2.0___Manual.rar

www.ojolink.com/**manual-winqsb/**

http://www.eumed.net/libros/2006c/216/1a.htm