**INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY**
**CAMPUS MONTERREY**
**DIVISIÓN DE INGENIERÍA Y ARQUITECTURA**
**PROGRAMA DE GRADUADOS EN INGENIERIA**


TECNOLÓGICO
DE MONTERREY®

# "Modelo de Optimización Binivel para la Determinación de Cuotas Carreteras"

## TESIS

**PRESENTADA COMO REQUISITO PARCIAL**
**PARA OBTENER EL GRADO ACADÉMICO DE:**
**MAESTRO EN CIENCIAS**
**CON ESPECIALIDAD EN SISTEMAS DE MANUFACTURA**

**POR:**
**GABRIEL PINTO SERRANO**

**MONTERREY, N. L.** **DICIEMBRE DEL 2010**

**INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY**

**CAMPUS MONTERREY**

**DIVISIÓN DE INGENIERÍA Y ARQUITECTURA**
**PROGRAMA DE GRADUADOS EN INGENIERÍA**

Los miembros del comité de tesis recomendamos que el presente proyecto de tesis presentado por el Ing. Gabriel Pinto Serrano sea aceptado como requisito parcial para obtener el grado académico de:

**Maestro en Ciencias**
**con Especialidad en Sistemas de Manufactura**

Comité de Tesis:

Dr. José Luis González Velarde
Asesor

Dr. José Fernando Camacho Vallejo
Co-asesor

Dr. Neale Ricardo Smith Cornejo
Sinodal

Aprobado:

Dr. Ciro Ángel Rodríguez González
Director de la Maestría en Sistemas de Manufactura

# Agradecimientos

**A mis Padres y Hermanos.**

Por todo su apoyo y enseñanza.

**Dr. José Luis González Velarde**

Por su guía, experiencia, pero sobretodo, paciencia durante todo este tiempo.

**Dr. José Fernando Camacho Vallejo.**

Por aceptar ser parte de este proyecto y por su gran apoyo durante su desarrollo.

**Dr. Neale Ricardo Smith Cornejo.**

Por aceptar ser parte de este proyecto y sus valiosos comentarios que ayudaron a enriquecer este trabajo.

**A la Cátedra de Investigación de Cadena de Suministro del Centro de Calidad y Manufactura del Instituto.**

Por la beca proporcionada para poder realizar mis estudios de posgrado.

**A mis amigos.**

Por hacerme sentir como en casa.

# Resumen

Para este trabajo se tomó el problema de determinar un conjunto óptimo en las cuotas para los arcos de una red de transporte multiproducto. El problema se formuló como un problema binivel, en donde el nivel superior consiste en una administración que establece las cuotas de la red, mientras que el nivel inferior está representado por un grupo de usuarios que viajan en los caminos más cortos con respecto a un costo de viaje generalizado.

El objetivo no es tan sólo incrementar las cuotas, sino también mantener un flujo óptimo en los arcos de la red para maximizar los beneficios. Se propone una metodología para resolver este problema utilizando el optimizador Cplex y la meta heurística “Scatter Search”.

# Tabla de Contenidos

# Índice de Figuras

# Índice de Tablas

# Capítulo 1.

## Introducción.-

Una de las cualidades fundamentales de todo ser humano es su capacidad para tomar decisiones, ya que esto significa el poder analizar las diferentes alternativas existentes y evaluarlas en términos de su comportamiento respecto de los objetivos que desean conseguir. Es una actividad tan rutinaria que prácticamente no le prestamos atención. Sin embargo, cuando el problema al que nos enfrentamos es muy complejo, con un sin fin de alternativas posibles, resulta difícil realizar este proceso de análisis y evaluación. Aún así se deben de tomar decisiones sobre la manera óptima de usar todos los recursos disponibles para lograr los objetivos planteados. Una manera de conseguir esto es mediante el uso de las herramientas que nos proporciona la Programación Matemática, la cual estudia el uso de modelos matemáticos para ayudar en la toma de decisiones.

En muchos de los procesos para la toma de decisiones existe una jerarquización, lo cual quiere decir que se toman decisiones en diferentes niveles. Estos son problemas de optimización que tienen un subconjunto de sus variables restringido a ser una solución óptima de otros problemas de optimización. Si sólo se consideran dos niveles en la jerarquización se trata de un problema de programación Binivel, en el cual el conjunto de las variables se divide en dos vectores ($x$, $y$). En donde el vector $x$ se toma como la solución óptima para un segundo problema parametrizado en $y$. Ésta es una de las áreas de investigación más intensa en la actualidad en optimización tanto por los retos que plantea su complejidad como por el número de sus aplicaciones.

Para este trabajo se generó un programa en lenguaje C que utiliza el optimizador Cplex 11.1 para resolver el nivel inferior como un problema de flujo en redes, mientras que para el nivel superior se utilizó la meta heurística Scatter Search. El principal objetivo de este trabajo es comparar los resultados obtenidos contra otros métodos propuestos para resolver este problema y así poder evaluar su desempeño y obtener conclusiones válidas para contribuir con el desarrollo de este tema.

## 1.1 **Justificación.-**

La importancia del transporte para el crecimiento económico y productivo de cualquier organización, e inclusive país, es indiscutible y un área que ha tomando una gran relevancia dentro de este tema es la de los caminos de cuota. Las cuotas ayudan a descongestionar caminos, y a su vez, ponen la carga monetaria en los mismos usuarios de esta infraestructura.

La literatura existente acerca de caminos de cuota es muy basta, sin embargo, la gran mayoría se enfoca primordialmente en reducir los congestionamientos y regular la demanda. Dejando los estudios sobre la asignación de cuotas para la maximización de beneficios un tanto rezagados.

## 1.2 **Definición del Problema.-**

En este trabajo de investigación se modeló una red de transporte multiproducto con la finalidad de establecer una asignación óptima en las tarifas de dichos caminos. El problema se formuló como un problema binivel, en donde el nivel superior consiste en una administración que establece las cuotas de la red, mientras que el nivel inferior está representado por un grupo de usuarios que viajan en los caminos más cortos con respecto a un costo de viaje generalizado. Esta tesis trata de responder a las siguientes preguntas:

1. ¿Cómo mantener rentable un sistema de caminos de cuota frente a la competencia de caminos libres?
2. ¿Cómo puede la programación binivel ser usada para encontrar un punto óptimo entre las cuotas establecidas y mantener un flujo de usuarios en la red?
3. ¿El modelo de optimización binivel diseñado con la meta heurística Scatter Search junto con el software de optimización Cplex será capaz de encontrar mejores resultados que los modelos propuestos actualmente?

## 1.3 **Estructura de la Tesis.-**

Este trabajo de tesis se ha dividido en 6 capítulos:

- Capítulo 1: Capítulo introductorio con la definición del problema y su justificación.
- Capítulo 2: Se realiza una revisión bibliográfica de los artículos más relevantes para este tema de investigación acerca de optimización de redes de transporte.
- Capítulo 3: Ofrece una visión general de las herramientas de optimización utilizadas para resolver este problema.
- Capítulo 4: En este capítulo se presenta la formulación utilizada para resolver el problema de asignación de cuotas carreteras.
- Capítulo 5: Describe el modelo de optimización binivel que fue utilizado y se muestran los estudios de caso con los que fue comprobado dicho modelo.
- Capítulo 6: Finalmente se muestran las conclusiones obtenidas y recomendaciones para posibles trabajos futuros.

# Capítulo 2.

# Revisión Bibliográfica.-

En este capitulo se presentará una revisión sobre los artículos de investigación más relevantes al presente tema. Primero se muestran los relacionados con los antecedentes y la historia. Después se presentan aquellos que están directamente relacionados, para terminar presentando algunas conclusiones y el enfoque hacia el cual ha estado siendo dirigido en general este tema de investigación.

## 2.1 Antecedentes e Historia.-

Los estudios sobre modelos de transporte no sólo no son nuevos en la literatura, sino que han sido estudiados ampliamente en áreas como ruteo de vehículos, el problema de la ruta más corta, el problema de trasbordo, el problema del flujo de costo mínimo, el problema del agente viajero, selección de modo de transporte, etc. Sin embargo los estudios sobre optimización de cuotas con restricciones de capacidad y equilibrio no son tan abundantes.

Los problemas de transporte se caracterizan por la facilidad que tienen de ser descritos mediante un grafo o una red (Maroto Álvarez, Alcázar Soria & Ruiz García, 2002).

Un grafo consiste en dos unidades fundamentales: vértices (o nodos) y aristas. Los grafos se representan mediante una serie de puntos (que representan a los vértices) unidos con líneas (representando a las aristas), en donde cada línea significa que existe algún tipo de conexión entre ese par de vértices. Puede representar, por ejemplo, una carretera que une dos ciudades. La notación para describir una red es ($N$, $A$) donde N es el numero de nodos y $A$ es el numero de arcos. Por ejemplo, la red de la figura 1 se describe de la siguiente forma (Taha, 2004):

$$N = \{1,2,3,4,5\} \quad A = \{(1,2), (1,3), (2,3), (2,5), (3,4), (3,5), (4,2), (4,5)\}$$

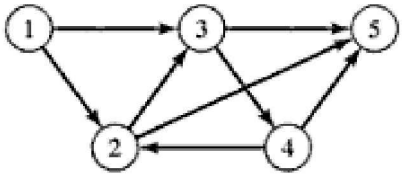



**Fig. 1 Ejemplo de una red. (Taha, 2004)**

Los grafos pueden ser dirigidos o no dirigidos, si los grafos son dirigidos las aristas se llaman arcos y se debe poner una por cada sentido del flujo.

Un grafo es conexo si cada par de vértices está conectado por un camino; es decir, si para cualquier par de vértices (*a*, *b*), existe al menos un camino posible desde *a* hacia *b*. Un grafo es fuertemente conexo si cada par de vértices está conectado por al menos dos caminos disjuntos; es decir, es conexo y no existe un vértice tal que al sacarlo el grafo resultante sea disconexo.

Como ya se mencionó, este tipo de problemas son fácilmente descritos mediante grafos. El modelo general de transporte puede ser representado en la red de la figura 2. En donde existen *m* fuentes y *n* destinos, cada fuente y cada destino son representados por un nodo. Los arcos representan las rutas que enlazan las fuentes y los destinos, así el arco (*i*, *j*) que une a la fuente *i* con el destino *j* lleva dos clases de información: el costo de transporte por unidad $c_{ij}$, y la cantidad transportada $x_{ij}$.

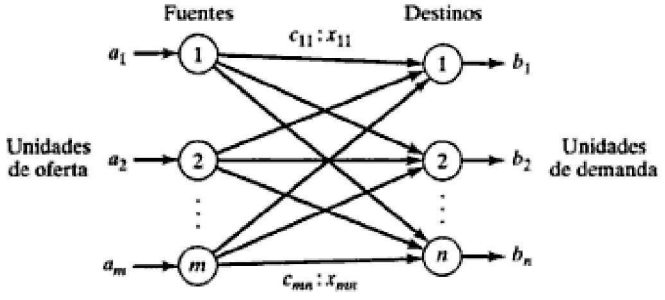



**Fig. 2 Representación del modelo de transporte. (Taha, 2004)**

Los problemas de optimización de redes pueden ser representados en términos generales a través de uno de estos cuatro modelos (Taha, 2004):

- Modelo de minimización de redes.
- Modelo de la ruta más corta.
- Modelo del flujo máximo.
- Modelo del flujo de costo mínimo.

Una parte del problema tratado en este trabajo en particular se modelará como un problema de transporte de flujo de costo mínimo. El cual, al igual que el resto de los modelos del problema de transporte, ha sido estudiado ampliamente. Gran parte de esto se debe a que a medida que la complejidad y el número de restricciones en el problema aumentan, se hace más difícil su resolución de forma exacta, por lo que resulta necesario recurrir a técnicas heurísticas de resolución (Gottinger & Weimann, 1990). Los autores (Garrido & Onaindía, 1999) proponen un algoritmo para resolver este problema empleando técnicas de búsqueda por profundización iterativa, mientras que (Edmonds, 1972) propone una técnica de escalamiento. Sin embargo los métodos más usados para resolver este tipo de problemas son (Zadeh, 1973):

- El método del camino (Path), empieza con un flujo de cero y lo va aumentando conforma va avanzando seleccionando la ruta más barata del origen al destino.
- El método del camino-m (M-Path), es un refinamiento del método anterior en dónde se usan números en los nodos para poder aplicar el algoritmo de Dijkstra.
- El método del ciclo (Cycle), empieza asignando flujos de una manera factible para después irse acercando al optimo al aumentar alrededor de un ciclo más negativo en cada iteración.

- El método primal dual, el cual se basa en que sólo cuando las soluciones tanto del problema dual como el del problema primal son iguales, éstas son las óptimas.
- El método Simplex, el cual se refiere al método que trae la columna con el coeficiente del costo más negativo a la base.

## 2.2 **Optimización de cuotas carreteras.-**

Cómo ya se ha mencionado, los trabajos de investigación acerca de caminos de cuotas son muy abundantes, sin embargo la mayoría de estos se enfocan en reducir los congestionamientos y demás efectos negativos que esto conlleva (cómo la contaminación) mediante la regulación de la demanda (Cropper & Oates, 1992).

El congestionamiento es un problema importante y cada vez más común. Los esfuerzos realizados para resolver este problema no han sido exitosos en gran parte debido a que una vez construidas las soluciones no se ha garantizado que la nueva capacidad sea apropiadamente utilizada. Los esfuerzos por atraer a la gente fuera de sus vehículos han sido igualmente ineficaces (Button, 2004).

La tarificación vial es un concepto muy simple que extiende la común práctica de utilizar precios para reflejar la escasez y asignar recursos para su uso más eficiente que se da en prácticamente cualquier otro sector de la economía. La introducción del concesionamiento de líneas en Singapur en 1975 es el clásico caso de estudio de una aplicación pionera, pero a pesar de su éxito no es comúnmente visto como un ejemplo relevante para las ciudades de Europa Occidental y Norte América. Sin embargo el esquema de Singapur ofrece evidencias de los efectos de la tarificación vial. La política inicial de simplemente cobrar a los vehículos que entraban al centro de la ciudad entre las 7:30 am y las 9:30 am demostró ser inadecuada ya que el tráfico simplemente se

desplazaba hacia fuera de los períodos de concesión. Poco tiempo después este período de extendió hasta las 10:15 am y se incluyó un período por las tardes. El impacto de este nuevo esquema fue impresionante desde un principio ya que logró una reducción de 24,700 automóviles durante los períodos pico.

| Time | Vehicle | May 1975 | May 1976 | May 1977 | May 1978 | May 1979 |
|---|---|---|---|---|---|---|
| 0700-0730 | Cars | 5384 | 5675 | 6488 | 6723 | 5723 |
| | | | (+5.4) | (+14.3) | (+3.6) | (-14.9) |
| | Car pools | 176 | 509 | 636 | 606 | 497 |
| | | | (-17.5) | (+25.0) | (-4.7) | (-18.1) |
| | Total | 9800 | 10332 | 11489 | 11692 | 10596 |
| | | | (+5.4) | (+11.2) | (-1.8) | (-11.9) |
| 0730-1015 | Cars | 42790 | 10754 | 10350 | 11350 | 13181 |
| | | | (-74.9) | (-3.8) | (+9.7) | (+16.1) |
| (Control | Car pools | 2369 | 4641 | 5337 | 5684 | 5756 |
| Period) | | | (+95.9) | (+15.0) | (+6.5) | (+1.3) |
| | Total | 74014 | 37587 | 44318 | 47503 | 49606 |
| | | | (-49.2) | (+17.2) | (+7.2) | (+4.4) |
| 1015-1045 | Cars | n.a. | 6459 | 6636 | 6326 | 5527 |
| | | | | (+2.7) | (-4.7) | (-12.6) |
| | Car pools | n.a. | 320 | 280 | 281 | 232 |
| | | | | (-12.5) | (+3.2) | (-17.4) |
| | Total | n.a. | 13441 | 13805 | 14308 | 15179 |
| | | | | (+2.7) | (+3.6) | (+6.1) |

(Parenthesis indicates percentage change over previous year.)

**Fig. 3 Evolución del tráfico con la aplicación del esquema de concesión de calles en Singapur (Button, 2004).**

Un ejemplo más cercano lo tenemos en la autopista de cuota Monterrey-Saltillo, la cual fue inaugurada el año 2009. En un principio se fijo una tarifa de $39.00 pesos por utilizar dicha vía, pero tan sólo un par de meses después y sin registrar aumento alguno en sus flujos dicha tarifa se aumentó a $49.00 pesos, y en menos de un año se volvió a aumentar en $11.00 pesos para llegar a una tarifa de $60.00 pesos. Lo que nos muestra claramente que la determinación de dichas tarifas no había sido la correcta (tal vez por mala planeación o estimación de la demanda que tendría esta vía, o simplemente porque fue determinada de una forma impositiva sin hacer un buen estudio previo). De ahí la importancia de este tema, ya que de haberse realizado un buen trabajo de planeación y estudio previo no fuera necesario el tener que modificar las cuotas de una manera tan constante.

Mekky realizó una investigación precisamente de acerca de esto en su trabajo "Toll Revenue and Traffic Study of Highway 407 in Toronto" (Mekky, 1995), en el cual se estimó el futuro tráfico y posibles beneficios que pudiera llegar a tener la autopista 407 de cuota en la ciudad de Toronto que en esos momentos se encontraba en construcción. Las conclusiones a las que este autor llegó fueron que esta nueva autopista no sólo ayudaría a descongestionar la vieja autopista 401 y disminuiría los tiempos de viaje de los usuarios, sino que también podría generar beneficios sustanciales, al grado de poder llegar a ser considerada una infraestructura auto financiable. Lo que nos demuestra que este tipo de infraestructura no tiene por qué ser considerada tan sólo como una inversión social que el gobierno debe de realizar, sino que si se realiza un buen plan de trabajo pueden llegar a ser desarrollada como un exitoso modelo de negocio.

En el articulo "A survey of road pricing" (Morrison, 1986) se repasa la literatura más relevante acerca del tema de tarificación vial, no sólo desde un punto de vista económico, sino también ingenieril y de planeación. En esta obra se toman ejemplos tanto empíricos como teóricos para demostrar la viabilidad e importancia de la tarificación vial.

En el trabajo "Congestion Toll Pricing of Traffic Networks" de los autores (Bergendor, Hearn, & Ramana, 1997) se trató el problema de asignación de cuotas en redes de transporte, pero buscando tan sólo evitar los congestionamiento mediante la óptima utilización de esta red, dejando de lado maximizar los beneficios obtenidos mediante los ingresos por las tarifas.

En los trabajos de (Beckmann, 1965), (Verhoef, 1996) y (Viton, 1995) si se consideró la maximización de los beneficios en una red de transporte de cuota, sin embargo se ha observado que en la práctica los análisis para estas cuestiones se hacen generalmente mediante simulaciones.

Por otro lado, la demanda de viajes es principalmente una demanda derivada, es decir, cada viaje es generalmente demandado no por sí mismo, sino como forma de consumir algún otro beneficio o servicio. Debido a que las actividades asociadas con el transporte tienden a variar con el tiempo, la demanda para estos viajes no es constante. Por ejemplo, la mayoría de las ciudades sufren de congestionamientos durante las horas de entrada y salida a los trabajos (mañanas y tardes), mientras que las rutas vacacionales experimentan congestionamientos estacionales (en los veranos por ejemplo). Y debido a que toda infraestructura de transporte tiene una capacidad finita, cuando los usuarios de una ruta en particular empiezan a interferir con otros usuarios debido a esta limitante, empiezan a surgir los problemas por congestionamientos. Y aunque cierto grado de congestionamiento es deseable para evitar la sub utilización de la capacidad, la congestión excesiva presenta un gran número de problemas. Por lo cual surge la interrogante de ¿Cuál es el nivel óptimo de congestionamiento?

En el libro "Traffic Assignment Problem: Models and Methods" del autor (Patriksson, 1994) se revisa la evolución de los modelos y métodos para el problema de la estimación de los flujos de equilibrio de tráfico en las zonas urbanas y se muestran los alcances y limitaciones de los modelos de tráfico actuales. Y debido a que para parte de este trabajo se está buscando asignar los flujos óptimos en una red de transporte, los temas tratados en este libro vienen siendo de gran utilidad para comprender de una mejor forma el problema que se desea resolver.

Enfocándonos específicamente en el tema de este trabajo de investigación, el cual trata acerca de la asignación de cuotas para redes de transporte por carreteras para la maximización de los beneficios, el autor (Camacho Vallejo, 2009) desarrolla y compara varios modelos de resolución binivel para este problema en particular. Los modelos que compara el autor en cuestión y que se tomarán para este trabajo están basados en métodos de aproximación por gradiente, Quasi-

newton y un método de resolución directa respectivamente, obteniendo muy buenos resultados con cada uno de estos. De esta forma podemos confirmar la viabilidad de el enfoque binivel para abordar este problema.

Por otro lado, en el trabajo "A Bi-level Optimization Model for Port Selection" del autor (Celaya Bustamante, 2010) se abordó un problema muy similar, en el que para su resolución se desarrolló un modelo se optimización binivel utilizando la meta heurística "Scatter Search", basado en la implementación que los autores (Martí & Laguna, 2003) desarrollaron en su libro "Scatter Search: Methodology and Implementations in C". La cual resultó ser efectiva al arrojar muy buenos resultados.

Para el presente trabajo de investigación se tiene la de idea de que, tomando como base el modelo utilizado en el trabajo de (Celaya Bustamante, 2010), un modelo de optimización binivel para resolver el problema de asignación de cuotas carreteras desarrollado con la meta heurística Scatter Search y apoyado por el optimizador CPLEX, puede arrojar mejores resultados que los obtenidos en el trabajo del autor (Camacho Vallejo, 2009).

# Capítulo 3.

## Herramientas de Optimización.-

En este capítulo se verá un repaso de las herramientas de optimización utilizadas para resolver el problema de asignación de cuotas carreteras. Como ya se mencionó anteriormente, este problema fue abordado como uno de programación binivel, por lo que se utilizaron dos herramientas diferentes para su resolución (una para cada nivel). El optimizador Cplex versión 11.1 para el nivel inferior y la meta heurística Scatter Search para el nivel superior. Todo esto programado en lenguaje C con el software Visual Basic 6.0.

### 3.1 Optimización Binivel.-

Los problemas de programación binivel son problemas de optimización jerárquica, donde un tomador de decisiones puede ser capaz de influir en el comportamiento de otro tomador de decisiones en un nivel inferior sin tomar por completo el control de sus acciones. En un sistema multinivel, la función objetivo y el espacio de decisiones de una unidad puede ser determinado por variables controladas por otra unidad en un nivel paralelo o subordinado.

La importancia de la programación binivel se basa en el hecho de que la toma de decisiones en cualquier organización grande raramente procede de un solo punto de vista.

Los sistemas multinivel comparten las siguientes características (Bard, 1998):

- Existe una toma de decisiones interactiva en los diferentes niveles jerárquicos.
- Cada nivel subordinado lleva a cabo sus políticas solo después de que un nivel superior lleve a cabo sus decisiones.

- Cada unidad optimiza sus beneficios netos de forma independiente, pero se ven afectadas por las acciones de otras unidades.
- Los efectos externos sobre el problema de un tomador de decisiones se reflejan en la función objetivo y el conjunto factible de soluciones.

La programación multinivel se definió por primera vez en la década de los 70s por Chandler y Norton como una generalización de la programación matemática, en su trabajo ilustran cómo dos niveles de programación pueden ser utilizados para analizar la dinámica de una economía regulada, centrándose en el desarrollo agrícola del norte de México. (Chandler & Norton, 1977).

Para poder formular el problema de una forma matemática, se supone que el líder tiene control sobre el vector $x \in X \subseteq R^n$, y a su vez, que el seguidor tiene control sobre el vector $y \in Y \subseteq R^m$. El líder empieza, y selecciona una $x$ tratando de minimizar $F(x, y(x))$, la cual puede estar sometida a ciertas restricciones. La componente $y(x)$ indica que el problema del líder está implícito en las variables $y$. Después de observar las acciones del líder, el seguidor reacciona seleccionando una $y$ que minimice su función objetivo $f(x, y)$, sujeta a un conjunto de restricciones.

Este problema se puede definir de la siguiente forma:

$$\min_{x \in X} \quad F(x, y)$$
$$subject\ to \quad G(x, y) \leqq 0$$

$$\min_{y \in Y} \quad f(x, y)$$
$$subject\ to \quad g(x, y) \leqq 0$$

Dónde $F, f: R^n \times R^m \rightarrow R^1$, $G: R^n \times R^m \rightarrow R^p$ y $g: R^n \times R^m \rightarrow R^q$. Los conjuntos *X* e *Y* pueden tener restricciones adicionales en sus variables, como la no negatividad o la integridad de sus elementos. Además se puede jugar con la formulación para generar nuevas formas del

problema, como por ejemplo cambiar el operador “min” por uno “max” de acuerdo a las necesidades especificas del problema tratado en cuestión.

## 3.2 **Scatter Search.-**

Scatter Search (o búsqueda dispersa en español) es una meta heurística evolutiva que ha venido siendo aplicada de forma exitosa en problemas difíciles de optimización. Está basada en formulaciones y estrategias desarrolladas en los 60s, pero no fue sino hasta 1977 que fue propuesto por Glover como un método en sí. En este año Glover describió a Scatter Search como “un método que usa una sucesión de inicializaciones coordinadas para generar soluciones” (Glover, 1977).

Scatter Search trabaja con un conjunto de soluciones llamado *conjunto de referencia,* que son combinadas para crear soluciones nuevas. A diferencia de los algoritmos genéticos, este conjunto de referencia tiende a ser pequeño y está basado en métodos y diseños sistemáticos, en lugar de asignación al azar pura al crear nuevas soluciones. Esta metodología no sólo considera la calidad de las soluciones propuestas, sino también la diversidad del *conjunto de referencia*. Scatter Search puede ser descrito de la siguiente forma generalizada (Martí & Laguna, 2003):

1. **Método de Generación y Diversificación.-**

   Es un generador de soluciones diversas. Se basa en generar un conjunto *P* de soluciones diversas, del cual se extraerá un subconjunto más pequeño (el conjunto de referencia o *RefSet*).

2. **Método de Mejoramiento.-**

   Es un método de mejora basado, generalmente, en un método de búsqueda local y debe de ser capaz de tomar una solución no factible y obtener de esta una que si lo sea, para

después mejorar su valor. Si el método no logra obtener una mejor solución a la original, se considera que el resultado es la propia solución inicial.

**3. Método de Creación y Actualización.-**

Como su nombre lo indica, este método crea y actualiza el conjunto de referencia *RefSet* que consiste en las mejores soluciones extraídas del conjunto de soluciones iniciales *P*. Las soluciones en este conjunto se ordenan de mejor a peor respecto de su calidad.

3.1 Creación. Se inicializa el conjunto de referencia con las *b*/2 mejores soluciones de *P*. Las *b*/2 restantes se extraen procurando maximizar la mínima distancia con las ya incluidas en el conjunto de referencia para favorecer la diversidad.

3.2 Actualización. Las soluciones que surjan de las combinaciones pueden entrar en el conjunto de referencia y reemplazar a alguna de las ya incluidas, para que de esta forma el conjunto mantenga su tamaño, pero el valor de sus soluciones vaya mejorando a lo largo de la búsqueda.

**4. Método de Generación de Subconjuntos.-**

Se generan subconjuntos del conjunto de referencia *RefSet* para aplicarles un método de combinación. Aquí se debe especificar la forma en que se seleccionarán estos subconjuntos para aplicarles el método de combinación, generalmente se restringe la búsqueda a parejas de soluciones para que de esta forma el método pueda considerar a todas las parejas que se pueden formar con el conjunto de referencia y aplicarles el método de combinación a todas.

**5. Método de Combinación de Soluciones.-**

En este método se tratan de combinar todas las soluciones del conjunto de referencia. Estas soluciones combinadas pueden ser introducidas en el conjunto de referencia o almacenadas temporalmente en una lista hasta terminar de realizar todas las combinaciones para después ver qué soluciones entran en éste.

La figura 4 muestra el esquema básico descrito anteriormente. Se debe notar que este algoritmo se detiene cuando al tratar de combinar no existen nuevos elementos en el conjunto de referencia.

Si bien este esquema general puede ser implementado en un sin número de problemas de investigación (Laguna y Martí enlistan entre los diferentes ejemplos de aplicación a: ruteo de vehículos, el problema del agente viajero, ruteo de arcos, asignación cuadrática, diseño de productos financieros, formación de redes neuronales, creación de grafos, ordenamiento lineal, optimización sin restricciones, asignación multi objetivo, el problema del árbol, carga de vehículos, programación de trabajos de taller, etc.) no es un esquema único para aplicar Scatter Search y siempre conviene revisar las ultimas implementaciones realizadas.

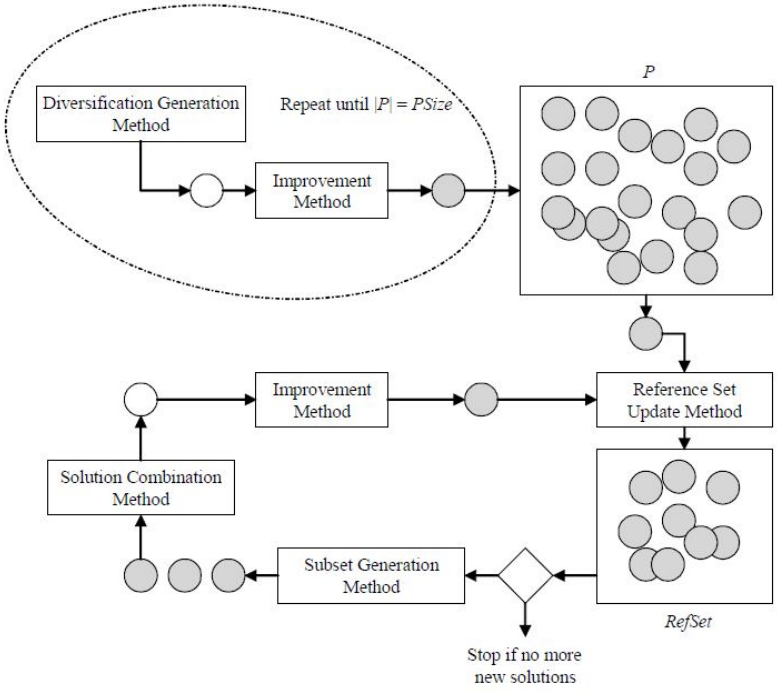



**Fig. 4 Representación de Scatter Search. (Laguna & Martí, 2003)**

# Capítulo 4.

# Formulación.-

Como ya se ha mencionado, el problema de optimización de cuotas carreteras puede ser abordado desde la perspectiva de un líder y su seguidor, los cuales interactúan en una red multiproducto $G = (K, N, A)$ definida por un conjunto de productos (orígenes-destino) $K$, un conjunto de nodos $N$ y el conjunto de arcos $A$. Este último conjunto a su vez es particionado en el subconjunto $A_1$ que representa a los arcos con cuotas y su subconjunto complementario $A_2$, que representa a los arcos libres. Se dotó a cada arco $a \in A$ con un costo de viaje generalizado $C_a$ que representa el costo mínimo por viajar a través de cada arco. También se debe considerar que cada arco $a \in A$ en la red tiene un límite en su capacidad $q_a$. Así como existe una capacidad fija para cada arco y un conjunto de productos $K$, también existe una componente $n^k$ que representa a la demanda existente para cada producto entre los nodos de origen $o(k)$ y destino $d(k)$ asociados con el producto $k \in K$. Con cada producto se asocia además un vector de demanda $b^k$, cuyos componentes son (para cada nodo $i$ de la red):

$$b_i^k = \begin{cases} -n^k if\ i = o(k), \\ n^k if\ i = d(k), \\ 0\ de\ otra\ forma. \end{cases}$$

Por último se introdujo la componente $t_a$ en los arcos de cuota $a \in A_1$ que representa un costo extra por viajar dentro de estos arcos, la cual debe ser determinada. Dejando de esta forma a $x_a^k$ representar a los flujos entre los arcos para cumplir con las demandas de cada uno de los productos existentes.

Ahora, si consideramos que las cuotas $t_a$ no pueden exceder un máximo preestablecido $t_a^{max}$ y que los flujos $x_a^k$ deben ser parte de la solución óptima del nivel inferior que a su vez es parametrizado por el vector de cuotas del nivel superior, este problema puede ser formulado como un programa binivel con restricciones lineales.

| | |
|---|---|
| *K:* | Conjunto de productos. |
| *N*: | Conjunto de nodos. |
| *A*: | Conjunto de arcos. |
| $A_1$: | Conjunto de arcos con cuota. |
| $A_2$: | Conjunto de arcos sin cuota. |
| $a = (i, j)$: | Elemento de A. |
| $i^-$ : | Nodo destino asociado al nodo $i : i^- = \{(k, i) \in A; k \in N\}$. |
| $i^+$: | Nodo origen asociado al nodo $i : i^+ = \{(i, k) \in A; k \in N\}$. |
| $c_a$: | Costo generalizado de viajar en el arco $a \in A$, independiente de la cuota. |
| $q_a$: | Capacidad del arco $a \in A$. |
| $t_a$: | Cuota del arco $a \in A_1$. |
| $t_a^{max}$: | Cota superior de las cuotas asociadas con los arcos $a \in A_1$. |
| $b^k$: | Demanda del vector asociado con el producto $k \in K$. |
| $x_a^k$: | Vector de flujos asociados con el producto $k \in K$. |

**Tabla 1: Notación para la formulación del problema de optimización de cuotas carreteras.**

De esta forma nos queda que el problema de optimización de cuotas carreteras puede ser formulado de la siguiente forma:

$$\max_{t,x} \sum_{k\in K} \sum_{a\in A_1} t_a\, x_a^k \qquad (1)$$

$$sujeto\ a{:}\ t_a \le t_a^{max} \qquad \forall a \in A_1 \quad (2)$$

$$t_a \ge 0 \qquad \forall a \in A_1 \quad (3)$$

$$\forall k \in K \quad \begin{cases} x^k \in \operatorname{argmin}_x \sum_{a\in A_1}(c_a + t_a)x_a^k + \sum_{a\in A_2} c_a\, x_a^k \\ sujeto\ a{:}\ -\sum_{a\in i^-} x_a^k + \sum_{a\in i^+} x_a^k = b_i^k \qquad \forall i \in N \quad (4) \\ x_a^k \ge 0 \qquad \forall a \in A \end{cases}$$

$$\sum_{k\in K} x_a^k \le q_a \qquad \forall a \in A \quad (5)$$

Conjunto de ecuaciones 4:

$$\forall k \in K \quad \begin{cases} x^k \in \operatorname{argmin}_x \sum_{a\in A_1}(c_a + t_a)x_a^k + \sum_{a\in A_2} c_a\, x_a^k \qquad (4.1) \\ sujeto\ a{:}\ -\sum_{a\in i^-} x_a^k + \sum_{a\in i^+} x_a^k = b_i^k \qquad \forall i \in N \quad (4.2) \\ x_a^k \ge 0 \qquad \forall a \in A \quad (4.3) \end{cases}$$

El objetivo del líder es maximizar los beneficios totales, los cuales son la suma de la multiplicación entre las cuotas $t_a$ y el flujo de usuarios en el arco *A* (ecuación 1). Sin embargo el conjunto de restricciones inferiores (conjunto de ecuaciones 4) obligan a que el seguidor asigne flujos a los caminos más cortos con respecto a las cuotas actuales, es decir, el objetivo del nivel inferior es minimizar el costo total de las rutas seleccionadas por los usuarios (ecuación 4.1). Las restricciones del nivel inferior se encargan de la conservación (ecuación 4.2) y la no negatividad en

los flujos (ecuación 4.3). Por último se agrega una restricción para no sobrepasar las capacidades en los arcos con los flujos asignados (ecuación 5).

Otro punto a considerar es que para evitar soluciones triviales se asumió que no existen costos negativos por viajar en la red y que para cada producto existe al menos una ruta libre de cuotas, esto junto con la ecuación (2), que establece una cota superior para las cuotas, evita que el valor de las cuotas se dispare demasiado dando resultados triviales.

# Capítulo5.

## Modelo de Optimización Binivel.-

El problema de asignación de cuotas carreteras puede ser formulado como un modelo de optimización binivel, en dónde el nivel superior consiste en una administración que establece las cuotas de la red buscando maximizar los ingresos, mientras que el nivel inferior está representado por un grupo de usuarios que viajan en los caminos más cortos con respecto a un costo de viaje generalizado. Para poder lograr que el modelo sea binivel se le ha agregado a la función objetivo del líder una restricción del tipo *argmin* que optimiza los flujos en los arcos de la red de transporte una vez que se han asignado las cuotas, para de esta forma encontrar un equilibrio entre las cuotas establecidas y el número de usuarios que usan dichos caminos.

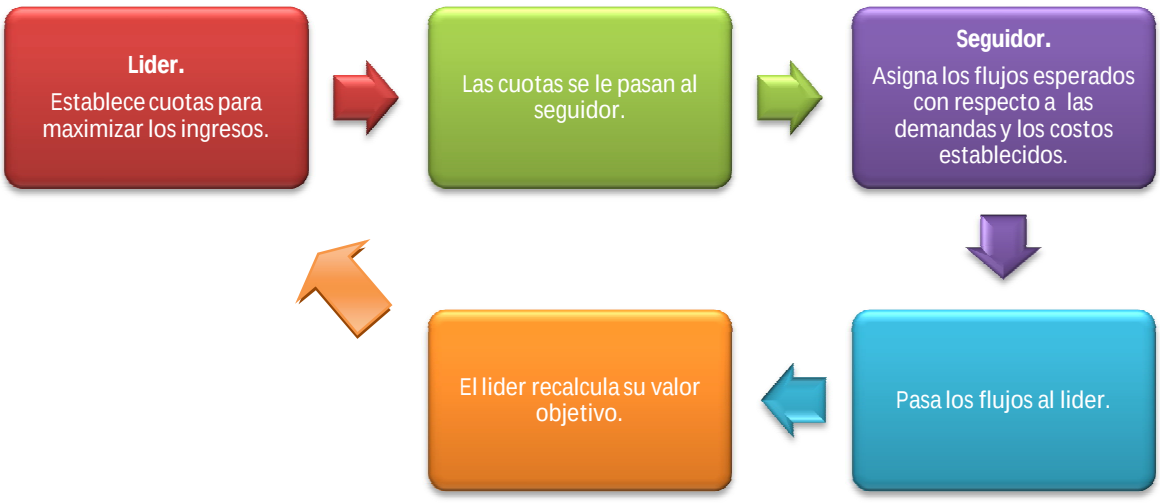



**Fig. 5 Diagrama general del modelo de optimización binivel.**

Para este modelo se tomó como base la implementación que desarrollaron los autores (Laguna & Martí, 2003) de la meta heurística "Scatter Search" en su libro "Scatter Search: Methodology and Implementations in C" para resolver el nivel superior que busca determinar la combinación optima de cuotas para maximizar los beneficios en la red de transporte y se utilizó el optimizador CPLEX para resolver el problema del nivel inferior, el cual trata de determinar los flujos óptimos en la red de transporte que minimicen los costos por viajar en ella.

El modelo desarrollado empieza resolviendo el problema del nivel inferior sin considerar las variables de las cuotas del nivel superior, es decir, resuelve un problema simple de flujo de costo mínimo. Una vez establecidos los flujos iniciales, se genera un conjunto inicial de *P* soluciones diversas para las variables de las cuotas del nivel superior de forma aleatoria pero controlada a la vez, para garantizar que las soluciones iniciales generadas estén dentro de los rangos preestablecidos (en este caso establecidos como 0 y $t_a^{max}$ para las cotas inferiores y superiores respectivamente). Para generar estas soluciones se inicia dividiendo los rangos de cada variable (entre 0 y $t_a^{max}$) en 4 sub rangos de igual tamaño. Después se selecciona uno de estos sub rangos de forma aleatoria y se genera una solución que quede dentro del sub rango seleccionado.

Una vez generado el conjunto inicial de soluciones se procede a mejorar dichas soluciones, y debido a que el método de generación de soluciones sólo construye soluciones que queden dentro de los rangos permitidos, el método de mejora siempre empezará a trabajar con una solución factible. El método de mejora utilizado consiste en el método simplex de Nelder y Mead (Nelder & Mead, 1965) el cual es un optimizador

clásico para problemas de optimización no lineales sin restricciones. Este método requiere de un parámetro de entrada que le especifique el número de evaluaciones que realizara contra la función objetivo, naturalmente, entre más evaluaciones se realicen la calidad de las soluciones aumentará.

El siguiente paso es el de extraer las mejores soluciones generadas para formar el sub conjunto de referencia. Este sub conjunto debe estar formado en un 50% por soluciones seleccionadas por su calidad con respecto a la función objetivo y el 50% restante por su diversidad. Para lograr esto se ordenan a las soluciones de mejor a peor con respecto a su calidad y selecciona a las primeras del 50% de la cantidad que se desea para el subconjunto de referencia (por ejemplo, si se desea formar un subconjunto de referencia de 10, se seleccionarían a las primeras 5 soluciones de la lista) y se borran del conjunto inicial *P*. Después se calcula la mínima distancia euclidiana entre el resto de las soluciones que quedan en el conjunto inicial *P* y las soluciones seleccionadas para el sub conjunto de referencia. Se selecciona a la solución con la máxima de las mínimas distancias calculadas, se agrega al sub conjunto de referencia y se borra del conjunto inicial *P*. Una vez hecho esto el proceso de repite hasta completar el tamaño deseado para el sub conjunto de referencia, el cual como resultado de todo esto posee a las soluciones con la mayor calidad y diversidad.

Una vez hecho esto, se generan sub conjuntos del a su vez sub conjunto de referencia para aplicarles un método de combinación. Este método de combinación consiste en crear 3 soluciones de prueba para cada par de soluciones del sub conjunto de referencia.

La primera solución de prueba se calcula de la siguiente forma:

$$C1.x = x^{'} - d$$

La segunda solución de prueba se calcula de la siguiente forma:

$$C2.x = x^{'} + d$$

La tercera solución de prueba se calcula de la siguiente forma:

$$C3.x = x^{''} + d$$

En dónde *d* es igual a:

$$d = r\frac{x^{''} - x^{'}}{2}$$

Y *r* representa un número generado de forma aleatoria en el rango (0, 1).

Una vez generadas las soluciones con el método de combinación, son sometidas al método de mejora y se selecciona a la mejor solución resultante. Si esta solución es mejor que alguna del sub conjunto de referencia se reemplaza por la solución que ha sido mejorada. Este ciclo se repite hasta que el sub conjunto de referencia ya no cambia y todas las soluciones en el sub conjunto de referencia han sido sometidas al método de combinación. En este punto el método de diversificación es usado para reconstruir la mitad del conjunto de referencia y continuar con la búsqueda.

Todo este procedimiento se realiza un número pre establecido de iteraciones, al terminar este número de iteraciones el método a llegado a su final.

Cabe mencionar que cada vez que se genera un conjunto de soluciones nuevas para las variables del nivel superior, se re calculan los valores de los flujos del problema del nivel inferior con los nuevos valores generados y estos flujos nuevos son utilizados para medir la calidad de las soluciones generadas al calcular el valor de la función objetivo.

## 5.1 **Ejemplos de Aplicación.-**

Para probar el modelo binivel propuesto se tomaron 3 diferentes grafos de redes de transporte multiproducto. Para el primer grafo se corrieron 8 ejemplos. Tanto el grafo como los parámetros fueron tomados de la tesis: "Comparing various algorithms performance: application to bilevel toll setting problem" (Camacho Vallejo, 2009). Para el segundo y tercer grafo se corrieron 6 ejemplos (tomados de la misma fuente). Finalmente el último ejemplo probado es una variación del segundo grafo en el cual se le agregó un producto más, esto con la intención de demostrar que el método es capaz de resolver instancias más grandes sin mayores problemas.

El primer grafo es una red de transporte que consta de 7 nodos y 12 arcos (7 de cuota, indicados por la línea punteada, y 5 libres). La estructura de dicho grafo es la siguiente:

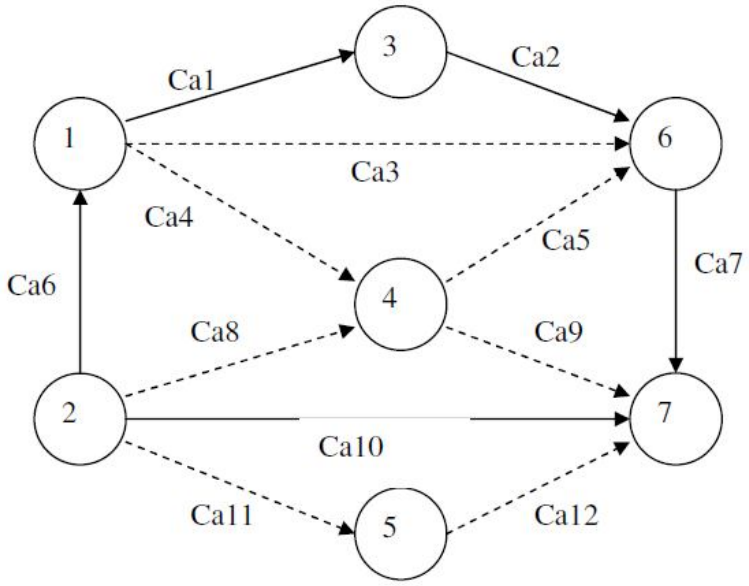



**Fig. 6 Grafo 1. (Camacho Vallejo, 2009)**

El segundo grafo utilizado consta de una red de 20 nodos y 35 arcos (15 de cuota y 20 libres), de igual forma los arcos de cuotas están indicados por la línea punteada. La estructura de este segundo grafo es la siguiente:

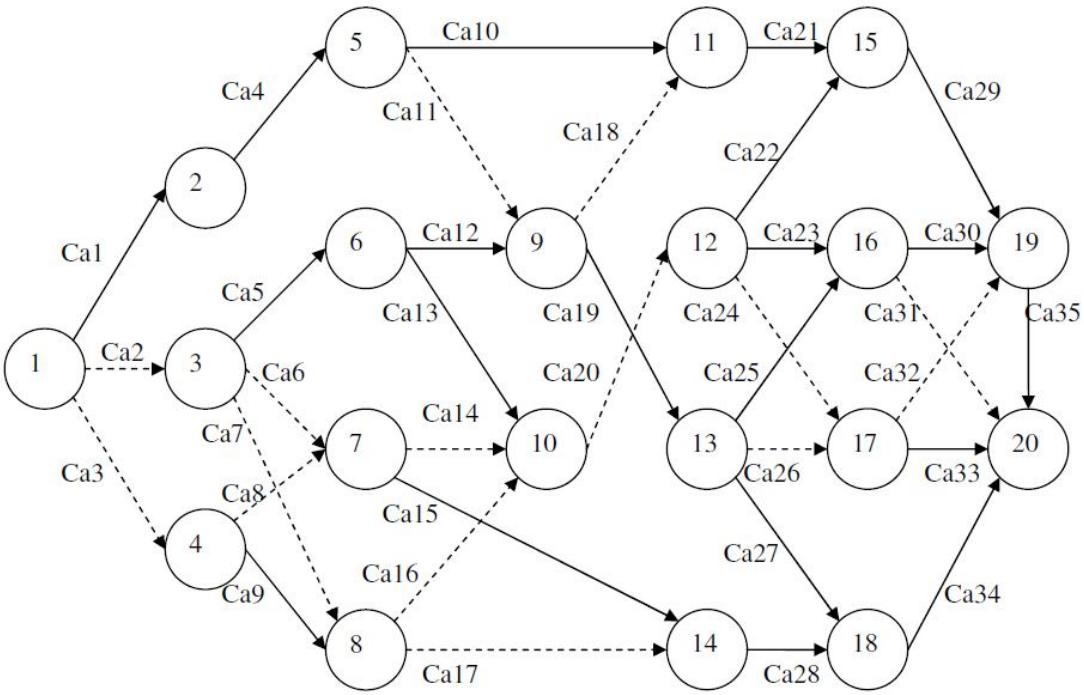



**Fig. 7 Grafo 2. (Camacho Vallejo, 2009)**

El tercer grafo está formado por 25 nodos y 40 arcos (20 de cuota y 20 libres). De nuevo, los arcos de cuota vienen indicados por una línea punteada. La estructura de este grafo es como sigue:

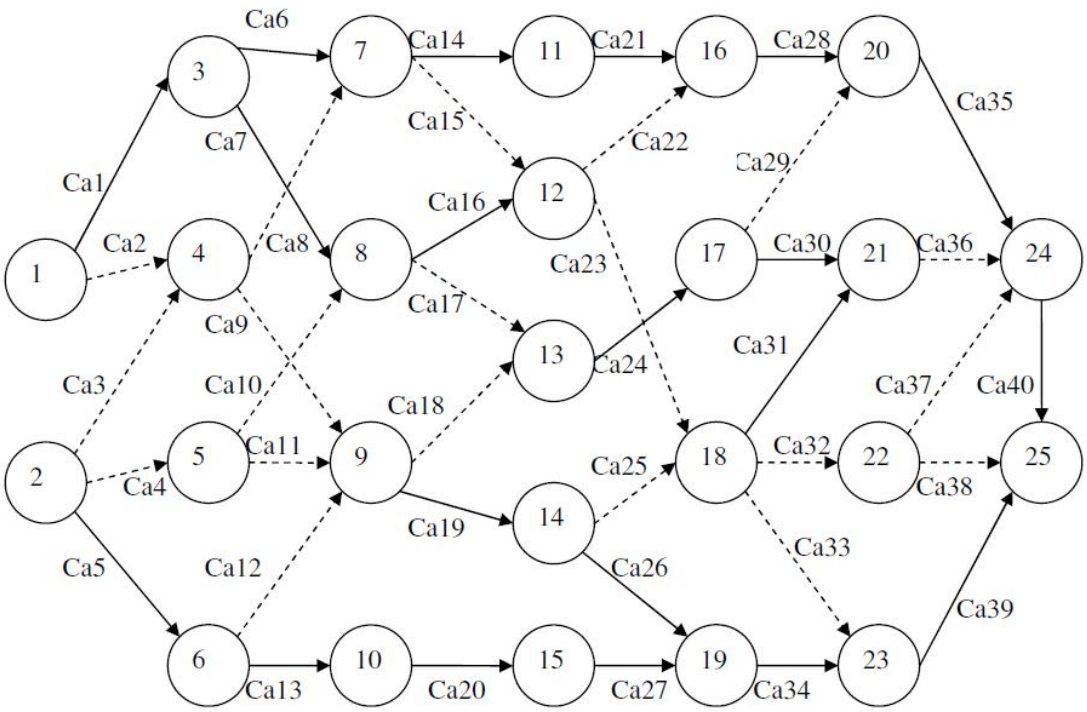



**Fig. 8 Grafo 3. (Camacho Vallejo, 2009)**

Los parámetros utilizados para cada uno de los diferentes ejemplos vienen indicados en el anexo **B**.

## 5.2 **Programa Computacional**

El programa desarrollado para resolver este problema fue codificado en lenguaje C y compilado con el software de Microsoft Visual C++ 6.0. Todos los ejemplos fueron ejecutados en una computadora Dell OptiPlex 960, con un procesador Intel Core 2 Quad de 2.66 GHz y 3.21 GB de RAM, bajo el sistema operativo Microsoft Windows XP Profesional con Service Pack 3.

Como ya se ha mencionado anteriormente, para resolver el nivel superior del método binivel se utilizó la meta heurística Scatter Search, para la cual los autores Martí y Laguna publicaron una implementación en C de esta metodología (Martí & Laguna, 2003), la cual fue aprovechada para este trabajo de investigación.

Para resolver el nivel inferior se utilizaron las librerías del Software CPLEX 11.1 para optimización de la compañía IBM. Las funciones *main* y *sol_value* del código desarrollado se encuentran en los anexos **C** y **D** respectivamente.

La función *main* se encarga de establecer las principales variables utilizadas y de inicializar un ambiente para que CPLEX pueda resolver el nivel inferior, mientras que la función *sol_value* re calcula la función objetivo del seguidor y calcula la función objetivo del líder para cada iteración.

# Capítulo 6.

## Resultados.-

Se corrieron 21 diferentes ejemplos para probar el método propuesto en este trabajo de investigación. Estos ejemplos pueden ser divididos para facilitar su análisis de la siguiente forma:

- Grafo 1. De 12 arcos (8 ejemplos).
- Grafo 2. De 35 arcos (6 ejemplos).
- Grafo 3. De 40 arcos (6 ejemplos).
- Variante del grafo 2 con 4 productos (1 ejemplo).

El objetivo de escoger estos ejemplos de aplicación para probar el modelo (a excepción de la variante del grafo 2) fue el de compararlos con los resultados obtenidos en la tesis "Comparing various algorithms performance: application to bilevel toll setting problem" (Camacho Vallejo, 2009) en el cual comparó el desempeño de cuatro diferentes algoritmos (un método de aproximación por gradiente, un algoritmo Quasi-Newton y un algoritmo de resolución directo) para la resolución de este mismo problema. El último ejemplo se seleccionó con la finalidad de probar que este modelo trabaja bien inclusive para instancias de más de 3 productos (en las cuales los modelos del trabajo citado se vuelven ineficientes).

Cabe señalar que no sólo se compararon los métodos por su valor de función objetivo obtenidos, sino también en cuanto al tiempo que demoraron en encontrar dicha solución. Los resultados obtenidos en el presente trabajo de investigación son los mostrados en la última columna.

- Resultados de los ejemplos del Grafo 1:

| Grafo de 12 arcos | | | | |
|---|---|---|---|---|
| Ejemplo | Aprox. Gradiente | Quasi-Newton | Directo | **SS/CPLEX** |
| 1 | 162.850 | 162.880 | 162.360 | **180.990** |
| 2 | 274.890 | 274.890 | 274.340 | **274.950** |
| 3 | 109.850 | 109.850 | 108.870 | **199.990** |
| 4 | 150.860 | 150.840 | 150.320 | **146.550** |
| 5 | 112.860 | 112.860 | 112.040 | **133.840** |
| 6 | 203.950 | 203.960 | 202.820 | **199.973** |
| 7 | 41.970 | 41.960 | 41.650 | **41.976** |
| 8 | 104.950 | 104.950 | 104.110 | **125.990** |

**Tabla 2 Valores de la función objetivo para los ejemplos del Grafo 1.**

- Resultados de los ejemplos del Grafo 2:

| Grafo de 35 arcos | | | | |
|---|---|---|---|---|
| Ejemplo | Aprox. Gradiente | Quasi-Newton | Directo | **SS/CPLEX** |
| 1 | 1,342.235 | 1,342.763 | 1,341.835 | **1,328.983** |
| 2 | 7,184.852 | 7,184.804 | 7,184.422 | **7,112.128** |
| 3 | 1,577.953 | 1,577.911 | 1,577.661 | **1,686.028** |
| 4 | 420.701 | 420.772 | 420.257 | **1,022.000** |
| 5 | 764.931 | 764.961 | 763.890 | **1,514.000** |
| 6 | 2,350.855 | 2,350.875 | 2,350.356 | **3,004.492** |

**Tabla 3 Valores de la función objetivo para los ejemplos del Grafo 2.**

- Resultados para los ejemplos del Grafo 3:

| Grafo de 40 arcos | | | | |
|---|---|---|---|---|
| Ejemplo | Aprox. Gradiente | Quasi-Newton | Directo | **SS/CPLEX** |
| 1 | 1,456.802 | 1,456.833 | 1,456.034 | **1,631.644** |
| 2 | 2,247.873 | 2,247.851 | 2,246.889 | **2,611.994** |
| 3 | 3,891.831 | 3,891.873 | 3,891.445 | **3,263.977** |
| 4 | 5,621.804 | 5,621.820 | 5,621.109 | **11,339.610** |
| 5 | 3,433.720 | 3,433.794 | 3,432.771 | **3,419.900** |
| 6 | 544.871 | 544.893 | 543.908 | **868.594** |

**Tabla 4 Valores de la función objetivo para los ejemplos del Grafo 3.**

- Resultado para los ejemplos de la variante del Grafo 2:

| Grafo de 35 arcos | |
|---|---|
| Ejemplo | **SS/CPLEX** |
| 1 | **1,368.657** |

**Tabla 5 Valores de la función objetivo para el ejemplo de la variante del Grafo 2 con 4 productos.**

Como se puede observar, el resultado obtenido para la función objetivo del líder se mejoró para la gran mayoría de los casos, y para aquellos en los que no se pudo mejorar el resultado obtenido no quedó muy alejado del logrado por cualquiera de los métodos propuestos en el trabajo.

A continuación se muestra una tabla en la que se despliegan los porcentajes de incremento obtenidos por el método binivel propuesto basado en Scatter Search y CPLEX (para cada uno de los diferentes ejemplos) con respecto al mejor resultado obtenido en el trabajo previo.

- Porcentajes de incremento para los ejemplos del Grafo 1:

| Ejemplo | Incremento |
|---|---|
| 1 | 11.12% |
| 2 | 0.02% |
| 3 | 82.06% |
| 4 | -2.86% |
| 5 | 18.59% |
| 6 | -1.95% |
| 7 | 0.01% |
| 8 | 20.05% |

**Tabla 6 Porcentajes de Incremento para los ejemplos del Grafo 1.**

- Porcentajes de incremento para los ejemplos del Grafo 2:

| Ejemplo | Incremento |
|---|---|
| 1 | -1.03% |
| 2 | -1.01% |
| 3 | 6.85% |
| 4 | 142.89% |
| 5 | 97.92% |
| 6 | 27.80% |

**Tabla 7 Porcentajes de incremento para los ejemplos del Grafo 2.**

- Porcentajes de incremento para los ejemplos del Grafo 3:

| Ejemplo | Incremento |
|---|---|
| 1 | 12.00% |
| 2 | 16.20% |
| 3 | -16.13% |
| 4 | 101.71% |
| 5 | -0.40% |
| 6 | 59.41% |

**Tabla 8 Porcentajes de incremento para los ejemplos del Grafo 3.**

Logrando obtener de esta forma una media de **28.66%** de incremento en los valores de la función objetivo para todos los ejemplos probados en el presente trabajo de investigación con respecto al trabajo previo.

Como ya se mencionó anteriormente, no sólo se comparó el valor de la función objetivo logrado por el método propuesto, sino también el tiempo que tardó dicho método en encontrar este valor para cada uno de los diferentes ejemplos. Los resultados obtenidos se muestran en la siguiente tabla (de igual forma los tiempos logrados por el método propuesto en el presente trabajo de investigación son aquellos que se encuentran en la última columna de la tabla):

- Tiempos de demora para la resolución para los ejemplos del Grafo 1:

| Ejemplo | Aprox. Gradiente | Quasi-Newton | Directo | **SS/CPLEX** |
|---|---|---|---|---|
| 1 | 2.107 | 2.023 | 2.950 | **1,620.000** |
| 2 | 2.449 | 2.433 | 3.235 | **1,320.000** |
| 3 | 1.573 | 1.426 | 2.035 | **1,503.000** |
| 4 | 3.414 | 3.173 | 3.415 | **1,527.000** |
| 5 | 2.006 | 1.940 | 2.002 | **1,461.000** |
| 6 | 2.178 | 2.156 | 3.782 | **1,442.000** |
| 7 | 1.308 | 1.277 | 1.259 | **1,449.000** |
| 8 | 3.065 | 2.899 | 1.959 | **1,620.000** |

**Tabla 9 Tiempos (en segundos) para la resolución de los ejemplos del Grafo 1.**

- Tiempos de demora para la resolución para los ejemplos del Grafo 2:

| Ejemplo | Aprox. Gradiente | Quasi-Newton | Directo | **SS/CPLEX** |
|---|---|---|---|---|
| 1 | 601.000 | 517.000 | 723.000 | **1,710.000** |
| 2 | 781.000 | 748.000 | 917.000 | **5,100.000** |
| 3 | 492.000 | 437.000 | 714.000 | **2,656.000** |
| 4 | 278.000 | 220.000 | 439.000 | **1,853.000** |
| 5 | 644.000 | 591.000 | 835.000 | **2,095.000** |
| 6 | 831.000 | 806.000 | 980.000 | **2,593.000** |

**Tabla 10 Tiempos (en segundos) para la resolución de los ejemplos del Grafo 2.**

- Tiempos de demora para la resolución para los ejemplos del Grafo 3:

| Ejemplo | Aprox. Gradiente | Quasi-Newton | Directo | **SS/CPLEX** |
|---|---|---|---|---|
| 1 | 1,004.000 | 985.000 | 1,054.000 | **7,560.000** |
| 2 | 525.000 | 498.000 | 893.000 | **2,820.000** |
| 3 | 439.000 | 421.000 | 922.000 | **3,480.000** |
| 4 | 1,723.000 | 1,755.000 | 1,480.000 | **2,778.000** |
| 5 | 764.000 | 785.000 | 761.000 | **3,030.000** |
| 6 | 89.000 | 84.000 | 835.000 | **3,520.000** |

**Tabla 11 Tiempos (en segundos) para la resolución de los ejemplos del Grafo 3.**

- Tiempo de demora para la resolución de la variante del Grafo 2:

| Ejemplo | **SS/CPLEX** |
|---|---|
| 1 | **3,281.000** |

**Tabla 12 Tiempo (en segundos) de resolución de la variante del Grafo 2 con 4 productos.**

Es notable el incremento en el tiempo requerido para la resolución de los ejemplos por el método propuesto basado en Scatter Search y CPLEX, pero para poder hacer un mejor análisis de esto se presenta a continuación una tabla con la diferencia entre el tiempo requerido por este método contra el mejor tiempo logrado por alguno de los métodos propuestos en el trabajo previo.

- Diferencias entre los tiempos para los ejemplos del Grafo 1:

| Ejemplo | Diferencia |
|---|---|
| 1 | 1,617.977 |
| 2 | 1,317.567 |
| 3 | 1,501.574 |
| 4 | 1,523.827 |
| 5 | 1,459.060 |
| 6 | 1,439.844 |
| 7 | 1,447.741 |
| 8 | 1,618.041 |

**Tabla 13 Diferencias (en segundos) entre el tiempo logrado contra el mejor tiempo previo.**

- Diferencias entre los tiempos para los ejemplos del Grafo 2:

| Ejemplo | Diferencia |
|---|---|
| 1 | 1,193.000 |
| 2 | 4,352.000 |
| 3 | 2,219.000 |
| 4 | 1,633.000 |
| 5 | 1,504.000 |
| 6 | 1,787.000 |

**Tabla 14 Diferencias (en segundos) entre el tiempo logrado contra el mejor tiempo previo.**

- Diferencias entre los tiempos para los ejemplos del Grafo 3:

| Ejemplo | Diferencia |
|---|---|
| 1 | 6,575.000 |
| 2 | 2,322.000 |
| 3 | 3,059.000 |
| 4 | 1,298.000 |
| 5 | 2,269.000 |
| 6 | 3,436.000 |

**Tabla 15 Diferencias (en segundos) entre el tiempo logrado contra el mejor tiempo previo.**

Obteniendo de esta forma que se tiene una media de **2,178.632** segundos (o lo que vendría siendo igual **36.31** minutos) de diferencia entre los tiempos logrados para cada uno de los diferentes ejemplos probados por el método propuesto en el presente trabajo de investigación con respecto a los tiempos logrados por el trabajo de investigación previo.

Con el objetivo de corroborar la razón por la cual este método incrementa tanto el tiempo de respuesta, y de esta forma poder ayudar a los posibles trabajos futuros a comprender mejor como solucionar este problema, se midió el tiempo en que tarda el método tan sólo por las veces que se llama al optimizador CPLEX, sin contar el resto de las operaciones que realiza, para el ejemplo que

demoró mayor cantidad tiempo en cada uno de los Grafos. Los resultados obtenidos se muestran a continuación:

Para el Grafo 1 se tomó el primer ejemplo.-

| Tiempo Total | Tiempo CPLEX | Diferencia | Llamadas a CPLEX |
|---|---|---|---|
| 1,620 | 1,531.615 | **88.385** | 407 |

**Tabla 16 Comparación del tiempo de resolución (en segundos) entre sólo CPLEX y el problema completo del ejemplo 1 del Grafo 1.**

Lo que nos da un promedio de **3.763 segundos** por llamada a CPLEX.

Para el Grafo 2 se tomó el segundo ejemplo.-

| Tiempo Total | Tiempo CPLEX | Diferencia | Llamadas a CPLEX |
|---|---|---|---|
| 5,100 | 5,061.1485 | **38.851** | 402 |

**Tabla 17 Comparación del tiempo de resolución (en segundos) entre sólo CPLEX y el problema completo del ejemplo 2 del Grafo 2.**

Lo que nos da un promedio de **12.589 segundos** por llamada a CPLEX.

Para el Grafo 3 se tomó el primer ejemplo.

| Tiempo Total | Tiempo CPLEX | Diferencia | Llamadas a CPLEX |
|---|---|---|---|
| 7,560 | 7,504.743 | **55.275** | 401 |

**Tabla 18 Comparación del tiempo de resolución (en segundos) entre sólo CPLEX y el problema completo del ejemplo 1 del Grafo 3.**

Lo que nos da un promedio de **18.715 segundos** por llamada a CPLEX.

Con lo que se puede comprobar que efectivamente la razón por la que el método demora tanto en resolver el problema es la forma en que se resuelve el nivel inferior.

# Capítulo 7.

# Conclusiones y Trabajos Futuros.-

## 7.1 Conclusiones.-

Al terminar este trabajo de investigación se puede concluir que es posible mantener rentable un sistema de caminos de cuota frente a la competencia de caminos libres, siempre y cuando se empleen las herramientas adecuadas y se tenga una buena planeación, ya que como vimos a pesar de que para cada destino existía al menos un camino libre siempre se tuvieron beneficios, lo cual indica que los usuarios decidían hacer uso de esta infraestructura.

También se observó que el método propuesto de resolución para el problema de determinación de cuotas carreteras basado en la meta heurística Scatter Search y el optimizador CPLEX dio mejoras significativas con respecto a los métodos propuestos en el trabajo de investigación previo (Camacho Vallejo, 2009) (un incremento del 28.56% para ser precisos entre todos los ejemplos probados), los cuales consistían en uno de aproximación por gradiente, uno algoritmo Quasi-Newton y un último algoritmo de resolución directo.

Un punto importante a tomar en cuenta es la determinación de la cota máxima para las tarifas en los arcos de cuotas, ya que de establecerse de forma errónea pudiera darse el caso en se restrinja demasiado al modelo y no se obtenga el máximo beneficio posible de la red (si se establece una cuota demasiado baja) o se sub utilice está red de transporte al obligar a los clientes a irse por los arcos libres (si se establece una cuota demasiado alta).

Por otro lado, si bien es cierto que los beneficios (en términos del valor logrado en la función objetivo) al utilizar este modelo son significativamente mayores, debe considerarse que de igual forma se incrementó el tiempo de obtención de los resultados (en 33.462 minutos en promedio para cada ejemplo), por lo cual se recomienda hacer un análisis de costo/beneficio para la situación en particular para la cual se pudiera necesitar resolver este problema. Sacrificar un poco de beneficios a favor de lograr resultados de una forma más inmediata, o sacrificar tiempo a favor de lograr mayores beneficios en el resultado obtenido.

## 7.2 **Trabajos Futuros.-**

Para los posibles trabajos futuros se sugieren las diferentes mejores y recomendaciones que se presentan a continuación:

- Utilizar datos de casos reales para observar el comportamiento del modelo propuesto en el presente trabajo.
- Utilizar algún otro meta heurístico para comparar su desempeño frente a Scatter Search.
- Lograr reducir el tiempo de respuesta del modelo. Tratar de hacer más eficiente el código o experimentar con otros lenguajes de programación.
- Desarrollar nuevos modelos para resolver el problema de asignación de cuotas carreteras para compararlo tanto con el modelo propuesto en el presente trabajo de investigación como con los anteriores.

## Referencias


Al-Kazili, J. (1982). Modeling containerized shipping for developing countries. *Transportation Research: Part A. Vol 16, No. 4.* , 271-283.

Anandalingam, G., & Friesz, T. (1992). Hierarchical optimization. *Annals of Operations Research Vol. 34, No. 1.* , 1-11.

Bard, J. F. (1998). *Practical Bilevel Optimization. Algorithms and Applications.* Austin: Kluwer Academic Publishers.

Beckmann, M. (1965). On optimal tolls for highway tunnels and bridges. *In: Vehicular Traffic Science, EDIE L; HERMAN, R; ROTHERY R; Ed. Elsevier, New York* , 331-341.

Bergendor, P., Hearn, D. W., & Ramana, M. (1997). *Congestion in Toll Pricing of Traffic Networks.* Gainesville: University of Florida. Dept. of Industrial & Systems Engineering.

Button, K. (2004). *Road Pricing.* Fairfax: Pergamon Press.

Camacho Vallejo, J. F. (2009). Comparing various algorithms performance: application to bilevel toll setting problem. *Monterrey, ITESM - Ph.D. Dissertation.*

Camacho, J. F., Kalashnikov, V., & Kalashnykov, N. (2008). A penalty function approach to solve the bilevel tolls problem. *3rd International Conference on Innovative Computing Information and Control. Innovative Computing Information and Control* , 217.

Celaya Bustamante, C. A. (2010). A Bilevel Optimization Model for Port Selection. Monterrey: ITESM - Tesis de Maestría.

Chandler, W., & Norton, R. (1977). *Multi-Level Programming and development policy. Número 258 de World Bank staff working paper.* University of Texas: World Bank.

Colson, B., Marcotte, P., & Savard, G. (2007). An overview of bilevel optimization. *Ann Oper Res Vol. 153, No. 1.* , 235–256.

Colson, B., Marcotte, P., & Savard, G. (2005). Bilevel Programming - A survey. *4OR: A Quarterly Journal of Operations Research. Vol. 3, No. 2.* , 87-107.

Cropper, M. L., & Oates, W. E. (1992). Environmental economics: a survey. *Journal of Economic Literature. Vol. 30, No. 2* , 675-740.

Edmonds, K. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM. Vol. 19, No. 2* , 248 - 264.

Garrido, A., & Onaindía, E. (1999). Un algoritmo para a optimización de rutas de transporte. *Jornadas de Transferencia Tecnológica de Inteligencia Artificial. Vol. 2, No. 4* , 1-8.

González-Velarde, J. L., & Martí, R. (2008). Adaptive memory programing for the robust capacitated international sourcing problem. *Computers & Operations Research Vol. 35, No. 3* , 797 – 806.

González-Velarde, J. L., Álvarez, A. M., & De Alba, K. (2005). Grasp Embedded Scatter Search for the Multicommodity Capacitated Network Design Problem. *Journal of Heuristics, Vol. 11 , No. 3* , 233 - 257.

Gottinger, H., & Weimann, H. (1990). *Artificial Intelligence. A Tool for Industry and Management.* Ellis Horwood Ltd.

Laguna, M., & Martí, R. (2003). *Scatter Search: Methodology and Implementations in C.* Norwell: Kluwer Academic Publishers.

Martí, R., & Laguna, M. (2003). Scatter Search: Basic Design and Advanced Strategies. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. Vol. 7, No.19* , 123-130.

Mekky, A. (1995). Toll Revenue and traffic study of highway 407 in toronto. *Transportation research record. No. 1498* , 5-15.

Morrison, S. A. (1986). A survey of road pricing. *Transportation Research Part A: General. Vol. 20, No. 2* , 87-97.

Nelder, & Mead. (1965). Nelder and Meads simplex method. *Computer Journal Vol. 7, No. 4* , 308-313.

Patriksson, M. (1994). *The Traffic Assignment Problem: Models and Methods.* VSP International Science Publishers.

Taha, H. (2004). *Investigación de Operaciones. 7a Edición.* México: Pearson Educación.

Verhoef, E. (1996). *Economic efficiency and social feasibility in the regulation of road transport externalities. Volume 108 of Tinbergen Institute research series.* Thesis Publ.

Viton, P. A. (1995). Private roads. *Jorunal of Urban Economics. Vol. 37, No. 3* , 260-269.

Wright, S. J. (1997). *Primal-dual interior-point methods.* Philadelphia: Siam.

Zadeh, N. (1973). A bad network problem for the simplex method and other minimum cost flow algorithms. *Mathematical Programming 5. Vol. 5, No. 1.* , 255-266.

## Anexo A: Scatter Search - Pseudo código

***Algoritmo*** *Scatter Search*

1. *Comenzar con P = Ø. Utilizar el* ***método de generación*** *para construir una solución y el* ***método de mejora*** *para tratar de mejorarla; sea x la solución obtenida. Si* $x \notin P$ *entonces añadir x a P. (i.e.,* $P = P \cup x$*), en otro caso, rechazar x. Repetir esta etapa hasta que P tenga un tamaño prefijado.*
2. *Construir el* ***conjunto de referencia*** *RefSet* $= \{ x^1, \ldots, x^b \}$ *con las b/2 mejores soluciones de P y las b/2 soluciones de P más diversas a las ya incluidas.*
3. *Evaluar las soluciones en RefSet y ordenarlas de mejor a peor respecto a la función objetivo.*
4. *Hacer NuevaSolución = TRUE*

***Mientras*** *(NuevaSolución)*

5. *NuevaSolucion = FALSE*
6. *Generar los subconjuntos de RefSet en los que haya al menos una nueva solución.*

   ***Mientras*** *(Queden subconjuntos sin examinar)*

   7. *Seleccionar un subconjunto y etiquetarlo como examinado.*
   8. *Aplicar el* ***método de combinación*** *a las soluciones del subconjunto.*
   9. *Aplicar el* ***método de mejora*** *a cada solución obtenida por combinación. Sea x la solución mejorada:*
      *Si(* $f(x) < f(x^b)$ *y x no está en RefSet)*
   10. *Hacer* $x^b = x$ *y reordenar Refset*
   11. *Hacer NuevaSolucion = TRUE*

**Fig. 9 Algoritmo Scatter Search. (Martí & Laguna, 2003).**

## Anexo B: Parámetros

A continuación se presentan los listados para los parámetros utilizados en los ejemplos resueltos en el presente trabajo de investigación. Se describen los costos fijos para cada arco $c_a$, la demanda de cada producto $n^k$, los productos $k = [o(k), d(k)]$ en dónde $o(k)$representa el nodo de origen y $d(k)$el nodo de destino para cada producto k. Por último las capacidades de los arcos vienen denotadas por $q_a$.

Los parámetros considerados para los 8 ejemplos del Grafo 1 son los siguientes:

| Ejemplo | Parámetros |
|---|---|
| 1 | $c$ = (7,4,1,2,5,3,8,4,3,12,3,2) , $k$ = {(1,6), (2,7)}, $q$ = (25,25,10,8,8,25,25,5,5,25,9,9) , $n^k$ = (10,9) |
| 2 | $c$ = (9,9,3,4,2,5,6,2,3,15,3,4) , $k$ = {(1,6), (2,7)}, $q$ = (20,20,18,18,18,20,20,18,18,20,18,18) , $n^k$ = (15,5) |
| 3 | $c$ = (8,9,5,2,2,1,3,2,3,10,1,2) , $k$ = {(1,6), (2,7)}, $q$ = (16,16,10,10,10,16,16,10,10,16,10,10) , $n^k$ = (10,10) |
| 4 | $c$ = (5,4,5,3,1,2,4,2,3,13,1,1) , $k$ = {(1,6), (2,7)}, $q$ = (17,17,10,10,10,17,17,10,10,17,10,10) , $n^k$ = (5,12) |
| 5 | $c$ = (7,7,3,4,5,8,10,3,3,9,6,2) , $k$ = {(1,6), (2,7)}, $q$ = (19,19,6,6,6,19,19,6,6,19,6,6) , $n^k$ = (10,9) |
| 6 | $c$ = (5,10,8,2,3,2,1,3,5,12,4,2) , $k$ = {(1,6), (2,7)}, $q$ = (26,26,8,8,8,26,26,8,8,26,8,8) , $n^k$ = (17,9) |
| 7 | $c$ = (5,6,4,3,2,3,1,1,1,5,3,2) , $k$ = {(1,6), (2,7)}, $q$ = (13,13,3,3,3,13,13,3,3,13,3,3) , $n^k$ = (5,8) |
| 8 | $c$ = (5,5,3,1,1,3,5,6,2,10,5,4) , $k$ = {(1,6), (2,7)}, $q$ = (26,26,7,7,7,26,26,7,7,26,7,7) , $n^k$ = (11,15) |

Los parámetros utilizados para los ejemplos del Grafo 2 son los siguientes:

| Ejemplo | Parámetros |
|---|---|
| 1 | $c$ =(15,12,18,20,26,4,9,15,11,33,34,30,13,12,47,13,22,9,33,12,41,30,41, 11,16,15,13,30,10,8,19,23,20,21,5)<br><br>$q$ =(30,23,10,30,30,23,15,15,30,30,15,28,28,23,30,20,20,15,28,23,30,28, 25,23,30,18,28,28,30,28,17,15,30,25,20)<br><br>$k$ ={(1,20), (1,20), (1,20)}, $n^k$=(14,10,4) |
| 2 | $c$ =(29,14,15,24,29,17,8,30,28,32,3,3,29,11,12,12,5,29,2,12,25,25,1,3, 3,14,19,27,20,22,12,16,21,32,25)<br><br>$q$ =(90,71,17,90,90,50,71,17,30,90,25,80,80,17,20,71,50,90,80,20,88,28, 50,23,30,48,58,48,33,25,73,50,80,25,90)<br><br>$k$ ={(1,20), (1,20), (1,20)}, $n^k$=(25,15,53) |
| 3 | $c$ =(15,12,18,20,26,4,9,15,11,33,34,30,13,12,47,13,22,9,33,12,41,30,41, 11,16,15,13,30,10,8,19,23,20,21,5)<br><br>$q$ =(45,22,17,45,45,30,20,44,12,45,33,25,23,67,80,22,34,43,40,37,45,30, 52,37,45,20,30,45,50,30,29,40,37,29,40)<br><br>$k$ ={(1,20), (1,20), (1,4)}, $n^k$=(25,20,2) |
| 4 | $c$ = (16,10,7,27,28,4,17,17,23,8,14,29,30,3,31,6,19,9,23,23,17,9,29,20, 27,19,21,26,22,21,12,8,5,27,23)<br><br>$q$ = (30,14,1,30,20,15,16,14,20,40,25,18,18,33,20,30,25,25,18,33,32,18, 15,33,23,19,31,29,31,32,18,16,20,26,22)<br><br>$k$ = {(1,13), (1,8),(1,20)}, $n^k$= (6,1,14) |
| 5 | $c$ = (14,8,20,2,21,5,1,22,6,9,18,17,16,9,18,16,14,23,11,1,11,9,13,10,10, 13,19,12,6,8,9,16,7,22,21)<br><br>$q$ = (43,21,4,43,30,23,15,15,43,43,15,29,25,23,43,24,32,15,21,23,43,21, 25,23,43,18,21,21,43,21,21,30,43,25,19)<br><br>$k$ = {(1,4), (1,13), (1,20)}, $n^k$= (4,9,28) |
| 6 | $c$ = (33,6,7,40,35,26,36,25,32,26,10,39,39,10,39,24,21,15,35,4,16,26, 27,36,32,18,31,39,37,26,37,5,17,39,35)<br>$q$ = (50,22,9,50,30,30,30,50,30,60,35,38,38,33,30,30,30,35,38,33,30,38, 35,33,30,38,50,46,30,38,37,35,50,55,50)<br><br>$k$ = {(1,7), (1,20), (1,13)}, $n^k$= (9,29,3) |

Los parámetros utilizados para los ejemplos del Grafo 3 fueron los siguientes:

| Ejemplo | Parámetros |
|---|---|
| 1 | $c$ = (4,1,3,4,10,12,11,2,1,2,2,2,11,12,2,9,2,4,11,4,10,5,1,9,7,13,16,12,9, 10,13,2,4,12,10,8,7,4,7,9)<br><br>$q$ = (70,33,40,30,70,70,25,10,14,30,25,18,70,70,11,20,20,15,28,70,70,18, 25,23,30,18,70,70,20,28,17,15,10,70,70,50,30,20,70,70)<br>$k$ = {(1,12), (2,19), (2,25)}, $n^k$= (12,24,30) |
| 2 | $c$ = (6,9,3,7,5,2,4,1,5,3,4,4,7,7,4,8,9,1,6,10,6,4,6,5,5,3,8,6,6,11,10,1,6,9, 3,7,7,4,5,10)<br><br>$q$ = (160,100,100,100,160,160,80,100,10,66,50,100,160,160,71,80,100, 45,41,160,160,100,100,30,90,100,160,160,100,42,100,100,100,160, 160,100,100,90,32,100)<br><br>$k$ = {(1,12), (2,19), (1,25)}, $n^k$= (31,41,120) |
| 3 | $c$ = (3,3,2,5,8,9,9,6,8,6,8,5,3,5,1,2,7,3,3,6,9,2,1,9,4,2,5,9,1,7,8,1,7,9,2,1, 6,5,2,9)<br><br>$q$ = (300,100,170,37,300,63,16,80,85,100,22,100,300,300,100,30,100,90, 100,300,300,100,300,200,200,100,300,300,150,150,180,110,110, 300,300,100,100,80,100,300)<br><br>$k$ = {(2,23), (2,19), (1,25)}, $n^k$= (101,107,98) |
| 4 | $c$ = (9,1,4,2,7,7,5,4,3,1,4,3,4,9,3,7,10,6,9,7,8,5,3,9,2,8,9,7,4,8,3,2,2,3,8, 3,2,6,9,9)<br><br>$q$ = (300,165,30,20,300,59,16,165,30,100,20,100,300,300,100,30,100, 90,100,300,300,100,300,200,200,50,300,300,150,150,55,110,110, 300,300,130,150,80,120,300)<br><br>$k$ = {(1,25), (2,19), (1,12)}, $n^k$= (165,50,59) |
| 5 | $c$ = (10,1,5,2,8,9,11,6,3,5,2,3,6,9,7,10,2,5,7,2,7,1,6,7,9,6,9,10,3,6,10,1,3, 5,8,8,1,1,5,9)<br><br>$q$ = (200,84,19,26,200,50,26,150,103,100,60,55,200,200,50,120,100, 200,33,200,200,100,200,100,100,50,200,200,40,30,55,110,110, 200, 200,130,150,80,120,200)<br><br>$k$ = {(1,25), (2,19), (2,25)}, $n^k$= (84,45,71) |

| | |
|---|---|
| 6 | $c$ = (7,4,3,6,3,5,10,4,4,3,2,3,10,9,3,10,2,7,10,10,7,3,4,7,5,8,11,10,7,10, 8,1,6,8,9,4,4,5,8,10)<br><br>$q$ = (4,6,2,11,30,20,26,8,10,10,60,50,30,30,50,20,10,30,22,30,30,40,30, 50,34,23,30,30,40,30,55,70,40,30,30,65,65,54,7,30)<br><br>$k$ = {(1,25), (2,23), (2,25)}, $n^k$= (10,6,8) |

Finalmente, los parámetros utilizados para la variante del ejemplo del Grafo 2 con 4 productos fueron los siguientes:

| **Ejemplo** | **Parámetros** |
|---|---|
| 1 | $c$ =(15,12,18,20,26,4,9,15,11,33,34,30,13,12,47,13,22,9,33,12,41,30,41, 11,16,15,13,30,10,8,19,23,20,21,5)<br><br>$q$ =(30,23,10,30,30,23,15,15,30,30,15,28,28,23,30,20,20,15,28,23,30,28, 25,23,30,18,28,28,30,28,17,15,30,25,20)<br><br>$k$ ={(1,20), (1,20), (1,20) , (1,19)}, $n^k$=(14,10,4,9) |

## Anexo C: Función main del Código de Programación

```
/*********************************
 *          Multinetwork.c          *
 *    Autor: Gabriel Pinto Serrano  *
 *                      2010        *
 *********************************/

#include <stdio.h>
#include <string.h>
#include <time.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#include <ilcplex/cplex.h>

//**Scatter Search**//
#include "SS1.h"
//SS

void   input_data  (char *filename, inputdata *pdata);
void setproblemdata (datacplex *pplex, inputdata pdata);


int    *ivector    (int nl, int nh);
int    **imatrix   (int nrl, int nrh, int ncl, int nch);
double *dvector    (int nl, int nh);
double **dmatrix   (int nrl, int nrh, int ncl, int nch);
char *cvector     (int nl, int nh);

CPXLPptr *CPXLPvector(int nl, int nh);

void   free_ivector(int *v, int nl);
void   free_dvector(double *v, int nl);
void   free_imatrix(int **m, int nrl, int nrh, int ncl);
void   free_dmatrix(double **m, int nrl, int nrh, int ncl);
void   nrerror     (char *error_text);

int main(int argc,char **argv)

{

      Int status;
      int k, i, j;
      int nvar;

      FILE *ofp;

      inputdata pdata;
      datacplex pcplex;

      CPXENVptr env = NULL;
      CPXLPptr  lp  = NULL;

      double *x;
```

```
      int cur_numrows;
      int cur_numcols;
      int cur_numnz;


//**Scatter Search**//
      SS *pb;                              /* Pointer problem           */
      int b       =   10;          /* Size of Reference Set          */
      int PSize   =  100;          /* Size of P                      */
      int Iter    =    1;          /* Number of Current Iteration    */
      int MaxIter =  50;           /* Maximum number of iterations */
      double  *sol;
      double value;
//SS


      if (argc != 3)
      {
            printf("Uso: multinetwork <archivo de entrada> <archivo de
            salida>\n");
            exit(1);
      }

      input_data(argv[1], &pdata);

      nvar = pdata.m;

      /* Initialize the CPLEX environment*/
      env = CPXopenCPLEX (&status);
      if ( env == NULL ) {
            char  errmsg[1024];
            fprintf (stderr, "Could not open CPLEX environment.\n");
            CPXgeterrorstring (env, status, errmsg);
            fprintf (stderr, "%s", errmsg);
      }

          setproblemdata (&pcplex,pdata);
            lp = CPXcreateprob(env, &status, pcplex.probname);
            status = CPXcopylp (env, lp, pcplex.numcols, pcplex.numrows,
            pcplex.objsen, pcplex.obj, pcplex.rhs,
            pcplex.sense, pcplex.matbeg, pcplex.matcnt, pcplex.matind,
            pcplex.matval, pcplex.lb, pcplex.ub, NULL);

            cur_numrows = CPXgetnumrows(env,lp);
            cur_numcols = CPXgetnumcols(env,lp);
            cur_numnz   = CPXgetnumnz(env,lp);

            x = dvector(0,cur_numrows-1);

            status = CPXcopyctype(env, lp, pcplex.ctype);
            status = CPXwriteprob(env, lp, "multinetwork.lp", NULL);
            status = CPXmipopt(env, lp);

            if(status) {
                  fprintf(stderr, "Failed to optimize MIP.\n");
                  goto TERMINATE;
```

```
            }

            status = CPXgetmipx (env, lp, x, 0, cur_numcols-1);

//CPX

                  //**Scatter Search**//

      pb = SSProblem_Definition(nvar,b,PSize);

      sol   = SSDouble_array(pb->nvar);

      SSCreate_P(pb, x, &pcplex, pdata, nvar, env, lp, cur_numcols);

      SSCreate_RefSet(pb);
//SS

                  for(Iter=1; Iter<MaxIter; Iter++)
      {

            if(pb->rs->NewSolutions){

                  SSUpdate_RefSet(pb, x, &pcplex, pdata, nvar, env, lp,
                  cur_numcols);

                        }
            else{

                  SSRebuild_RefSet(pb, x, &pcplex, pdata, nvar, env, lp,
                  cur_numcols);

                  }


      }
            //SS


      ofp = fopen(argv[2], "w");/* abre o crea archivo para escritura*/

/* imprime los resultados tanto en la pantalla como en el archivo de
salida */

      k = 0;
      for(j=1; j<= pdata.k; j++) {
            printf ("\nPara el producto %d:\n", j);
            fprintf (ofp, "\nPara el producto %d:\n", j);
            for(i=1; i<= (pdata.m+pdata.ml); i++) {
                        if(x[k]>0) printf ( "En el arco %d se tiene un
                        flujo de %f\n", i, x[k]);
                        if(x[k]>0) fprintf ( ofp, "En el arco %d se tiene
                        un flujo de %f\n", i, x[k]);
                        k++;
                  }
            }

      //**Scatter Search**//
```

```
   /* Print Best Solution Found */
      SSBestSol(pb,sol,&value);
      printf("\n\nBest Solution Found:\nValue = %f\n",value*-1);
      for(i=1;i<=nvar;i++) printf("x[%d]  = %f\n",i,sol[i]);

      free(sol+1);

      SSFree_DataStructures(pb);
   //SS


           if ( lp != NULL ) {
             status = CPXfreeprob (env, &lp);
        if ( status ) {
                  fprintf (stderr, "CPXfreeprob failed, error code
%d.\n", status);
             }        }


TERMINATE:


   /* Free up the problem as allocated by CPXcreateprob, if necessary */

   if ( lp != NULL ) {
      status = CPXfreeprob (env, &lp);
      if ( status ) {
         fprintf (stderr, "CPXfreeprob failed, error code %d.\n",
status);
      }
   }

   /* Free up the CPLEX environment, if necessary */

   if ( env != NULL ) {
      status = CPXcloseCPLEX (&env);

      /* Note that CPXcloseCPLEX produces no output,
         so the only way to see the cause of the error is to use
         CPXgeterrorstring.  For other CPLEX routines, the errors will
         be seen if the CPX_PARAM_SCRIND indicator is set to CPX_ON. */

      if ( status ) {
      char  errmsg[1024];
         fprintf (stderr, "Could not close CPLEX environment.\n");
         CPXgeterrorstring (env, status, errmsg);
         fprintf (stderr, "%s", errmsg);
      }
   }


         return (status);

/* End Main */}
```

## Anexo D: Función sol_value del Código de Programación

```
/*************************************************
 *
 *       SCATTER SEARCH SOL VALUE
 *
 *************************************************/
double sol_value(double sol[], double *x, datacplex *pcplex, inputdata
pdata, int nvar, CPXENVptr env, CPXLPptr lp, int cur_numcols)
{
      double value=0;
      int k;
      int j;
      int i;
      int cnt;
      int *indices;
      double *valores;

      int status = 0;

      cnt = nvar*pdata.k;
      indices   = ivector(0, cnt-1);
      valores   = dvector(0, cnt-1);

      //llena el vector de indices
            k=0;
            z=0;
            for (i = 1; i <= pdata.k; i++){
                  for(j = 0; j <= (pdata.m)-1; j++){
                        indices[k]=z;
                        k++;
                        z++;
            }
                  z=z+pdata.ml;
      }
            //CPXchgobj

            //llena el vector de los valores nuevos en la solucion
       k=0;
   for(i=1;i<=pdata.k;i++) {
         for(j=1;j<=pdata.m;j++) {
               valores[k]=pdata.a[j]+sol[j];
               k++;
         }
   }
//

            //CPXchgobj

            status = CPXchgobj (env, lp, cnt, indices, valores);
            status = CPXwriteprob(env, lp, "multinetwork.lp", NULL);

            status = CPXmipopt(env, lp);

            status = CPXgetmipx (env, lp, x, 0, cur_numcols-1);
```

```
      //CPXchgobj


            //Función Objetivo
            k=0;
      for(i=1;i<=pdata.k;i++) {
         for(j=1;j<=pdata.m;j++) {
               value  = value + (x[k]*sol[j]);
               k=k+1;
                  }
         k=k+pdata.ml;
   }

      value=-1*value;

            //Función Objetivo

      return value;
}
```

## Anexo E: Función para la Definición del Problema en el Código de Programación

```
SS *SSProblem_Definition(int nvar,int b,int PSize)
{
      int i;
      SS *pb;

      pb = (SS*) calloc(1,sizeof(SS));
    if(!pb) SSAbort("Memory allocation problem: SS");

      /* Problem Parameters */
      pb->CurrentIter = 0;
      pb->freq        = SSInt_matrix(nvar,4);

      /* Variable definition */
      pb->nvar = nvar;
      pb->high = SSDouble_array(nvar);
      pb->low      = SSDouble_array(nvar);
      for(i=1;i<=nvar;i++) {
            pb->low[i] = 0;
            pb->high[i] = 20;
      }

      /* Diverse Set */
      pb->p = (P*) calloc(1,sizeof(P));
    if(!pb->p) SSAbort("Memory allocation problem: P");

      pb->p->PSize  = PSize;
      pb->p->sol    = SSDouble_matrix(PSize,nvar);
      pb->p->ObjVal = SSDouble_array(PSize);

      /* Reference Set */
      pb->rs = (REFSET*) calloc(1,sizeof(REFSET));
    if(!pb->rs) SSAbort("Memory allocation problem: REFSET");

      pb->rs->b      = b;
      pb->rs->sol    = SSDouble_matrix(b,nvar);
      pb->rs->ObjVal = SSDouble_array(b);
      pb->rs->order  = SSInt_array(b);
      pb->rs->iter   = SSInt_array(b);

      /* Solution's Pool */
      pb->pool = SSDouble_matrix(4*b*b,nvar);

      return pb;
}
```

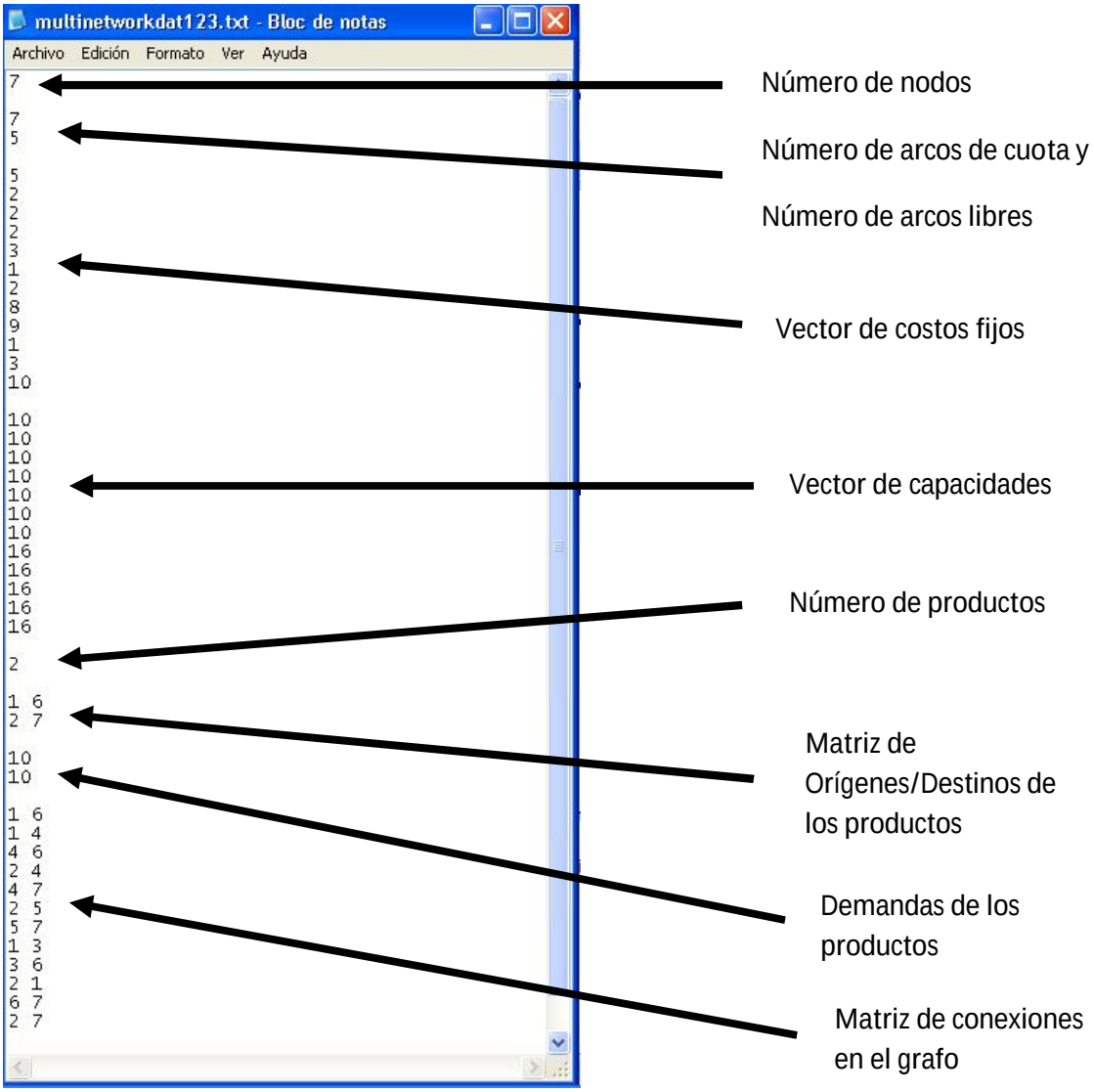
## Anexo F: Librería Principal Utilizada en el Código de Programación

```
#ifndef SS_H
#define SS_H

#include "stdio.h"
#include "stdlib.h"
#include "malloc.h"
#include "math.h"
#include "string.h"

#include <ilcplex/cplex.h>

typedef struct _inputdata {
      int    n;
      int    m;
      int    ml;
      int    k;
      int    **c;
      int    *a;
      int    *b;
      int    **o;
      int    *d;
} inputdata;


typedef struct _datacplex {
      char     probname[16];  // Nombre de problema
      int      numcols;
      int      numrows;
      int      objsen;
      double   *obj;
      double   *rhs;
      char     *sense;
      int      *matbeg;
      int      *matcnt;
      int      *matind;
      double   *matval;
      double   *lb;
      double   *ub;
      char     *ctype;
      int      *index;
      double   *values;
      int      *solstat;
      double   *objval;
      double   *z;
      double   *pi;
      double   *slack;
      double   *dj;

} datacplex;
```

```
typedef struct REFSET
{
      int   b;    /* Size*/
      double**sol;   /* Solutions*/
      double  *ObjVal;    /* Objective value of solutions*/
      int   *order;     /* Order of solutions*/
      int   *iter;            /* Input iteration number of solutions*/
      int   NewSolutions;     /* =1 if a new elements has been added*/

} REFSET;

typedef struct P
{
      int         PSize;      /* Size of the Set P */
      double      **sol; /* Solutions in the Set */
      double  *ObjVal;  /* Objective value of sols.*/

} P;

typedef struct SS
{
      int   nvar;        /* Number of variables of the problem*/
      double  *high;           /* Upper bound of variables*/
      double  *low;            /* Lower bound of variables*/

      int   **freq;            /* Frequency count for div. generator*/
      int   CurrentIter; /* Number of Current Iteration*/

      double  **pool;   /* Combined solutions*/
      int   pool_size;   /* Number of elements in pool*/

      REFSET  *rs;
      P       *p;

} SS;

/* Definitions */
#define SSGetrandom(min,max)  (int)( (SSRandNum()*(max+1-min)) + (min))
#define MAXPOSITIVE                       10000000

/* User defined function */
double sol_value(double sol[], double *x, datacplex *pcplex, inputdata
pdata, int nvar, CPXENVptr env, CPXLPptr lp, int cur_numcols);

/* Functions in file SSImprove1.c */
void  SSImprove_solution(SS *pb, double sol[],double *value, double *x,
datacplex *pcplex, inputdata pdata, int nvar, CPXENVptr env, CPXLPptr lp,
int cur_numcols);
double      SSMove(SS *pb,double *worst_point,double *worst_value, double
*psum, double factor, double *x, datacplex *pcplex, inputdata pdata, int
nvar, CPXENVptr env, CPXLPptr lp, int cur_numcols);
void  SS_Simplex(SS *pb,double **simplex,double *values,int max_eval,
double *x, datacplex *pcplex, inputdata pdata, int nvar, CPXENVptr env,
CPXLPptr lp, int cur_numcols);

/* Functions in file SSMemory1.c */
double      *SSDouble_array(int size);
```

```
double      **SSDouble_matrix(int nrows,int ncolumns);
void  SSFree_DataStructures(SS *pb);
void  SSFree_double_matrix(double **matrix,int nrows);
void  SSFree_int_matrix(int **matrix,int nrows);
int         *SSInt_array(int size);
int         **SSInt_matrix(int nrows,int ncolumns);
SS          *SSProblem_Definition(int nvar,int b,int PSize);

/* Functions in file SSP1.c */
void  SSCreate_P(SS *pb, double *x, datacplex *pcplex, inputdata pdata,
int nvar, CPXENVptr env, CPXLPptr lp, int cur_numcols);
void  SSGenerate_Sol(SS *pb,double sol[]);
void  SSPrint_P(SS *pb);

/* Functions in file SSRefSet1.c */
void  SSCombine(SS *pb,double sol1[],double sol2[],double **newsols);
void  SSCombine_RefSet(SS *pb);
void  SSCreate_RefSet(SS *pb);
void  SSPrint_RefSet(SS *pb);
void  SSRebuild_RefSet(SS *pb, double *x, datacplex *pcplex, inputdata
pdata, int nvar, CPXENVptr env, CPXLPptr lp, int cur_numcols);
void  SSTryAdd_RefSet(SS *pb,double sol[],double value);
void  SSUpdate_RefSet(SS *pb, double *x, datacplex *pcplex, inputdata
pdata, int nvar, CPXENVptr env, CPXLPptr lp, int cur_numcols);

/* Functions in file SSTools1.c */
void  SSAbort(char texto[]);
void    SSBestSol(SS *pb,double sol[],double *value);
double  SSDist_RefSet(SS *pb,int num,double sol[]);
int         SSEqualSol(double sol1[],double sol2[],int dim);
int         SSIsInRefSet(SS *pb,double sol[]);
int         SSMax_dist_index(SS *pb,double dist[]);
int         *SSOrder(double weights[],int num,int type);
float SSRandNum(void);
void    SSUpdate_distances(SS *pb,double min_dist[],int rs_index);

#endif  /* SS_H */
```

## Anexo G: Ejemplo de Archivo de Entrada para el Programa



## Anexos Adicionales:

Anexos adicionales (como el código de programación completo y los archivos de entrada usados) fueron incluidos en un CD junto con este trabajo de investigación.

## Datos Personales

**Gabriel Pinto Serrano** nació en la ciudad de Hermosillo, Sonora en México el día 14 de Agosto del año 1985. Estudió la carrera de Ingeniería Industrial en la Universidad del Valle de México, Campus Hermosillo, de la cual se tituló en el año 2007, y una maestría en Ciencias con especialidad en Sistemas de Manufactura en el Instituto Tecnológico y de Estudios Superiores de Monterrey, Campus Monterrey, de la cual se tituló en el año 2010. Tiene experiencia e intereses por las aéreas de manufactura (especialmente de la industria automotriz), planeación de la producción y logística.

Dirección Permanente:

Gabriel Pinto Serrano.
Teléfono: +52 (662) 2145766
Calle: Herminio Ciscomani #610
Colonia: Pitic, C.P. 83150
Hermosillo, Sonora, México
gabrielpinto_s@hotmail.com