SEMINARIO DE INVESTIGACIÓN:
ALGUNOS MÉTODOS DE SOLUCIÓN PARA EL CVRP

DEISY CAROLINA CANTILLO CALDERÓN
MARILUZ ESPINOSA SANDOVAL
SILVIA ADRIANA GALVÁN NÚÑEZ
MARGARETH YESENIA ORTIZ GUZMÁN

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FÍSICO-MÉCANICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2010

SEMINARIO DE INVESTIGACIÓN:
ALGUNOS MÉTODOS DE SOLUCIÓN PARA EL CVRP

DEISY CAROLINA CANTILLO CALDERÓN
SILVIA ADRIANA GALVAN NUÑEZ
MARILUZ ESPINOSA SANDOVAL
MARGARETH YESENIA ORTIZ GUZMÁN

Trabajo de grado para optar al título de Ingeniera Industrial

Director
Dr. HENRY LAMOS DÍAZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FÍSICO-MÉCANICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2010

Doy gracias a Dios ante todo por permitirme vivir esta experiencia, por ser el centro de mi vida.

A mi madre querida por su apoyo incondicional, por su perseverancia, cariño, amor y por ser un ejemplo de mujer para mi y mis hermanos.

A mis hermanos y mi familia por creer en mi.

A Nando por ser un ángel que me ha acompañado desde el inicio de mi universidad y ahora ocupa una parte especial en mi vida.

A Humberto por su cariño, consejos y respaldo durante todo este tiempo, por ser esa persona tan especial.

A Blancus, Piedad, profe Henry, Claus, Maria, Moni, Diego, los Sergios, Jhon, Fer, a las chinitas OPALO, amigos que llevo en el corazón.

A todos mi compañeros de estudio y de trabajo por sus consejos y acompañamiento durante esta etapa tan importante.

Con Cariño,

Deisy

A Dios por acompañarme y fortalecerme todos los días.

A mis padres y hermanos, por la confianza que han depositado en mí,
por el apoyo incondicional
y por darme la oportunidad de estudiar.

A Esperanza, por creer en mis capacidades, sus consejos
y todos los momentos compartidos.

A Diego, por acompañarme durante todo este tiempo,
por su cariño y comprensión.

A Deisy, Silvia, Margareth, Sandra y Nelson por su amistad.

Mary

A mis papás por su apoyo constante, su amor,

su entrega total y su buen ejemplo.

A mis hermanos por su comprensión, orientación

y por enseñarme a creer en utopias.

A mi familia por ser la fuerza que me

impulsa a ser cada día mejor.

A Deimer, Nelson, Leydy, Fabián, Sandra, Miguel y Cindy por las largas sesiones de charlas y sobre todo por brindarme su amistad incondicional en el transcurso de mi carrera.

A Deisy, Mary, Margareth y el profe Henry

por tener la osadía de llevar a cabo este proyecto juntos.

A todos mis amigos y personas que de alguna forma hicieron que mi paso por la universidad fuera una de las etapas más gratas de mi vida.

Silvia

Agradezco a la vida por este logro.

A mis padres y hermano
por su perseverancia, cariño y compañía.

A toda mi familia
por su apoyo y confianza.

A Esther, Néstor, Nelson y Tatiana
por su amistad incondicional de tantos años
y por la fuerza que transmiten.

A Deisy, Mari y Silvia
por vivir juntas este proyecto.

A todos aquellos que siguieron cada avance
y sintieron este trabajo como propio.

Margareth

AGRADECIMIENTOS

A la Escuela de Estudios Industriales y Empresariales por formarnos como Ingenieras Industriales y ofrecernos su apoyo en el transcurso de la carrera.

Al Doctor Henry Lamos Díaz por enseñarnos, dirigirnos, acompañarnos y apoyarnos incondicionalmente en todo el proyecto, por ser el quinto integrante, por su profesionalismo y ante todo por ser nuestro amigo.

A los profesores de la Universidad Industrial de Santander por aportar a nuestro crecimiento profesional, en especial a los profesores de la Escuela de Estudios Industriales y Empresariales.

Al grupo OPALO por creer en nosotras y brindarnos su respaldo para que este proyecto tan importante para nosotras y para el grupo saliera adelante.

Al profesor Julio Carrillo por su orientación y apoyo en esta etapa de nuestra vida universitaria.

A los Ingenieros Fabián Cardozo y César Vargas por su amistad, apoyo y asesoría, por ser personas especiales para nosotras.

A todos nuestros amigos de la sede principal y del socorro que nos acompañaron durante nuestra vida universitaria, por los bellos momentos vividos, por los momentos de estudio, por ser nuestros amigos, por estar ahí siempre y ser parte importante de nuestras vidas.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

TÍTULO: SEMINARIO DE INVESTIGACIÓN: ALGUNOS MÉTODOS DE SOLUCIÓN PARA EL CVRP [*]

AUTORES: CANTILLO CALDERÓN, Deisy Carolina
GALVÁN NÚÑEZ, Silvia Adriana
ESPINOSA SANDOVAL, Mariluz
ORTIZ GUZMÁN, Margareth Yesenia [**]

PALABRAS CLAVES: RUTEO DE VEHÍCULOS, BRANCH AND BOUND, OPTMIZACIÓN COMBINATORIA, ALGORITMOS GENÉTICOS, COLONIA DE HORMIGAS.

DESCRIPCIÓN:

En este proyecto se presentan los conceptos básicos y la formulación matemática del problema del agente viajero, el problema del agente viajero múltiple y el problema de ruteo de vehículos con capacidad, seguidos del estudio del método exacto Branch and Bound, y las metaheurísticas Algoritmos genéticos y Colonia de hormigas como métodos de solución al problema de ruteo de vehículos con capacidad (CVRP).

Para el desarrollo de este trabajo se llevó a cabo una extensa revisión bibliográfica con la que se estableció el estado del arte del CVRP y las técnicas de solución mencionadas para resolverlo.
El estudio de cada una de las técnicas se realizó con la explicación de conceptos básicos generales y posteriormente enfocados a la solución de ejercicios específicos del CVRP. Se muestran las relajaciones básicas y las relajaciones avanzadas propuestas para encontrar una solución con la aplicación del método exacto en GAMS utilizando CPLEX como optimizador. Se explica detalladamente el algoritmo genético y el algoritmo colonia de hormigas junto con el desarrollo de un ejemplo del CVRP con solución en Matlab. Estos temas fueron recopilados en un libro, anexo de este documento.
Para mayor claridad se realizó un tutorial de GAMS que muestra el contenido de los temas desarrollados en el documento con la aplicación de diversos ejercicios. La herramienta de algoritmos genéticos de MatLab (Genetic Algorithm Tool) es ilustrada mediante la implementación de una instacia del CVRP, así como desarrollo de un algoritmo específico; de manera similar se explica el desarrollo de cada elemento del algoritmo Colonia de Hormigas. Los documentos pretenden guiar a los lectores en el manejo de estos programas/herramientas y sobre sus funciones principales.

Finalmente, el libro describe la metodología de trabajo utilizada por el grupo para el desarrollo del tema mediante la modalidad de seminario de investigación.

---

[*] Proyecto de grado en la modalidad Seminario de Investigación

[**] Facultad de ingenierías Físico Mecánicas, Escuela de Estudios Industriales y Empresariales. Dirigido por Henry Lamos Díaz.

ABSTRACT

TITTLE: RESEARCH SEMINAR: METHODS OF SOLUTION OF THE CVRP *

AUTHORS: CANTILLO CALDERÓN, Deisy Carolina
ESPINOSA SANDOVAL, Mariluz
GALVÁN NÚÑEZ, Silvia Adriana
ORTIZ GUZMÁN, Margareth Yesenia**

KEY WORDS: VEHICLE ROUTING, COMBINATORY OPTIMIZATION, BRANCH AND BOUND, GENETIC ALGORITHMS, ANT COLONY, METAHEURISTICS.

DESCRIPTION:

This project introduces the basic concepts and the mathematical formulation of the traveling salesman problem, the multiple traveling salesmen problem and the capacitated vehicle routing problem. The study adhered to the exact method Branch and Bound, as well as the metaheuristics, which consists of Genetic algorithms and the Ant Colony; as methods of resolving the capacitated vehicle routing problem (CVRP).

For the development of this work an extensive review of literature was carried out, which established the state-of-the-art techniques of the CVRP and the mentioned solution techniques in order to solve it.

The study of each technique is performed with an explanation of general concepts and then focuses on solving specific exercises of the CVRP. It exhibits the basic relaxations and also the proposed-advanced relaxations as solutions to the implementation of the exact method in GAMS using CPLEX as an optimizer. It explains in detail the genetic algorithm and ant colony algorithm with the development of a sample solution in the Matlab CVRP. These issues were compiled into a book, which is an appendix to this document.

For the purpose of clarity a GAMS tutorial was performed, which demonstrates the contents of the themes developed in the document with the application of different exercises. The MatLab genetic algorithm tool (Genetic Algorithm Tool) is illustrated by implementing an instance of CVRP; which similarly explains the development of each element of the Ant Colony algorithm. The documents are intended to guide readers in managing these programs/tools and their principal functions.

Finally, the book describes the methodology used by the group in working to develop the theme through the modality of a research seminar.

---

* Grade project in the Seminar Research modality

** Physics-Mechanical faculty, School of Industrials Studies and Management. Directed by Henry Lamos Díaz.

INTRODUCCIÓN

Una de los deberes de la Universidad Industrial de Santander y de la Escuela de Estudios Industriales y Empresariales es brindar al estudiantado múltiples formas de desarrollar sus competencias, una de esas formas es el Seminario de Investigación, opción de proyecto de grado que enfoca al estudiante hacia la profundización en temas de ciencia e investigación formando pequeños investigadores.

La metodología con la cual se desarrolla el Seminario de Investigación debe ser clara, llevar una adecuada planeación y organización para mantener el flujo de cada una de las sesiones, de la metodología depende el cumplimiento del alcance y objetivos planteados en el plan del proyecto.

El seminario de Investigación está constituido por cuatro partes. La primera, es la labor investigativa por parte del director del proyecto y los estudiantes. La segunda, corresponde a las relatorías, correlatorías y retroalimentación de los temas indicados para cada sesión. La tercera, corresponde a la recopilación de cada uno de temas para la conformación del documento final. La cuarta y última parte hace referencia a los logros obtenidos durante el seminario junto con las respectivas conclusiones y recomendaciones.

El documento esta ordenado de la siguiente manera: una explicación acerca del desarrollo del seminario, justificación, características y organización, de igual manera, se presenta la metodología que se va a desarrollar. Seguidamente, la selección de temas y subtemas y exploración bibliográfica. Una tercera parte en la que se resumen los temas abordados durante el seminario, posteriormente se encuentran las conclusiones y recomendaciones. La quinta y última parte corresponde a los anexos, logros obtenidos a través del Seminario (libro, algoritmos y manuales de GAMS y MATLAB).

1. SEMINARIO DE INVESTIGACIÓN: MÉTODOS DE SOLUCIÓN PARA EL CVRP

1.1 ¿EN QUÉ CONSISTE EL SEMINARIO DE INVESTIGACIÓN?

A finales del siglo XVIII en una actividad académica en la Universidad de Gottingen de Alemania los investigadores dieron origen al seminario de investigación, con el propósito de sustituir la palabra cátedra y ante todo demostrar que si se puede fusionar la investigación y la docencia.

El seminario de investigación está compuesto por un grupo de personas que se intercomunican exponiendo (la relatoría) un tema específico, complementándolo, evaluándolo (la correlatoría), aportando entre todos (la difusión), para finalmente sacar conclusiones y planear nuevos interrogantes, dejando todo ello por escrito en la memoria escrita (el protocolo)[1].

El Seminario de Investigación se programa por temas los cuales son seleccionados con la orientación del director del seminario, quien con su experiencia y conocimiento del tema central, guía la selección con la debida pertinencia, actualidad y ubicación en el contexto. Los temas son desarrollados en sesiones planificadas, en las cuales, los miembros del grupo deben asumir diferentes roles, de acuerdo con la descripción anterior, manteniendo una relación de interés y compromiso con el conocimiento, sin jerarquías, en un clima de colaboración y participación activa.

Por medio del seminario de investigación los miembros del grupo adquieren capacidad de analizar, manejo de público, buen nivel de interpretación, trabajo en equipo, capacidad de identificar el problema, sintetización, aplicabilidad entre

[1] Carlos M. Vélez S, Resumen El Seminario Investigativo, basado en el Simposio permanente sobre la Universidad 1990-1992 ASCUN.

otras. La idea principal del seminario es comprender que no solo el profesor puede ser aquel que da su opinión, se trata es de convertir la cátedra en un juego de dar y recibir, en donde todos aportan, todos investigan, todos aprenden.

1.2 OBJETIVO DEL SEMINARIO DE INVESTIGACIÓN[2]

Formar a los participantes para la investigación científica mediante el desarrollo de habilidades específicas aplicadas al asumir los diferentes roles dentro del seminario. Dichas habilidades están orientadas a desarrollar la capacidad de lector crítico de resultados de investigación en cualquiera de las áreas del conocimiento, a fortalecer la capacidad de observar e identificar los problemas presentes en tópicos bajo análisis, a buscar respuestas a preguntas claves y sustentarlas teórica y metodológicamente en forma verbal y por escrito, y a identificar las relaciones del problema objeto de estudio con el contexto económico, político o social, a fin de enriquecer con una mirada de integralidad, el conocimiento para el grupo de estudiantes. Para ello se programan y ejecutan ejercicios estructurados que permiten a los estudiantes, desarrollar competencias iniciales de investigador, avanzar en el conocimiento y aportar buenas revisiones y análisis sobre tópicos que pueden facilitar el desarrollo de la investigación.

Para alcanzar dicho objetivo es preciso que haya una formación desde el trabajo personal hacia el trabajo en equipo; para esto, cada participante debe reconocer sus intereses, estilos de aprendizaje, su capacidad para aprender en interacción con pares; debe apropiarse de la metodología e instrumentos con los cuales trabajará, con el fin de lograr, al interactuar con los demás miembros del grupo en las sesiones del seminario, compartir, criticar y corregir las ideas que surjan de él, en un ambiente de la colaboración mutua.

[2] Carlos M. Vélez S, Resumen El Seminario Investigativo, basado en el Simposio permanente sobre la Universidad 1990-1992 ASCUN.

Los seminarios de investigación, no se enfocan hacia la repetición de trabajos ya realizados, sino hacia la búsqueda de respuestas con nuevos argumentos; por tal razón los trabajos que se deriven del cumplimiento del objetivo del Seminario, deben caracterizarse por su originalidad y estar acordes al nivel científico de formación de sus participantes.

1.3 JUSTIFICACIÓN DEL SEMINARIO DE INVESTIGACIÓN

La Universidad Industrial de Santander en su crecimiento vertical ha venido avanzando en la creación de nuevos programas de maestrías en aquellas Escuelas donde aun no los hay. La Escuela de Estudios Industriales y Empresariales presentó ante el Consejo Académico el proyecto de creación de una maestría en Ingeniería Industrial que fue aprobada y el cual fue puesto en marcha a mediados de julio del año 2009. Por medio del seminario de investigación se busca crear un espacio académico para que los estudiantes de pregrado conozcan tópicos avanzados de la programación matemática, fortaleciendo de esta manera los grupos de investigación existentes que tratan sobre problemas de programación matemática.

Por otro lado, muchas decisiones en la administración de operaciones involucran la utilización de modelos que describen ciertos fenómenos que por su naturaleza son altamente complejos. Aunque en el transcurso de la carrera existen dos asignaturas que abarcan temas de la programación matemática: Investigación Operacional I y II; temas complejos y de gran actualidad como el problema de ruteo de vehículos no se alcanzan a discutir.

El seminario de investigación será un espacio propicio para el encuentro y discusión de estos temas de gran actualidad, útiles en la toma de decisiones administrativas, que requieren una fundamentación teórica sólida de las matemáticas aplicadas como lo es la programación matemática que trata de métodos, procedimientos y teorías que se usan para resolver problemas de optimización.

1.4 CARACTERÍSTICAS DEL SEMINARIO DE INVESTIGACIÓN[3]

El seminario de investigación posee las siguientes características:

- Participación activa de todos los miembros del seminario, puesto que no solo el director (profesor) interviene, sino también todos los integrantes del grupo realizan su aporte desde el rol que estén desempeñando. En este proceso los participantes siendo discípulos empiezan a recorrer el camino hacia Maestros.
- El Seminario de Investigación está conformado por un grupo reducido, de aprendizaje activo y cooperativo, inducido a investigar, reflexionar, descubrir y concluir.
- Empleo del dialogo permanente para compartir los conocimientos adquiridos.
- Ambiente amable y cooperativo fomentando la mayor participación de los integrantes del grupo.
- Sesiones desarrolladas utilizando medios didácticos de apoyo al aprendizaje.
- La estructura del seminario y todas las actividades y parámetros para desarrollarlas, son planificados en la primera sesión.

[3] Carlos M. Vélez S, Resumen El Seminario Investigativo, basado en el Simposio permanente sobre la Universidad 1990-1992 ASCUN.

- El seminario de investigación exige a los participantes una alta responsabilidad para lograr la preparación adecuada, que les permita tener bases para llevarlo a cabo.

1.5 ORGANIZACIÓN DEL SEMINARIO DE INVESTIGACIÓN

La dinámica del seminario de investigación gira en torno a las actividades: relatoría, correlatoría, discusión y el protocolo. Para llevar a cabo dichas actividades se requiere previamente haber escogido un tema específico, el cual será preparado con anticipación por cada una de los participantes del grupo (principalmente por el relator), cada miembro de grupo tomará un papel determinado durante la sección que se realizará en un lugar determinado.

El Seminario de Investigación Métodos de Solución para CVRP está compuesto por:

Director: Dr. Henry Lamos Díaz

Estudiantes: Deisy Carolina Cantillo Calderón, Mariluz Espinosa Sandoval, Silvia Adriana Galván Núñez y Margareth Yesenia Ortiz Guzmán

Las sesiones se llevaron a cabo en las diferentes aulas del edificio de la Escuela de Estudios Industriales y Empresariales, los días jueves con duración de dos horas. El número de sesiones se determina de acuerdo al tema a tratar:

1.6 DIRECCIÓN DEL SEMINARIO DE INVESTIGACIÓN

La dirección del seminario de investigación Algunos Métodos de Solución para el CVRP estuvo a cargo del profesor: Henry Lamos Díaz, quien cuenta con la siguiente formación académica:

- Ph. D en Física – Matemática

  Universidad Estatal de Moscú (Lomonosov).

- Magister en Informática

  Universidad Industrial de Santander.

- Magister en Matemáticas

  Universidad de la Amistad (Moscú).

- Matemático

  Universidad de la Amistad (Moscú).

1.7 METODOLOGÍA DEL SEMINARIO DE INVESTIGACIÓN

La metodología fue el eje principal del Seminario de Investigación, ya que, por medio de esta se llevo una adecuada organización y planeación que permitió alcanzar los objetivos propuestos. Para el desarrollo del Seminario de Investigación Algunos Métodos de Solución para el Problema de Ruteo de Vehículos con Capacidad se escogió la metodología ilustrada en el siguiente gráfico:

Figura 1. Representación metodología del seminario de investigación

Fuente: Lineamientos para el seminario de investigación

Vale la pena destacar que la metodología es tan solo una parte del proceso de aprendizaje en el Seminario de Investigación, que comprende los temas objeto de estudio y las técnicas necesarias para su desarrollo. También depende de la forma en que el investigador recolecte, organice y razone de acuerdo a lo estudiado con el fin de enfocar al Seminario a objeto de ciencia o conocimiento científico.

2. PLANEACIÓN

En la planeación se definieron los temas a estudiar con el fin de cumplir el alcance y objetivos planteados durante la primera etapa del Seminario de Investigación, los cuales se presentan en los numerales siguientes.

2.1 ALCANCE

Al finalizar el seminario de investigación se habrá elaborado un documento con los diferentes temas tratados. Este documento se organizará de forma sistemática con el fin de que sirva como texto guía para futuras generaciones de la Escuela de Estudios Industriales y Empresariales que deseen conocer formalmente uno de los temas de gran actualidad de la programación matemática; específicamente, el Problema de Ruteo con capacidad (CVRP Capacited Vehicle Routing Problem). Los métodos (algoritmos) de solución al problema CVRP que se discutirán en el seminario son el método de Branch and Bound) (B&B), Colonia de Hormigas y Algoritmos Genéticos. La escritura de los métodos se hará de manera formal y con ejemplos académicos usando software adecuado en cada uno de ellos. El documento consta de varios capítulos:

Capitulo 1. Modelación matemática. Modelos de decisión. Problemas combinatorios. Clasificación del Problema de Ruteo. Complejidad computacional.

Capitulo 2. Programación Entera. Métodos de solución. Técnica de ramificación y acotamiento (B&B, Branch and Bound) para la solución del problema CVRP. Introducción al lenguaje de modelado. Lenguaje de modelado GAMS. Solucionador CPLEX. Aplicación del método B&B para la solución del problema CVRP.

Capitulo 3. Introducción a los métodos heurísticos para la solución de problema CVRP. Métodos metaheurísticos. Colonia de Hormigas. Algoritmos Genéticos

(AG). Uso del software Matlab y el toolbox de AG para la solución de problemas combinatorios. Aplicación de los metaheurísticos para la solución del problema CVRP.

Capitulo 4. Conclusiones y recomendaciones.

2.2 OBJETIVOS

2.2.1 Objetivo General

Estudiar y discutir de manera formal el problema de ruteo de vehículos con capacidad (CVRP).

2.2.2 Objetivos específicos

- Recopilar los principales resultados obtenidos en investigaciones previas en temas relacionados con programación matemática y problemas de ruteo de vehículos.
- Presentar las definiciones, teoremas y algoritmos de la optimización combinatoria de forma sencilla para la comprensión.
- Estudiar y presentar el método de *Branch and Bound* para la solución del problema de ruteo de vehículos.
- Estudiar y presentar las técnicas metaheurísticas: *Colonia de hormigas* y *Algoritmos genéticos* para resolver el problema de ruteo de vehículos**.**
- Utilizar el lenguaje de modelado GAMS y el optimizador CPLEX como herramienta en la solución de los problemas.
- Usar el toolbox de matlab AG para la solución del problema CVRP.

- Usar el software Matlab para la implementación de ciertas rutinas del método de colonias de hormigas.

- Elaborar una guía que compile la información obtenida en el transcurso del seminario.

2.3 SELECCIÓN DEL TEMA

Los temas se seleccionaron teniendo en cuenta el conocimiento de los mismos por parte del Director de Proyecto en común acuerdo con los integrantes del seminario, de igual manera, participando en el Semillero de Investigación y haciendo un estudio previo sobre los temas con el fin de obtener la información necesaria.

Durante el Seminario de Investigación se estudiaron tres métodos de solución para el Problema de Ruteo de Vehículos con Capacidad, los cuales corresponden a un método exacto y dos meta-heurísticas. Se abordó el problema desde los tres métodos, presentando la metodología para la respectiva solución y por último, se ilustró por medio de ejemplos. Los temas seleccionados se presentan en la tabla 1.

Tabla 1. Temas principales del Seminario de Investigación.

CAPÍTULO	TEMA
PROBLEMA DE RUTEO DE VEHÍCULOS	MODELOS MATEMÁTICOS
MÉTODOS DE SOLUCIÓN EXACTOS PARA EL CVRP.	BRANCH AND BOUND
MÉTODOS DE SOLUCIÓN METAHEURÍSTICOS PARA EL CVRP.	ALGORITMOS GENÉTICOS COLONIA DE HORMIGAS

2.4 ESTUDIO BIBLIOGRÁFICO

La bibliografía utilizada para la investigación durante el Seminario de Investigación fue debidamente seleccionada y estudiada por cada uno de los integrantes y Director del Proyecto. Sin embargo, se recurrió a nuevas fuentes con el fin de profundizar los conceptos, formulaciones y metodologías de cada una de las técnicas de solución para el CVRP. En seguida se presenta la bibliografía principal aborda durante el Seminario haciendo la respectiva división en libros y papers.

Libros:

- Dorigo M., STÜTZLE T. Ant Colony Optimization (ACO). Massachusetts Institute of Technology. 2004. Este libro introduce rápidamente al campo de la optimización de colonia de hormigas. Da una apreciación global de aspectos de ACO, lo define y describe cómo el ACO generalmente puede aplicarse a una amplia gama de problemas de optimización combinatoria.

- Engelbrecht A. Computational Intelligence. University of Pretoria. 2 Texto que proporciona una apreciación global del ACO-MH, con especial atención en los principios básicos de algoritmos de ACO y los primeros algoritmos que se han desarrollado.

- Goldberg D. Genetic Algorithms in Search, Optimization, and Machine Learning. The University of Alabama. Ed. Addison-Wesley.: Texto en el que se expone la teoría, funcionamiento y aplicación de los algoritmos genéticos basados en la teoría de selección de Darwin y la genética.

- Taha, Hamdy A. Investigación de Operaciones.7 Edición. Pearson Educación. México. 2004.: Libro base para la ampliación de los conceptos aprendidos en Investigación de Operaciones, como el Problema Dual, Método del Plano de Corte y Branch and Bound.

- Toth P., Vigo D. The Vehicle Routing Problem. Siam. 2002.: Este libro fue la guía para el desarrollo del Seminario de Investigación, ya que, hace una presentación del problema de ruteo de vehículos, muestra las diferentes formulaciones matemáticas y da una introducción acerca de cada uno de los métodos de solución estudiados para la solución de VRP.

Papers:

- Athanasios M., ·Panos M., Yannis M. A new bilevel formulation for the vehicle routing problem and a solution method using a genetic algorithm. Springer Science. Business Media B.V. 2006.: Este documento se presentan fundamentos teóricos para la formulación del problema de ruteo de vehículos por la heurística Algoritmos Genéticos, se trata de una formulación en dos niveles para el CVRP, en el primer nivel se formula el CVRP como un problema de asignación y en segundo nivel se optimiza la ruta que debe seguir cada vehículo cumpliendo con las restricciones del problema.

- Birattari M. Ant Colony optimitation. Este artículo hace una introducción a la optimización de colonia de hormigas y resume las aplicaciones más notables, sus variantes principales, muestra algunos resultados teóricos notables e ilustra algunas de sus aplicaciones más exitosas.

- Perboli G., Pezzella F., Tadeia R. Hybrid algorithm for the Capacitated Vehicle Routing Problem. Springer-Verlag 2008.: En el escrito mencionado muestra un Algoritmo híbrido que emplea las heurísticas Búsqueda Tabú y Algoritmos Genéticos para la solución de CVRP, de igual manera, presenta conceptos teóricos fundamentales para dicha aplicación.

- Toth P., Vigo D. A branch and Bound algorithms for capacitated vehicle routing problem on directed graphs. Operations Research. 1994. Este paper un complemento del libro elaborado por los mismos autores, ya que, presenta detalladamente dos métodos de acotación basados en additive aproach para el CVRP y se comparan los resultados con los obtenidos en la literatura.

- Laporte G., Nobert Y. A Branch and Bound algorithms for the capacitated vehicle routing problem. Springer-Verlag. 1983. Este documento explica menudamente la formulación matemática del CVRP, expone los pasos del algoritmo de solución, resuelve tres (3) variantes diferentes y hace la respectiva presentación y comparación de resultados.

2.5 SELECCIÓN DE SUBTEMAS

Los temas del seminario de dividen en subtemas como se especifica en la tabla 2 en el orden en que fueron tratados y teniendo en cuenta el alcance y objetivos planteados en el seminario.

Tabla 2. Selección de subtemas

TEMA	SUBTEMA	
Capítulo 1 Problema de Ruteo de Vehículos	Problemas de Optimización.	
	Variantes del VRP.	
	Formulación matemática.	
Capítulo 2: Métodos de solución exactos para el CVRP	Método de Branch and Bound.	
	Relajaciones Básicas aplicadas al CVRP.	
	Relajaciones avanzadas aplicadas al CVRP.	
	Planteamiento de un programa lineal entero por Relajación de Lagrange	
	Modelado en GAMS/CPLEX	
Capítulo 3: Métodos solución metaheurísticos para el CVRP.	Algoritmos genéticos	Algoritmos Genéticos.
		Operadores Genéticos.
		Funcionamiento de los Algoritmos Genéticos.
		Aplicación de los Algoritmos Genéticos al CVRP.
		Ejemplo.
		Modelado en Matlab.
	Colonia de Hormigas	Optimización Colonia de Hormigas.
		Aplicación de la Optimización con Colonia de Hormigas al CVRP.
		Descripción del algoritmo Colonia de Hormigas.
		Modelado en Matlab y Ejemplo.

2.5.1 Planificación de las sesiones

Las sesiones se desarrollaron en el edificio de ingeniería industrial los días jueves con una duración de dos horas, en el horario de 4:00 pm a 6:00 pm llevándose a cabo la programación presentada en la tabla 3 en la que se especifica el subtema tratado, el número de sesiones dedicado al subtema, el tema asignado para la siguiente sesión, relator, correlator y fecha de ejecución.

Tabla 3. Planificación de las sesiones.

SUBTEMA	RELATOR
Problemas de Optimización.	Margareth Ortiz, Silvia Galván
Variantes del VRP.	Deisy Cantillo, Mariluz Espinosa
Método de Branch and Bound.	Silvia Galván
Relajaciones básicas aplicadas al CVRP.	Margareth Ortiz, Deisy Cantillo
Relajaciones avanzadas aplicadas al CVRP.	Silvia Galván, Mariluz Espinosa
Planteamiento de un programa lineal entero por Relajación de Lagrange	Mariluz Espinosa, Margareth Ortiz
Modelado en GAMS/CPLEX	Deisy Catillo, Silvia Galván
Algoritmos genéticos	Mariluz Espinosa, Margareth Ortiz
Algoritmo Colonia de Hormigas	Deisy Cantillo, Silvia Galván

3. EJECUCIÓN

Esta etapa se llevó a cabo siguiendo las especificaciones dadas en la Planeación, abarcando los temas y subtemas descritos.

El relator elaboraba un documento cumpliendo con todas las normas científicas y técnicas, soportado con una bibliografía definida. Este documento era entregado al director del proyecto y a los demás integrantes para ser revisado, estudiado y corregido, de esta manera, el día de la relatoría, todos se encontraban debidamente preparados para participar y aportar en el desarrollo de la sesión.

Las sesiones siguieron las pautas del protocolo, el cual, consta de los siguientes pasos:

ACTIVIDAD	DESCRIPCIÓN
Apertura de la sesión	• Lectura del plan de trabajo. • Verificación de la asistencia. • Confirmación o asignación de roles. • Asignación de los roles para la próxima sesión.
Lectura del protocolo	• Lectura del protocolo. • Se abre una sesión de preguntas aclaratorias del texto del protocolo. Los ajustes quedaran consignados en el protocolo siguiente.
Relatoría	• Exposición del tema. • Entrega el trabajo escrito.
Correlatoría	• Se complementa y evalúa la relatoría. • Se induce a la discusión.
Discusión	• Se realizan preguntas, intervenciones y aclaraciones sobre el tema. • Se valida el conocimiento demostrado por el relator.

Conclusión	• Evaluación de la sesión • Síntesis del trabajo • Redacción del acta, en la cual, se deja por escrito, la fecha, el lugar, los asistentes, el orden del día y los compromisos asignados para la siguiente sesión. • Aprobación de la síntesis y el protocolo

Las actas se encuentran relacionadas en el Anexo A y los documentos resultado de cada relatoría son los que conforman el Libro Introducción al CVRP.

4. FINALIZACIÓN

El seminario de investigación culmina con la realización de los documentos guía para los interesados en el tema de ruteo de vehículo, así como se propuso en el plan de proyecto. Los documentos resultados del seminario de investigación fueron: libro Ruteo de Vehículos con Capacidad CVRP y el Manual de GAMS aplicado al ruteo de vehículos (véase Anexo B y Anexo C).

El Libro Ruteo de Vehículos con Capacidad CVRP es la recopilación de todos los temas estudiados, fue elaborado en el editor LyX, que es un procesador avanzado de documentos de código abierto que funciona en Linux/Unix, Windows y Mac OS X. Se llama "procesador de documentos" porque, a diferencia de los procesadores de texto habituales, LyX fomenta para la escritura un enfoque basado en la estructura del documento, no en su apariencia.

LyX permite concentrarse en la escritura, dejando para el software los detalles del diseño visual. LyX automatiza el formato de acuerdo con un conjunto de reglas predefinidas, brindando consistencia completa incluso en los más complejos documentos. LyX genera salida de alta calidad, profesional -usando en segundo plano LaTeX, un motor de potencia industrial para tipografía, de código abierto[4].

El manual de GAMS fue desarrollado en Microsoft Word teniendo como soporte lo realizado durante el tema del seminario: Branch and Bound, ya que a medida que se realizaban las pruebas para los diferentes ejercicios, se redactaba el manual resaltando indicaciones importantes para el manejo de GAMS.

Así como se indica en los objetivos específicos, la herramienta de solución para las instancias de CVRP mediante las metaheurísticas Algoritmos Genéticos y Optimización Colonia de Hormigas fue Matlab. Matlab es un lengua utilizados para Matemática y computación, desarrollo de algoritmos, modelamiento y simulado,

[4] http://www.lyx.org/WebEs.WhatIsLyX

entre otras. Tiene su propio lenguaje de programación. Es un software altamente usado en las universidades debido a su gran capacidad y facilidad para dar respuesta.

CONCLUSIONES

- El Seminario de Investigación es una herramienta que contribuye a la formación académica de los estudiantes para la adquisición de conocimientos en temas de punta en la investigación.

- En el seminario se estudiaron tres técnicas diferentes para la solución del Problema de Ruteo de Vehículos con Capacidad, un método exacto y dos metaheurísticas. Los métodos exactos tienen una teoría de gran belleza matemática, sin embargo, por ahora no son eficientes para la solución de problemas con un número moderadamente grande de instancias, debido al tiempo computacional requerido para encontrar la solución óptima. Por otro lado, las metaheurísticas tienen por objetivo buscar óptimos globales en un tiempo razonable utilizando operadores que permitan el movimiento dentro del espacio factible y componentes randómicos para evitar caer en óptimos locales. Su complejidad radica en la construcción de dichos operadores y de la función objetivo, que dependen del problema a solucionar.

- Se estudió el método exacto Branch and Bound usando diferentes métodos de relajación para el CVRP. Se estudiaron las relajaciones básicas que permiten hallar cotas inferiores de la solución pero que tienen un gap bastante grande. Las relajaciones avanzadas son técnicas de relajación que encuentran mejores cotas inferiores, lo que permite llegar más rápido a la solución disminuyendo el tiempo de ejecución.

- Si bien los Algoritmos genéticos no fueron desarrollados inicialmente como técnicas de optimización, se convierten en una buena estrategia para la solución del CVRP porque parten de una solución inicial que cumple con todas las restricciones del problema ocasionando disminución en el tiempo de ejecución, del uso correcto de los operadores y de la habilidad del modelador depende el éxito del algoritmo.

- El algoritmo por colonia de hormigas fue creado para solucionar especialmente problemas combinatorios, específicamente el TSP, facilitando su implementación para la solución del problema CVRP. Los movimientos en el espacio factible se realizan mediante la regla de transición o probabilidad de transición, que permite únicamente la transición a un elemento del conjunto especial de nodos (factibles), donde el nodo que presenta mayor probabilidad no necesariamente es aquel que contiene la mejor decisión, es por esto que se utiliza la función selección por ruleta, que genera los nodos aleatoriamente de acuerdo a dicha probabilidad, permitiendo que todos los nodos tengan la probabilidad de ser escogidos, evitando de esta manera caer en mínimos locales.

- El trabajo en grupo permitió enriquecer el conocimiento en temas de investigación, orientar a los demás estudiantes sobre los temas desarrollados y vinculación al Grupo de Investigación. De igual manera, el grupo aporta un libro con los temas estudiados para que quienes hagan parte del Grupo de Investigación OPALO (Optimización en sistemas Productivos, Administrativos y Logísticos) puedan avanzar en el estudio del tema y presentar mejores resultados.

RECOMENDACIONES

- El campo en el cual se encuentra el Problema de Ruteo de Vehículos es demasiado amplio, cada organización tiene una estructura de transporte y clientes que deben servir con diferentes características, por lo que, se recomienda seguir explorando los diferentes métodos de solución y variantes con el fin de encontrar la técnica adecuada para la situación real que se presente.

- La eficiencia de los algoritmos puede ser probada con instancias grandes y comparada con los datos obtenidos en la literatura, es un tema que queda para quienes deseen continuar con la investigación en problemas de ruteo.

- El seminario de investigación es una opción de proyecto de grado interesante que no se puede dejar olvidado, lo mismo que la continuidad en los temas, el éxito radica en el compromiso de cada uno de los integrantes, egresados, profesores y directivos de la institución.

- Se recomienda a los estudiantes empezar desde muy temprano el interés por la investigación, el aprendizaje de los diferentes programas computacionales de optimización y el uso adecuado de las bases de datos con las que cuenta la Universidad.

- Para casos específicos de aplicación, se pueden determinar ventajas y desventajas obtenidas como consecuencia de la utilización del método exacto y las técnicas metaheurísticas estudiadas, soportadas con resultados de diseños experimentales y tiempos de ejecución.

BIBLIOGRAFÍA

- Athanasios M., ·Panos M., Yannis M. A new bilevel formulation for the vehicle routing problem and a solution method using a genetic algorithm. Springer Science. Business Media B.V. 2006.
- Carlos M. Vélez S, Resumen El Seminario Investigativo, basado en el Simposio permanente sobre la Universidad 1990-1992 ASCUN.
- Dorigo M., STÜTZLE T. Ant Colony Optimization. Massachusetts Institute of Technology. 2004.
- Engelbrecht A. Computational Intelligence. University of Pretoria.
- Goldberg D. Genetic Algorithms in Search, Optimization, and Machine Learning. The University of Alabama. Ed. Addison-Wesley.
- Laporte G., Nobert Y. A Branch and Bound algorithms for the capacitated vehicle routing problem. Springer-Verlag. 1983.
- Néstor H. Bravo Salinas, El seminario Investigativo – El seminario como práctica pedagógica para la formación integral.
- Perboli G., Pezzella F., Tadeia R. Hybrid algorithm for the Capacitated Vehicle Routing Problem. Springer-Verlag 2008.
- Taha, Hamdy A. Investigación de Operaciones.7 Edición. Pearson Educación. México. 2004.
- Toth P., Vigo D. A branch and Bound algorithms for capacitated vehicle routing problem on directed graphs. Operations Research. 1994.
- Toth P., Vigo D. The Vehicle Routing Problem. Siam. 2002.: Este libro fue la guía para el desarrollo del Seminario de Investigación.
- http://www.uv.es/eees/archivos/sem_inves.

ANEXOS

ANEXO A

INTRODUCCIÓN AL CVRP

ANEXO B

MANUAL GAMS

UNIVERSIDAD INDUSTRIAL DE SANTANDER

GENERAL ALGEBRAIC MODELING SISTEM (GAMS)

Autores

Deisy Carolina Cantillo Calderón
Mariluz Espinosa Sandoval
Silvia Adriana Galván Núñez
Margareth Ortiz Guzmán

2009

Director

Henry Lamos Díaz

BUCARAMANGA

TABLA DE CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE ILUSTRACIONES

Pág.

INTRODUCCIÓN A GAMS

Desde los años 50s se ha visto un rápido desarrollo de algoritmos y códigos computacionales para analizar y resolver un gran número de problemas de programación matemática. Un importante avance fue el desarrollo en los 80s de GAMS (General Algebraic Modeling Sistem), un sistema de modelado de alto nivel, análisis y resolución de problemas de programación matemática. La representación del modelo en GAMS se debe hacer de tal forma que pueda ser entendido por los lectores y los computadores al mismo tiempo, siendo esto uno de los aspectos más importantes.

Entre las principales ventajas de GAMS cabe destacar:

- Representa modelos complejos empleando un lenguaje algebraico de alto nivel y de manera concisa.
- Permite hacer cambios de forma simple y segura en los diferentes modelos.
- Proporciona un entorno en donde fácilmente se puede revisar y comprobar un modelo con pocos subíndices para luego ejecutar un modelo más complejo.
- La documentación permite el uso de un gran número de variables, ecuaciones y de igual forma nombrar los subíndices, escribir comentarios y definir datos.
- Permite encontrar fácilmente los errores y corregirlos de forma adecuada.
- GAMS es una interface que permite trabajar con diferentes solvers. El uso depende del tipo de problema a modelar.
- El proceso de modelado incluye: cálculo de datos, verificar si la declaración del modelo está escrita de manera exacta, chequear la formulación para corregir defectos obvios, permite la interfaz con un solver y el uso de la solución para redacción de informes.
- Facilita importar y exportar datos desde y hacia otros paquetes de software.
- Proporciona ejemplos de modelos de la librería (model library) como guía para el modelador en el desarrollo de nuevos modelos.

Todos los usuarios de GAMS deben tener especial cuidado con las reglas gramaticales ya que con solo incumplir una de ellas pueden aparecer una gran cantidad de errores o un modelo con resultados equivocados. Para un buen aprendizaje de la herramienta GAMS, se recomienda el manual del usuario de autoría de Richard E. Rosenthal y una versión más compleja y detallada de este tutorial, autoría de Bruce A. McCarl, profesor de la Universidad de Texas.

1 TUTORIAL

A continuación se presenta una rápida y sencilla explicación acerca de GAMS para que el lector conozca las bondades y ventajas del software, al igual que, se familiarice con el programa y el lenguaje.

1.1 MODELO BÁSICO

Existen diferentes tipos de problemas que se pueden modelar en GAMS, comprenden desde la programación lineal, entera, entera mixta, no lineal, cuadrática, problemas de optimización económica, entre otros. Para los problemas mencionados anteriormente existen algunos elementos del lenguaje GAMS que son comunes para todos y otros que dependen de los requerimientos del problema a modelar. A continuación se presenta un problema de programación lineal entera mixta (MIP) que servirá como ejemplo para familiarizarse con el lenguaje, ya que, es un problema que presenta una estructura simple y básica, como también, muestra las ventajas de GAMS como herramienta de modelado.

El problema consiste en minimizar el costo total de transporte decidiendo sobre la ubicación de **k** plantas, cada una con capacidad $\mathbf{a_k}$ en **i** alternativas de localización, para suplir la demanda $\mathbf{b_j}$, generada por cada uno de los **j** clientes a donde se deben transportar los productos. Además, se dan los costos de transportar la mercancía de la planta **i** al sitio **j**.

A continuación se muestra la representación algebraica de este problema:

Índices:

k= plantas a ubicar

i= alternativas de ubicación

j = clientes

Datos suministrados:

a_k = capacidad de cada planta k

b_j = demanda de cada cliente j

cij = costo de transporte desde las alternativas de localización i hasta los clientes j ($/U).

Variables de decisión:

x_{ij} = cantidad de productos a despachar desde el sitio i al cliente j, toma valores enteros y positivos.

Y_{ki} =variable de decisión, toma el valor de 1 si la planta k se ubica en el sitio j y 0 en caso contrario.

Restricciones:

Asociadas a la capacidad requerida en cada elemento: $\sum_j X_{ij} \leq \sum_k a_k * Y_{ki}$

Asociadas a la demanda: $\sum_i X_{ij} \geq b_j$

Asociadas al número de elementos posibles por alternativa de localización: $\sum_j Y_{ki} \leq 1$

Función a optimizar:

Función objetivo: $Min = \sum_i \sum_j X_{ij} * C_{ij}$

1.2 EXPLICACIÓN DEL LENGUAJE GAMS

Los comandos obligatorios del lenguaje GAMS son: variables (**VARIABLES**), ecuaciones (**EQUATIONS**), modelo (**MODEL**), solución (**SOLVE**), mientras que, los opcionales, es decir, los que dependen de las características del modelo y los requerimientos del programador son: conjuntos (**SET**), datos (**DATA**) y visualización (**DISPLAY**). Vamos a revisar cada uno de estos elementos.

Variables: Como en cualquier problema de investigación de operaciones, GAMS requiere que tanto las variables del problema como la función a optimizar sean identificadas. Esto se hace con el comando **VARIABLE** o **VARIABLES** como se ve a continuación.

```
VARIABLES           fo,  x(i,j),  y(k,i);
INTEGER VARIABLES x;
BINARY VARIABLES    y;
```

En GAMS también es necesario especificar los diferentes tipos de variables, la tabla 1 muestra el tipo de variable y la respectiva explicación.

Tabla 2. Tipos de variables.

TIPO DE VARIABLE	EXPLICACIÓN
VARIABLE	No existe restricción, el valor de la variable puede ir desde - ∞ hasta + ∞.
FREE VARIABLE	Idéntico al caso anterior.
POSITIVE VARIABLE	Solo valores positivos, variables con un rango de valores desde cero hasta + ∞.
NONNEGATIVE VARIABLE	Igual al caso anterior.
NEGATIVE VARIABLE	Valores con un rango desde - ∞ hasta cero, solo valores negativos.
BINARY VARIABLE	Las variables solo pueden tomar los valores de cero o uno.
INTEGER VARIABLE	Solo valores enteros, por defecto el rango es de 0 a 100

FUENTE: McCarl GAMS User Guide

Equations: La especificación de las ecuaciones en GAMS se divide en dos partes. En la primera se da el nombre a cada una de las ecuaciones y en la segunda se escribe la formulación matemática exacta. La notación que se emplea es la siguiente:

```
EQUATIONS

    OBJ  Minimizar el costo total asociado con el transporte
    CAPACIDAD(i)    Asociada a la capacidad
    DEMANDA(j)      Asociada a la demanda
    ALTERNATIVA(k)  Asociada a la alternativa de ubicación;

OBJ..              fo =E= SUM((i,j), C(i,j)*x(i,j));
CAPACIDAD(i)..     SUM(j,x(i,j)) =L= SUM(k,a(k)*y(k,i));
DEMANDA(j)..       SUM(i,x(i,j)) =G= b(j);
ALTERNATIVA(k)..   SUM(i,y(k,i)) =E= 1;
```

La sintaxis para definir si la ecuación es de igual o desigualdad es: `(=E=)` indica una relación de igualdad, `(=L=)` indica una relación de menor o igual y `(=G=)` indica una relación de mayor o igual.

Model: este comando identifica el modelo que será resuelto y se puede escribir de dos formas, escribiendo cada una de las ecuaciones a resolver o simplemente incluir todas la ecuaciones empleando la palabra `/ALL/`, como se muestra enseguida:

```
MODEL Localizacion /ALL/; ó
MODEL Localizacion /OBJ, CAPACIDAD, DEMANDA, ALTERNATIVA/;
```

Solve: El comando SOLVE hace que GAMS utilice un solver para optimizar el modelo resolviendo el sistema de ecuaciones. Para el ejemplo de este manual la notación es la siguiente:

```
SOLVE Localizacion USING MIP MINIMIZING fo;
```

Algunos de los tipos de modelos más conocidos son: **LP** para Programación Lineal, **MLP** para Programación No Lineal, **MIP** para Programación Entera Mixta y **MINLP** para Programación No Lineal Entera Mixta.

Set: Es el conjunto de índices a los cuales se les puede asignar un valor mediante un parámetro, una tabla o un escalar. Este comando se ejecuta escribiendo la palabra set seguida del nombre del conjunto, la descripción del conjunto (opcional) y los elementos que lo conforman (máximo 63 caracteres).

```
SETS i Posible ubicación /BOGOTA, BMANGA, CALI, CARACAS/
     j Clientes /BOGOTA, BRASILIA,MEXICO,CARACAS,LIMA/
     k plantas a ubicar /PLANTA1, PLANTA2/;
```

Subset: Se pueden definir subconjuntos que contengan parte de los elementos de otro conjunto. El formato es el siguiente:

```
SET ii(i) subconjunto de i /BOGOTA, BMANGA/;
```

Donde, `ii` es el nombre del subconjunto y
`i` es el conjunto original.

Alias: El comando alias permite usar otros nombres para el mismo conjunto, por ejemplo llamar el mismo conjunto i, r. La forma general de un alias es:

```
ALIAS (I,R);
```

Data: GAMS proporciona tres formas de proporcionar los datos de entrada, tablas (**TABLE**), parámetros (**PARAMETER**) y escalares (**SCALARS**) y se usan para definir una serie de datos fijos dentro del modelo.

- **Table**: Se emplea para entradas matriciales que involucran dos a más conjuntos y darles valores. El formato es el siguiente:

```
TABLE C(i,j) Costos de transporte
             BOGOTA    BRASILIA   MEXICO    CARACAS   LIMA
BOGOTA        0        140        170        90       100

BMANGA       50        150        160        70       110

CALI         60        120        180       100       90
```

```
CARACAS      90       130       150        0      80;
```

❖ **Parameter**: El comando se usa para dar nombre a los vectores y asignarles valores.

```
PARAMETER a(k) Capacidad de las plantas
          /PLANTA1  200000
           PLANTA2  135000/;
```

❖ **Scalar**: Es para entradas escalares, en el comando se especifica el nombre y el valor del escalar. Por ejemplo:

```
SCALAR CAP CAPACIDAD /40/;
```

Display: Este comando permite mostrar la salida de datos y formato que se desea para el problema. El atributo .l indica el nivel de la variable, para este caso, muestra el valor de las variables x y y que pertenecen a la solución óptima.

```
DISPLAY x.l, y.l;
```

1.3 PROBLEMA DE TRANSPORTE MODELADO EN GAMS

El modelo utilizado para ejemplificar los comandos se presenta a continuación de tal forma que le ayude al lector a entender la estructura de GAMS.

```
* Modelo de localización

SETS   i Posible ubicación /BOGOTA, BMANGA, CALI, CARACAS/
       j Clientes /BOGOTA, BRASILIA,MEXICO,CARACAS,LIMA/
       k plantas a ubicar /PLANTA1, PLANTA2/;

TABLE C(i,j) Costos de transporte

            BOGOTA   BRASILIA  MEXICO   CARACAS  LIMA

BOGOTA        0        140       170      90      100
```

```
BMANGA      50      150      160      70      110

CALI        60      120      180      100     90

CARACAS     90      130      150       0      80;

VARIABLES            fo,  x(i,j),  y(k,i);
POSITIVE VARIABLES x;
BINARY VARIABLES     y;

PARAMETER b(j) Demanda de los clientes
         /BOGOTA       70000
          BRASILIA     85000
          MEXICO       100000
          CARACAS      20000
          LIMA         60000/;

PARAMETER a(k) Capacidad de las plantas
          /PLANTA1   200000
           PLANTA2   135000/;

EQUATIONS
       OBJ  Minimizar el costo total asociado con el transporte
       CAPACIDAD(i)    Asociada a la capacidad
       DEMANDA(j)      Asociada a la demanda
       ALTERNATIVA(k) Asociada a la alternativa de ubicación;

OBJ..              fo =E= SUM((i,j), C(i,j)*x(i,j));
CAPACIDAD(i)..     SUM(j,x(i,j)) =L= SUM(k,a(k)*y(k,i));
DEMANDA(j)..       SUM(i,x(i,j)) =G= b(j);
ALTERNATIVA(k).. SUM(i,y(k,i)) =E= 1;

MODEL Localizacion /ALL/;
SOLVE Localizacion USING MIP MINIMIZING fo;
DISPLAY x.l, y.l;
```

Como se puede ver en el ejemplo anterior, al terminar cada comando se usa un punto y coma (;), el cual especifica que las instrucciones para cada comando han terminado. Al omitir el signo, GAMS no ejecutará el modelo y mostrará que se ha cometido un error.

1.4 EJECUCIÓN DEL MODELO

GAMS es un programa que está compuesto de dos partes. El primero es un editor que contiene instrucciones de GAMS y crea un archivo con extensión .gms que corresponde al formato presentado en el numeral 1.3. Después de ejecutar el modelo, un archivo con extensión .lst muestra la compilación, el listado de ecuaciones, variables, las estadísticas, resumen de la solución del modelo y reporte resumen.

Compilación: transforma las instrucciones originales del modelo en código legible para el solver. Mientras no haya errores, la primera parte que aparece es una copia del modelo .gms precedido de un número que ayuda a identificar con facilidad si la escritura se presenta algún error. Como no existen errores en el ejemplo, la salida es la siguiente:

```
 4 *Modelo de Localizacion
 6
 7 SETS i Posible ubicación /BOGOTA, BMANGA, CALI, CARACAS/
 8      j clientes /BOGOTA, BRASILIA, MEXICO, CARACAS, LIMA/
 9      k plantas a ubicar /PLANTA1, PLANTA2/;
10
11
12  TABLE C(i,j) Costos de transporte
13             BOGOTA   BRASILIA  MEXICO   CARACAS  LIMA
14  BOGOTA     0        140       170       90      100
15  BMANGA     50       150       160       70      110
16  CALI       60       120       180       100     90
17  CARACAS    90       130       150       0       80;
18
19  VARIABLES
20  fo, x(i,j), y(k,i);
```

```
POSITIVE VARIABLES x;
BINARY VARIABLES y;

PARAMETER b(j) Demenda de los clientes
         /BOGOTA      70000
          BRASILIA    85000
          MEXICO      100000
          CARACAS     20000
          LIMA        60000/;

PARAMETER a(k) Capacidad de las plantas
          /PLANTA1  200000
           PLANTA2  135000/;

EQUATIONS
       OBJ  Minimizar el costo total asociado con el transporte
       CAPACIDAD(i)  Asociada a la capacidad
       DEMANDA(j)    Asociada a la demanda
       ALTERNATIVA(k) Asociada a la alternativa de ubicación;

OBJ..            fo =E= SUM((i,j), C(i,j)*x(i,j));
CAPACIDAD(i)..   SUM(j,x(i,j)) =L= SUM(k, a(k)*y(k,i));
DEMANDA(j)..     SUM(i,x(i,j)) =G= b(j);
ALTERNATIVA(k).. SUM(i,y(k,i)) =E= 1;

MODEL Localizacion /ALL/;
SOLVE Localizacion USING MIP MINIMIZING fo;
DISPLAY x.l, y.l;
```

Lista de ecuaciones: en esta segunda parte aparecen las ecuaciones del modelo que se han escrito y se podrá verificar si el programa las asimilo correctamente. Por defecto la salida muestra un máximo de 3 ecuaciones por cada ecuación genérica definida en el modelo. Para mostrar todas las ecuaciones es necesario escribir el comando (option limrow = r) donde r es el número deseado de filas o ecuaciones específicas. A continuación se muestra la restricción de demanda ampliada para las tres primeras ciudades.

```
---- DEMANDA  =G=  Asociada a la demanda

DEMANDA(BOGOTA)..        x(BOGOTA,BOGOTA)  +  x(BMANGA,BOGOTA)  +
x(CALI,BOGOTA)     + x(CARACAS,BOGOTA) =G= 70000 ; (LHS = 0, INFES
= 70000 ****)

DEMANDA(CARACAS)..   x(BOGOTA,CARACAS)   +   x(BMANGA,CARACAS)   +
x(CALI,CARACAS) + x(CARACAS,CARACAS) =G= 20000 ; (LHS = 0, INFES =
20000 ****)

DEMANDA(BRASILIA)..  x(BOGOTA,BRASILIA)  +  x(BMANGA,BRASILIA)  +
x(CALI,BRASILIA) + x(CARACAS,BRASILIA) =G= 85000 ; (LHS = 0, INFES
= 85000 ****)

REMAINING 2 ENTRIES SKIPPED
```

Lista de variables: permite observar las variables del modelo. En GAMS aparecen por defecto tres variables específicas por cada variable genérica. La opción para ver todas las variables es (option limcol = c), donde c es el número deseado de columnas. A continuación se muestra la salida de la varibles x:

```
x(BOGOTA,BOGOTA)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
        1       CAPACIDAD(BOGOTA)
        1   DEMANDA(BOGOTA)

x(BOGOTA,CARACAS)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
      -90       OBJ
        1       CAPACIDAD(BOGOTA)
        1       DEMANDA(CARACAS)
```

```
x(BOGOTA,BRASILIA)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
     -140       OBJ
        1       CAPACIDAD(BOGOTA)
        1       DEMANDA(BRASILIA)

REMAINING 17 ENTRIES SKIPPED
```

Hacer limrow y limcol igual a cero es una buena opción para simplificar la salida de datos. La explicación más detallada del comando option se muestra en la tabla 2.

Tabla 3. Explicación del comando option.

COMANDO	EXPLICACIÓN
OPTION OPTCR `OPTION OPTCR=0.1;`	OPTCR (OPTimality CRiterion), Garantiza que la solución que se va a encontrar sea óptima, el valor por defecto es 0,10. Sin embargo, para problemas complejos es necesario disminuir la tolerancia.
OPTION LIMCOL `OPTION LIMCOL=3;`	En el fichero LST el comando proporciona el listado de las variables (columns). El valor por defecto es 3.
OPTION LIMROW `OPTION LIMROW=3;`	Permite ver en el fichero LST el listado de ecuaciones (rows). El valor por defecto es 3.
OPTION ITERLIM `OPTION ITERLIM=10000;`	Especifica el máximo de iteraciones que puede realizar GAMS antes de detenerse. El valor por defecto es 10000.
OPTION RESLIM `OPTION RESLIM=1000;`	Controla el máximo tiempo en segundos usado por el sistema. Una vez superado el límite el programa se detiene. El valor por defecto es 1000. Sin embargo, se puede cambiar este valor admitiendo cualquier número real positivo.
OPTION SOLPRINT `OPTION SOLPRINT=ON;`	Controla la salida de la solución del modelo en el archivo LST. Las opciones son ON, OFF y SILENT. Sin embargo, la opción por defecto es ON. La opción ON imprime la solución de una fila o una columna en cada fila, OFF no imprime nada y

	SILENT suprime toda la información de la solución.

FUENTE: Autoras del manual.

Estadísticas del modelo: GAMS arroja un grupo de estadísticas que muestra el número de ecuaciones y variables presentes en la modelo. Para nuestro ejemplo las estadísticas son las siguientes:

```
MODEL STATISTICS

BLOCKS OF EQUATIONS   4     SINGLE EQUATIONS    12
BLOCKS OF VARIABLES   3     SINGLE VARIABLES    29
NON ZERO ELEMENTS     75    DISCRETE VARIABLES  8
```

En la primera columna se cuentan el número de ecuaciones y variables genéricas y en la segunda el número total de variables y ecuaciones específicas. En caso tal que el problema modelado supere los valores límites GAMS no ejecutará el modelo.

Resumen de la solución: después de ejecutar el solver, GAMS genera un resumen corto de la solución cuyas entradas más importantes son SOLVER STATUS y MODEL STATUS y OBJETIVE VALUE, ya que, especifican si se ha encontrado la solución optima y el valor de la función objetivo.

```
S O L V E     S U M M A R Y

     MODEL   Localizacion        OBJECTIVE  fo

     TYPE    MIP                 DIRECTION  MINIMIZE
```

```
     SOLVER  CPLEX              FROM LINE  48



**** SOLVER STATUS     1 NORMAL COMPLETION

**** MODEL STATUS      1 OPTIMAL

**** OBJECTIVE VALUE   31500000.0000
```

Reporte resumen: El reporte resumen es la última parte de la compilación y muestra claramente el valor de las variables que intervienen en la solución óptima. La línea marcada con cuatro asteriscos es muy importante porque muestra si la solución no es óptima, es infactible o no acotada. A continuación se muestra el reporte:

```
**** REPORT SUMMARY :        0     NONOPT
                             0 INFEASIBLE
                             0  UNBOUNDED

----     49 VARIABLE y.L

             BOGOTA     CARACAS
PLANTA1                   1.000
PLANTA2        1.000

----     49 VARIABLE x.L

           BOGOTA   CARACAS   BRASILIA   MEXICO   LIMA
BOGOTA      70000              65000
CARACAS              20000     20000      100000   60000
```

La solución de nuestro ejemplo es ubicar la PLANTA1 en CARACAS y la PLANTA2 en BOGOTA, la PLANTA2 debe suplir la demanda de BOGOTA (70000) y BRASILIA (65000) y la PLANTA1 debe suplir la demanda de CARACAS (20000), BRASILIA (20000) MEXICO (100000) Y LIMA (60000) con un costo total de transporte de $3.150.000.

2 LENGUAJE AVANZADO

En este capítulo se explicará un lenguaje avanzado de GAMS necesario para programar problemas de tallas más elevadas, de igual manera, hacer que los modelos planteados en GAMS sean más claros y entendibles para el lector. Algunos ejemplos de problemas de este tipo son los problemas combinatorios de los que hace parte el ruteo de vehículos, la secuenciación de la producción, asignación de rutas aéreas, entre otros.

2.1 COMPARACIONES NUMÉRICAS

Se usan principalmente en modelos que necesiten condicionales y se pueden formar empleando condiciones lógicas la cuales comparan dos expresiones numéricas para ver si una es igual, diferente, mayor, menor, menor igual o mayor e igual que la otra. El comando empleado es el siguiente:

```
TERM$(TERMA OPERADOR TERMB)
```

Los operadores empleados en GAMS se muestran en la tabla 3.

Tabla 4. Operadores de GAMS.

RELACIÓN	OPERADOR GAMS	EXPLICACIÓN
Igualdad	**Eq ó =**	TermA = TermB
Diferente	**Ne ó <>**	TermA <> TermB
Mayor que	**Gt ó >**	TermA > TermB
Mayor o igual	**Ge ó >=**	TermA > =TermB
Menor que	**Lt ó <**	TermA < TermB
Menor o igual	**Lt ó <=**	TermA <= TermB
Ejemplo:	`P$(ORD(I) NE ORD(P))`	

FUENTE: McCarl GAMS User Guide

2.2 COMANDOS ORD Y CARD

Los comandos ordinalidad (ord) y cardinalidad (card) son funciones que permiten conocer la posición relativa y el número de elementos de un determinado conjunto de acuerdo a los requerimientos del problema a modelar.

- **Ord:** Hace referencia a la posición de cada elemento dentro de un determinado conjunto. Por ejemplo, en una ecuación al escribir el comando ***ORD (I) NE ORD (P)***, al ejecutar el modelo cuando I y P tomen el mismo valor de posición no se tendrá en cuenta este término en la ecuación.

- **Card**: Este comando da el número de elementos que conforman un conjunto, el comando a utilizar es el siguiente:

  ```
  NUMERO = CARD (i);
  ```

  Donde, el parámetro número es igual al número de elementos del conjunto i.

2.3 COMANDOS AND, OR Y DIAG

- **And:** Se usa cuando se desea realizar una acción y dos o más condiciones lógicas son verdaderas simultáneamente. El formato se muestra a continuación:

  ```
  ACCION$(CONDICION LOGICA 1 AND CONDICION LOGICA 2 AND
  CONDICION LOGICA 3)
  ```

- **Or**: Se usa cuando se desea que alguna de las dos o más condiciones lógicas se cumpla. Para su ejecución se usa el mismo formato cambiando la palabra AND por la palabra OR.

  Tanto el operador OR como el AND se pueden usar de manera simultánea y GAMS lo ejecutará teniendo en cuenta los todos los operadores definidos.

- **DIAG:** El comando `DIAG(I,J)` ó `DIAG(I,"TEXTO")` retorna 1 si `I` es igual a `J` ó si `I` es igual a `"TEXTO"` y cero en caso contrario.

2.4 $ CONDICIONALES

- **Condicional en una ecuación:** Para generar subconjuntos convenientes de los conjuntos ordenados originalmente puede utilizarse el símbolo '$'. El comando es:

```
LLEGAUNAVEZ(P)$(ord(P) gt 1)..    SUM(I, X(I,P)) =E= 1;
```

Es similar a decir que:

```
LLEGAUNAVEZ(2).. SUM(I, X(I,'2')) =E= 1;
LLEGAUNAVEZ(3).. SUM(I, X(I,'3')) =E= 1;
```

Note que el comando `(ORD(P) gt 1)` indica que solo los elementos cuyo ordinal sea mayor que 1 deben ser incluidos en la ecuación.

- **Condicional que ejecuta una asignación.**

Tabla 5. Condicionales de Asignación.

CONDICIÓN	EXPLICACIÓN
`X$(Y gt 0) = 20;`	Hace X=20 sólo si Y es mayor que 0
`PORC$(Y ne 0)= 100*(X-Y)/Y;`	Calcula el porcentaje (PORC) si Y no es igual a 0

FUENTE: Autoras del manual.

- **Condicional que agrega un término a una suma u otra operación.**

Tabla 6. Condicionales de adición.

CONDICIÓN	EXPLICACIÓN

`z=sum(I$(Y(I) gt 0),X(I));`	Incluye el termino para el conjunto de elementos de i solo si `Y(I)` `es mayor que` **0**
`z=sum((I,J)$(sameas(I,J)),X(I,J));`	Agrega el termino **X(I,J)** si **I** tiene el mismo valor de **J**.

FUENTE: Autoras del manual.

2.5 COMENTARIOS.

En GAMS, al igual que en otros programas de modelado, es posible escribir líneas de comentarios que faciliten la lectura del modelo. Se puede hacer de tres formas: primero empezando la línea con un (*), en el cual sólo se puede escribir un línea de comentario. La segunda forma es escribir el comando ($ontext) al inicio del renglón y ($offtext) al finalizar, la diferencia con el anterior es que se pueden escribir varias líneas de texto. La tercera forma se usa para hacer comentarios después del (;), la forma de utilizarlo es escribir el comando ($eolcom //) antes de hacer los comentarios y luego seguido del punto y coma usar (//) y hacer el comentario pertinente. A continuación se presentará un ejemplo.

- Uso del asterisco **(*)**:

  *A continuación se escribe el comentario que se desea.

- Uso de **$ontext** y **$offtext**:

  $ontext

  Se puede escribir más de una línea, principalmente se emplea para describir el problema que será modelado en GMAS.

  $offtext

❖ Uso de **$eolcom //:**

```
OBJ.. fo=E=SUM((i,j), C(i,j)*x(i,j)); //ecuación función objetivo
```

2.6 PALABRAS RESERVADAS

Al igual que otros sistemas de modelado, GAMS posee una lista de palabras reservadas que no se pueden usar para otro fin. Estas palabras presentadas en la tabla 6, se identifican en el archivo .gms por defecto con color azul. En este manual no se alcanzan a explicar todas las palabras reservadas. Sin embargo, se incluyen en la tabla para tenerlas en cuenta a la hora de formular el modelo.

Tabla 7. Palabras reservadas de GAMS.

PALABRAS RESERVADAS				
abort	equation	maximizing	parameters	smin
acronym	equations	Minimizing	positive	solve
acronyms	File	Model	prod	sos1
alias	files	Models	putpage	sos2
all	For	Na	puttl	sum
and	free	Ne	repeat	system
assign	Ge	Negative	sameas	table
binary	Gt	No	scalar	then
card	If	Not	scalars	until
diag	Inf	Option	semicont	using
display	integer	Options	semiint	variable

else	Le	Or	set	while
eps	loop	Ord	sets	xor
eq	Lt	Parameter	smax	yes

FUENTE: Tutorial by Richard E. Rosenthal

2.7 REGLAS EMPLEADAS EN GAMS

Existen reglas en gams para determinar el máximo número de caracteres que su pueden usar para dar nombres a los comandos, nombres a los subíndices y extensión del texto explicativo. A continuación se discutirán dichas reglas.

Reglas para asignar nombre a los comandos: Los comandos a los cuales se les puede asignar un nombre son: *sets, scalars, parameters, tables, files, acronyms, variables, equations* y *models.* El nombre de cualquiera de estos elementos debe cumplir con las siguientes reglas.

- La longitud de cada nombre no puede superar 63 caracteres.
- Comienza con un carácter alfabético.
- Puede contener caracteres alfanuméricos.
- No puede contener espacios.
- No puede contener caracteres especiales a excepción del guión de piso "_".
- El nombre deber se claro y preciso para evitar confusiones a la hora de entender el modelo.

En los siguientes ejemplos se puede ver en color negro el nombre asignado al comando set.

```
SET CIUDADES_DE_COLOMBIA /BMANGA, BOGOTA, CALI/;
SET i /BMANGA, BOGOTA, CALI/;
VARIABLE FUNCION1OBJ;
```

```
PARAMETER CAPACIDAD(CIUDADES_DE_COLOMBIA)/. . . /;
```

Reglas para nombrar los subíndices: Los nombres de los subíndices deben seguir el siguiente conjunto de reglas.

- El nombre del subíndice no puede tener más de 16 caracteres.
- En los primeros 10 caracteres se debe identificar el nombre del subíndice, ya que, se trunca en la salida de datos.
- Puede empezar por caracteres numéricos o alfabéticos.
- No están permitidos los espacios.
- No se permiten las palabras reservadas de GAMS como (set, table, variable, parameter, etc.).
- Se pueden usar caracteres como (+,-,_).

En seguida, en color verde se denota el uso de las reglas para nombrar los subíndices.

```
SET CIUDADES_DE_COLOMBIA /BMANGA, BOGOTA, CALI/;
SET CIUDADES_DE_COLOMBIA /B+MANGA, BO-TA, CA_LI/;
```

Reglas para los textos explicativos: Los textos explicativos se usan los comandos *sets, scalars, parameters, tables, files, acronyms, variables, equations* y *models* aplicando las siguientes reglas.

- El uso es opcional.
- Puede contener espacios.
- El máximo número de caracteres es de 255.
- No debe ocupar más de una línea.
- No puede contener palabras reservadas de GAMS.

A continuación se ejemplifica el uso del texto explicativo (color rojo) en la plataforma GMAS.

```
SET  i Posible ubicación /BOGOTA, BMANGA, CALI, CARACAS/;
PARAMETER a(k) Capacidad de las plantas
```

2.8 HERRAMIENTAS PARA LOS CÁLCULOS MATEMÁTICOS.

En todos los problemas modelados en GAMS son ampliamente utilizados los operadores matemáticos, con el fin de conocer un poco más a fondo el lenguaje de modelado GAMS a continuación se presentaran los diferentes tipos de operadores que emplea dicha herramienta.

2.8.1 Operadores

Los operadores básicos en la ecuaciones son el signo igual (=), dos puntos (. .), el signo más (+), menos (-), multiplicación (*), división (/), y exponenciación (**). De igual manera, los paréntesis ([, (y {) son considerablemente utilizados.

Signo igual: Se usa en todos las asignaciones del modelos en GAMS. Su presentación se ve a continuación.

```
X=7;
X(I)=10;
DEM(I)$Z(I)=Z(I);
X(I)=YES;
```

Lo que significa que la expresión del lado izquierdo es igual a la del lado derecho, el termino de la izquierda siempre debe pertenecer a los comandos set, scalar o parameter o puede emplear un condicional. El lado derecho de la igualdad puede ser una constante, un valor especial o una expresión.

Dos puntos: Es un signo elemental a la hora de escribir las ecuaciones en la interface GAMS. La formato se muestra enseguida.

```
OBJ..          Z =E= SUM(LT(I,J), D(I,J)*X(I,J));
REST1(K)..     SUM(LT(I,K), X(I,K)) + SUM(LT(K,J),X(K,J))=E=2;
```

Aritmética básica: GAMS emplea los siguientes símbolos para realizar las operaciones básicas que comúnmente suelen aparecer en las ecuaciones +, -, * y **. Veamos ejemplos de dichas operaciones.

```
X = 7 + 5 + SQRT(7);
F(I)= + X(I);
X = X - 3- SQRT(9);
F(I) = 18 - X;
REST2[I].. X(I) =E= 20 - X;
X = 9**SQRT(7);
F(I) = 20.0**X;
ZZ(I)$(X(I)-8 GT 0) = Z(I)**0.5;
Z(I)=19*X**2/9-8+2;
```

Como se puede observar, estos signos no pueden ir al lado izquierdo de la ecuación a menos que se use para evaluar un condicional. En GAMS los operadores aritméticos poseen una prioridad, la exponenciación tiene prioridad 1, la multiplicación y división prioridad 2 y la suma y resta prioridad 3. Los operaciones que contengan signos de prioridad 1 se resolverán primero, luego la 2 y seguidamente la 3. Sin embargo, si existen operaciones entre paréntesis, estas se resolverán primero.

Paréntesis: Cualquiera de esto paréntesis se pueden utilizar ( ), { } o [ ], las operaciones que estén dentro de estos son las primeras en resolver. Por ejemplo:

```
X{I}=+{25/(8+15)}+9;
REST1[I].. X{I}=G=10*[7+1]**{34/(8+15)}+10-2;
```

Acá, la suma(8 + 15) que se encuentra dentro de los paréntesis es la primera que se resuelve.

2.8.2 Operaciones en elementos dependientes

Otras operaciones que comúnmente encontramos en los problemas que se modelan en GAMS son encontrar el mayor (smax), o menor (smin) número dentro de un conjunto, hacer sumas (sum) y realizar productos (prod) entre los elementos. A continuación se discutirán y se ejemplificará cada uno de estas operaciones.

Smax y smin: La función del comando smax es encontrar el valor más grande dentro de un conjunto y smin el valor más pequeño. La notación se ejemplifica a continuación.

```
R=SMAX(I,X[I]);
R=SMAX((I,J),X(I)+Y(J));

R=SMIN(I,X[I]);
R=SMIN((I,J),X(I)+Y(J));
```

Sum: El formato utilizado para realizar sumas de un conjunto es como se ve a continuación.

```
OBJ..                      FO =E= SUM((I,J),C(I,J)*X(I,J));
R1(J)$(ord(J)gt 1)..       SUM(I, X(I,J)) =E= 1;
R3..                       SUM(J, X('0',J)) =E= K;
```

En la primera ecuación se realiza la sumatoria en I y J, en la segunda la sumatoria en I y en la tercera la sumatoria en J.

Prod: En GAMS también es posible calcular el producto entre los elementos de un conjunto, los siguientes ejemplos muestran como se usa la palabra reservada **prod.**

```
OBJ..                      FO =E= PROD((I,J),C(I,J)*X(I,J));
```

```
R1(J)$(ord(J)gt 1)..   PROD(I, X(I,J)) =E= 1;
R3..                   PROD(J, X('0',J)) =E= K;
```

En la primera ecuación se realiza el producto para los elementos de I y J, en la segunda, el producto para los elementos de I y en la tercera, el producto para los elementos de J.

2.8.3 Funciones

GAMS posee la siguiente lista de funciones disponibles en su biblioteca: ABS, EXECSEED, EXP, IFTHEN, LOG, LOG10, LOG2, ROUND, SQRT. La definición y ejemplificación de cada una de ellas se muestra en la tabla 7. De igual manera, se presentan otros tipos de funciones que posee la herramienta GMAS.

Tabla 8. Funciones matemáticas.

FUNCIÓN	DESCRIPCIÓN	EJEMPLOS

ABS	Devuelve el valor absoluto de una función. Es una función no continua, se puede usar en GAMS para el cálculo de las variables y parámetros del modelo. Sin embargo, si se usa en los cálculos de ecuaciones el modelo a resolver debe ser DNLP.	`X=ABS(I);`
		`X=ABS(Z+6);`
		`REST1..`
		`FOB=E=ABS(I);`

EXECSEED	Esta expresión se usa para restablecer y guardar la semilla de un generador de números aleatorios.	`CC=EXECSEED;`

EXP	Función que calcula e^x, es continua, se puede usar en GAMS para el cálculo de las variables y parámetros del modelo. Si se usa en ecuaciones, el modelo con el cual puede ser resuelto es NLP.	`X = EXP(I);`
		`X = EXP(Z+6);`
		`REST1..`
		`FOB=E=EXP(I);`

IFTHEN	La expresión contiene una función que cambia el valor dependiendo de una condición. Si la CONDICIÓN es verdadera, X toma el valor VERDAD, en otro caso, toma el valor FALSO. Se puede usar para el cálculo de datos, en ecuaciones se debe ejecutar con un modelo del tipo DNLP.	`X=IFTHEN(COND, VERDAD,FALSO);` `X=IFTHEN(Y=2, 5,7+Z);`

LOG, LOG10, LOG2	Función que calcula el logaritmo natural y el logaritmo en base de 10 de una expresión. Es una función continua, se puede usar para el cálculo de datos en GAMS. Si se usa en ecuaciones, el modelo con el cual puede ser resuelto debe ser un NLP.	`X=LOG(I);`
		`X=LOG(Z+6);`
		`EQ3..`
		`Z=E=LOG(I);`
		`X=LOG10(I);`
		`X=LOG10(Z+6);`
		`EQ4..`
		`Z=E=LOG10(I);`

ROUND	Esta función redondea el resultado de una numérico de una expresión. Presenta dos variantes, la primera redondea el número al entero más cercano y la segunda redondea el número con los decimales especificados. Solo se puede usar para hacer cálculos de datos en GAMS.	`X=ROUND(8.4561);` `X=ROUND(8.4461,2);`

SQRT	Calcula la raíz cuadrada de una expresión o un término. Es continua, se emplea para hacer cálculos de parámetros en GAMS, al emplearse en la ecuaciones del modelo se debe resolver con un NLP.	`X=SQRT(I+2);`
		`REST1..`
		`FOBJ=E= SQRT(I);`
		`X=SQRT(B('I5'));`

FUENTE: Autoras del manual.

Existen otras funciones presentadas en la tabla 8, no tan comunes en los modelos como las anteriormente mencionadas pero que se pueden utilizar en caso de ser requeridas teniendo en cuenta el tipo de modelo apropiado para su solución que además se menciona en la tabla. También, se mencionan otras funciones que la herramienta GAMS no las permite pero que son comunes en nuestra carrera de Ingeniería Industrial.

Tabla 9. Funciones matemáticas no comunes en GAMS.

FUNCIÓN	DESCRIPCIÓN	MODELO
SIN(X)	Devuelve el seno de X y X debe estar en radianes.	Requiere un modelo NLP y puede se discontinua.
COS (X)	Devuelve el coseno de X con el valor de X dado en radianes.	Requiere un modelo NLP y puede se discontinua.
TAN(X)	Devuelve el valor de la tangente de X y X debe estar en radianes.	Requiere un modelo NLP y puede se discontinua.
ARCSIN(X)	Reintegra el valor de la función invertida del seno de X, donde X debe estar en radianes.	El modelo requerido es un NLP.
ARCCOS (X)	Reintegra el valor de la función invertida del coseno de X, donde X debe estar en radianes.	El modelo requerido es un NLP.
ARCTAN(X)	Reintegra el valor de la función invertida de la tangente de X, donde X debe estar en radianes.	El modelo requerido es un NLP.

BINOMIAL[X,Y]	Función binomial generalizada, donde X eventos ocurren en un tiempo Y.	El modelo requerido es un NLP.
NORMAL(X,Y)	Números aleatorios normalmente distribuidos con media X y desviación estándar Y.	No está permitida.
UNIFORM(X,Y)	Números aleatorios distribuidos uniformemente entre X y Y.	No está permitida.
PI	Genera el número Pi = 3,141716…	Permitida.

FUENTE: Autoras del manual.

3 GAMS AND GAMS IDE

GAMS está dividido en dos partes. El primero es un editor que crea un archivo que contiene las instrucciones de GAMS y las guarda con extensión .gms., una vez la información este completa se ejecuta el programa con el solver elegido haciendo los cálculos necesarios creando un nuevo archivo de resultados con extensión .lst.

3.1 RAZONES FUNDAMENTALES PARA EL USO DE GAMS

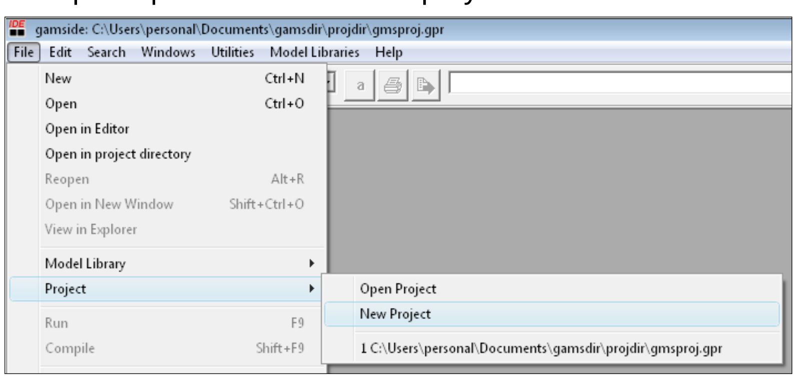
El desarrollo de GAMS viene desde antes de la disponibilidad de interfaces gráficas como Windows. Como resultado, GMAS fue diseñado para funcionar por un sistema de comandos en línea. Actualmente GAMS también se puede ejecutar en Windows mediante el Entorno Integrado de Desarrollo IDE por sus siglas en Inglés (Integrated Development Environment).

3.1.1 Pasos para usar IDE

Una vez instalado GAMS IDE se deben seguir una serie de pasos básicos para obtener resultados.

Paso 1: Crear el proyecto.

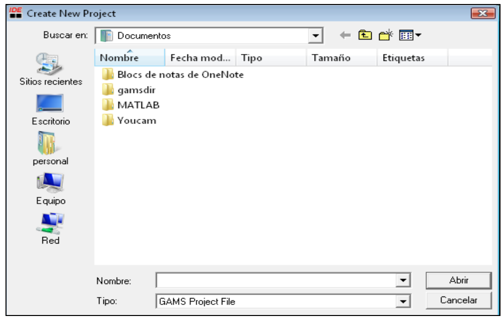
Ilustración 1. Opción para crear un nuevo proyecto.



FUENTE: GAMS

En el menú de GAMS, opción **file**, se selecciona **Project** y **New Project**, de allí será direccionado a la siguiente ventana en la que se especificará nombre y tipo de proyecto, al igual que, la dirección en donde el archivo va a estar localizado.

Ilustración 2. Opción para guardar proyectos nuevos.



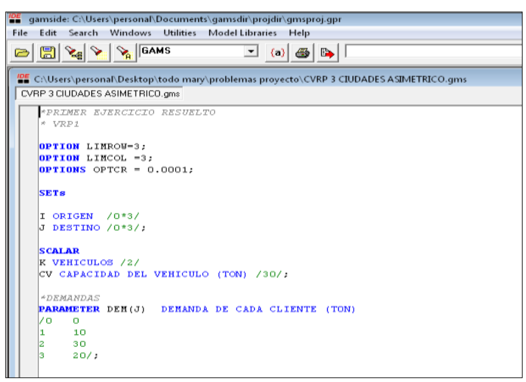
FUENTE: GAMS

En GAMS también es posible abrir un archivo ya existente o un modelo de la librería en la opción **Model Library.** La librería de GAMS contiene un gran número de ejemplos que sirven de ayuda al modelador para el desarrollo de nuevos proyectos.

Paso 2: Preparación del archivo para la ejecución.

IDE contiene un editor completo en el que se puede ir a través del archivo y cambiar lo que se desee en el momento que se requiera, aún después de ejecutar el modelo.

Ilustración 3. Editor GAMS.

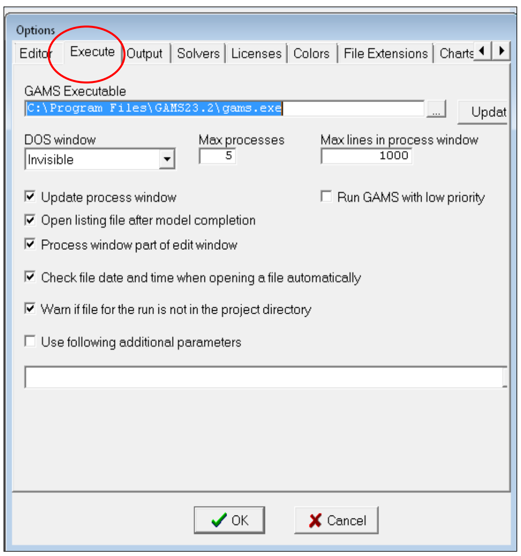


FUENTE: GAMS

Paso 3: Cambiar las funciones por defecto de IDE.

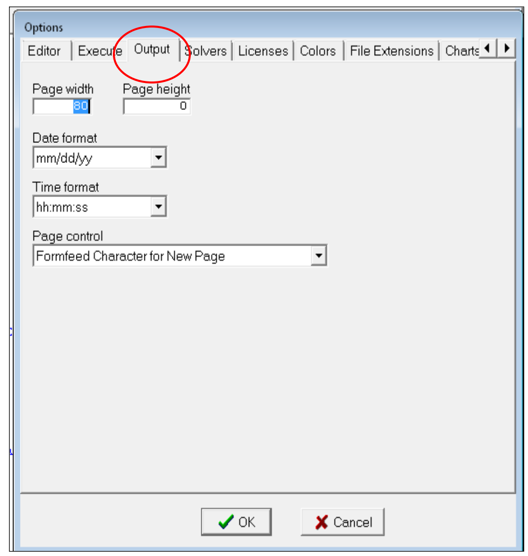
IDE permite personalizar las funciones que se van a utilizar, desplegando el menú file, options, en los iconos **Execute** y **Output** se podrá determinar el ancho y largo de la página. al igual que, haciendo click en el icono **File Extensions,** IDE puede hacer que el archivo que se abra sea un .gms, .lst, .gdx etc.

Ilustración 4. Opción execute.



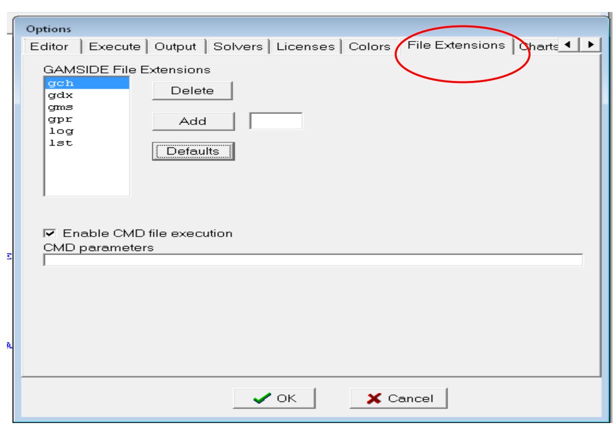
FUENTE: GAMS

Ilustración 5. Opción output.



FUENTE: GAMS

Ilustración 6. Opción file extensions.

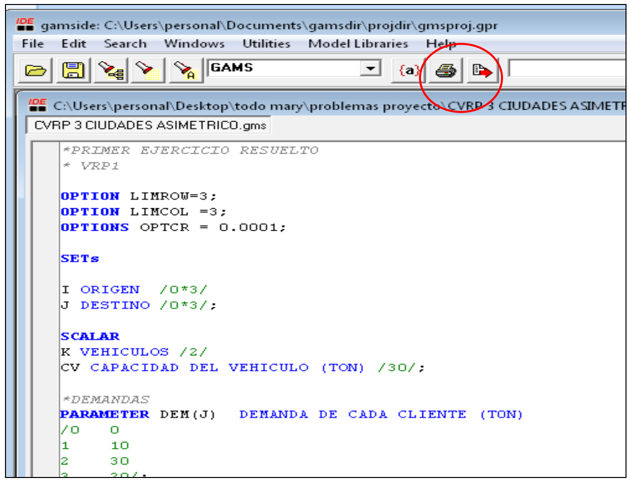


FUENTE: GAMS

Paso 4: Ejecutar el proyecto.

Para ejecutar el modelo simplemente se debe hacer clic en la flecha roja o presionar F9.

Ilustración 7. Opción para ejecutar el modelo.

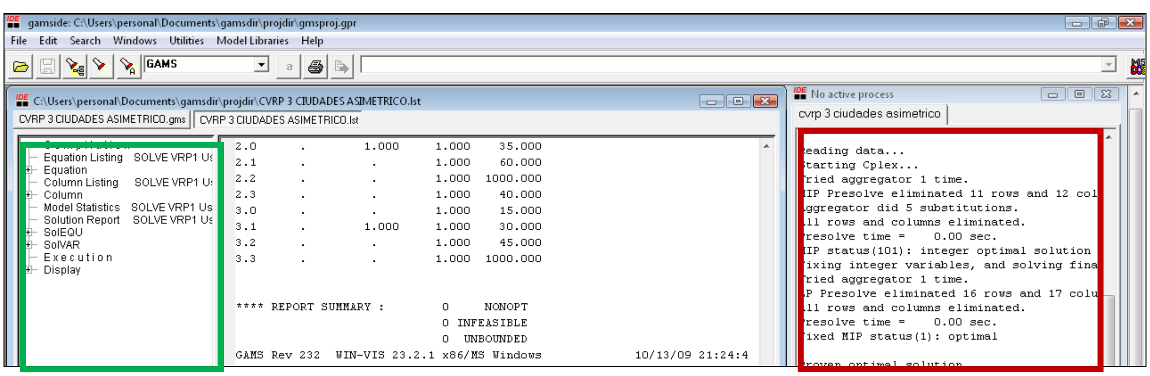


FUENTE: GAMS.

Paso 4: Conozca el archivo .lst.

Hay dos formas de hacerlo, utilizando la ventana de proceso (ver recuadro rojo) o puede utilizar el navegador (ver recuadro verde). Si está en la ventana de procesos y la línea está en rojo al hacer doble clic lo lleva al archivo .gms, mientras que, si el color es azul lo remitirá al archivo. lst. Teniendo acceso a líneas genarales o específicas del modelo.

Ilustración 8. Salida de GAMS - archivo .lst.



FUENTE: GAMS.

La tabla 9 muestra los colores posibles a encontrar en la ventana de proceso y la respectiva descripción.

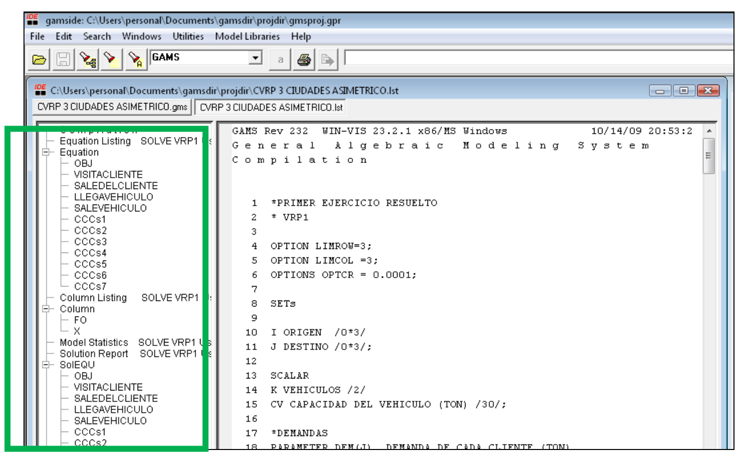
Tabla 10. Función de la ventana de proceso.

Color de línea	Función y destino al dar doble clic
Línea roja	Identifica los errores en el archivo original y muestra una descripción del error. Al hacer doble clic se abrirá el archivo .gms en la ubicación del código que causo el error.
Línea negra	Salta a la ubicación anterior de la última línea azul en el archivo LST.
Línea azul	Salta a la línea del archivo .lst correspondiente.

FUENTE: McCarl GAMS User Guide

Al hacer clic en el navegador (ver recuadro verde) puede tener acceso a la salida del programa en general o en lugares específicos. La posición en el archivo .lst está determinado por el tipo de línea a la que se hace clic.

Ilustración 9. Vista del navegador.



FUENTE: GAMS.

En la tabla 10 se muestra una lista con los tipos de líneas al que será direccionado al hacer clic en el navegador.

Tabla 11. Función del navegador.

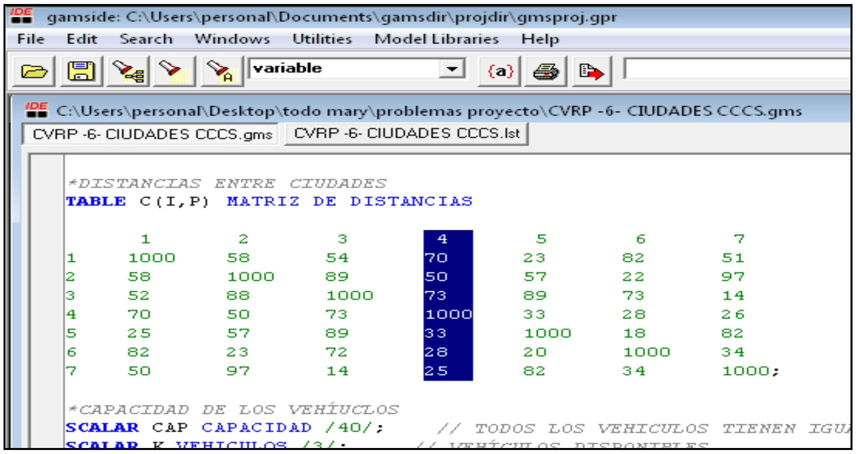
Tipo de línea	Función y destino al dar doble clic
Compilatión	Salta al inicio del archivo .lst
Equation listing	Va a la lista donde se encuentran las ecuaciones.
Equation	Salta específicamente en la ecuación que se desea ver.
Column listing	Salta al inicio de la lista de variables en el archivo .lst.
Column	Lleva a cada variable específica.
Model statistics	Va a las estadísticas del modelo.
Solution Report	Salta al reporte resumen de la solución.
Display	Muestra en el archivo .lst los valores de las variables del modelo.

FUENTE: McCarl GAMS User Guide

3.1.2 Mover bloques de columnas

Con solo teclear control + c y control + v, IDE permite copiar y pegar un texto normal. De igual forma, permite copiar o cortar una columna como la seleccionada en la figura.

Ilustración 10. Opción para seleccionar columnas en el editor.



FUENTE: GAMS.

Para esto primero se debe elegir la columna a copiar ubicando el cursor al inicio de esta, luego tecleando alt + shif y realizar el desplazamiento hacia abajo moviendo el mouse o las teclas de flecha.

Además de lo mencionado anteriormente GAMS IDE cuenta con un gran número de ventajas que le permiten al modelador hacer buen uso del programa.

3.2 TIPOS DE MODELOS

En GAMS es posible ejecutar problemas con diferentes tipos de modelos que van desde la programación lineal hasta modelos complejos como lo es la programación cuadrática. A continuación se presentaran algunos de los modelos disponibles, haciendo una corta explicación de sus características principales con el fin de que el modelador identifique claramente cuál es el tipo de modelo adecuado para resolver con la herramienta GAMS.

3.2.1 Modelos disponibles en GAMS

El lenguaje GAMS permite modelar problemas como:

Programación lineal (LP): Para esta clase de modelo de optimización, tanto la función objetivo como las restricciones son estrictamente lineales. Este tipo de programación también ha sido la base para el desarrollo de los algoritmos de solución en problemas más complejos como lo es la programación entera, no lineal y estocástica. La formulación de este modelo se muestra a continuación:

$$
\begin{aligned}
&\textbf{Optimizar} \quad CX \\
&\textbf{Sujeta a:} \quad AX \;\alpha\; B \\
&\qquad\qquad\quad\;\; X \geq 0
\end{aligned}
$$

Donde,

X, representa el vector de variables, con límites inferior y superior por lo general entre 0 e ∞ respectivamente.

AX α B, representa el conjunto de restricciones que pueden ser la forma =, ≤ ó ≥.

CX, es la función objetivo.

La Programación Lineal abarca problemas de agricultura, salud, transporte, economía, industria y militares generando problemas con miles de restricciones y variables lo que lo hace interesante para la programación.

Programación no lineal (NLP): Este tipo de programación se presenta cuando la función objetivo y/o las restricciones son no lineales. Un problema de programación no lineal se puede escribir matemáticamente de la siguiente manera.

$$
\begin{aligned}
&\textbf{Optimizar} \quad F(X) \\
&\textbf{Sujeta a:} \quad h_j(X) = 0 \qquad j = 1,\dots,m \\
&\qquad\qquad\quad\;\; g_j(X) \geq 0 \qquad j = m+1,\dots,p \\
&\qquad\qquad\quad\;\; X \in E^n
\end{aligned}
$$

Donde,

F(X), representa la función objetivo.

$h_j(X)$, Son funciones continuas con restricciones de igualdad.

$g_j(X)$, representa funciones continuas con restricciones de desigualdad.

X, representa el vector de variables que pertenecen a E^n.

Los casos a los cuales aplica la optimización no lineal son procesos químicos, cotización de proyectos, ajuste de curvas, problemas de diseño estructural, entre otros.

Programación cuadrática restringida (QCP): La programación cuadrática optimiza la función objetivo cuadrática bajo un conjunto de restricciones de igualdad y desigualdad. La formulación matemática es como sigue.

Optimizar $CX + X'QX$

Sujeta a: $A_iX + X'R_iX \; \alpha \; B_i$ Para todo i

$L \leq X \geq U$

Donde,

$C(X)$, corresponde a la parte lineal de la función objetivo.

$X'QX$, corresponde a la parte cuadrática de la función objetivo.

A_iX, representa la parte lineal de las restricciones.

$X'R_iX$, representa la parte cuadrática de las restricciones.

B_i, corresponde la i-ésimo elemento del lado derecho de las restricciones.

α, indica que las restricciones pueden ser =, ≥ ó ≤.

X, es el vector de variables continuas en los reales.

$L \leq X \geq U$, L es el límite inferior y U el límite superior de las variables.

Programación entera mixta (MIP): La programación entera tiene tanto la función objetivo como las restricciones igual a un problema de programación lineal, la diferencia radica en que algunas de las variables de decisión deben tomar valores enteros en la solución final. La programación entera se divide en tres partes: programación entera pura (todos los valores de las variables en la solución final deben ser enteros), programación entera mixta (se requiere que algunas de las variables de decisión de la solución final sean enteras) y programación entera binaria (son casos en los cuales la variables de decisión solo toman valores de 0 y 1).

Resolver un problema de programación entera es más difícil de resolver que un problema de programación lineal, ya que, dependiendo de la complejidad del problema aumenta el tiempo de solución. Uno de los algoritmos más empleados para dar solución exacta a los problemas de programación entera es el algoritmo Branch and Bound, algoritmo ampliamente estudiado en el desarrollo de Seminario de Investigación.

La formulación matemática general de los problemas de programación entera mixta se presenta a continuación.

$$
\begin{array}{lll}
\textbf{Optimizar} & C_1T + C_2U + C_3V + C_4W + C_5X + C_6Y + C_7Z & \\
\textbf{Sujeta a:} & A_1T + A_2U + A_3V + A_4W + A_5X + A_6Y + A_7Z & \\
 & T & \geq 0 \\
 & U & \geq 0, \leq L_2 \text{ y entera} \\
 & V & \in (0,1) \\
 & W & \in SOS1 \\
 & X & \in SOS2 \\
 & Y & = 0 \text{ ó } \geq L_6 \\
 & Z & \geq 0, \leq L_7 \text{ y entera}
\end{array}
$$

Donde,

$C_1T + C_2U + C_3V + C_4W + C_5X + C_6Y + C_7Z$, representa la función objetivo.

$A_1T + A_2U + A_3V + A_4W + A_5X + A_6Y + A_7Z \ \alpha \ B$, representa las restricciones, donde α puede ser =, ≥ ó ≤.

T, son variables continuas en los reales.

U, variables mayores que cero, con límite superior L_2 y enteras.

V, representa las variables binarias.

W, variables en las que solo una de ellas puede tomar valor diferente de cero.

X, variables en las que solo dos de ellas pueden tener valores distintos de cero. Dichos valores tienen que ser adyacentes.

Y, representa variables semi-continuas que pueden tomar el valor de cero ó mayor o igual a $L_{6.}$

Z, representa variables semi-enteras que pueden tomar el valor de cero ó menor o igual a L_7 y enteras.

Programación entera mixta relajada (RMIP): Problema similar a MIP, con la diferencia que se amplía el espacio de solución de las variables eliminando las condiciones SOS, semi-continuas y semi-enteras. Este problema es más fácil de resolver y es probable encontrar rápidamente la solución entera factible.

$$\textbf{Optimizar} \quad C_1T + C_2U + C_3V + C_4W + C_5X + C_6Y + C_7Z$$

$$
\begin{array}{lll}
\textbf{Sujeta a:} & A_1T + A_2U + A_3V + A_4W + A_5X + A_6Y + A_7Z & \\
 & T & \geq 0 \\
 & U & \geq 0, \ \leq L_2 \\
 & V & \epsilon \ (0,1) \\
 & W & \geq 0 \\
 & X & \geq 0 \\
 & Y & \geq 0 \\
 & Z & \geq 0
\end{array}
$$

Tanto para la MIP como para la RMIP se debe observar la matriz de capacidad del solver ilustración 12, donde se especifica el solver adecuado para cada tipo de modelo.

Además de los modelos mencionados anteriormente, GAMS tiene para disposición del usuario los siguientes modelos: Programación no lineal entera mixta (MINLP), Programación no lineal entera mixta relajada (RMINLP), Programación restringida cuadráticamente entera mixta (MIQCP), Programación restringida cuadráticamente entera mixta relajada (RMIQCP), entre otros.

3.3 SOLVERS

En la tabla 11 se presenta la lista de Solvers disponibles en la versión 23.3 de GAMS haciendo énfasis en CPLEX, licencia con la cual cuenta la Escuela de Estudios Empresariales y posee la versión estudiante de GAMS. Es de vital importancia destacar que GAMS Corporation no desarrolla software para resolver los modelos (solver), solo crea los procedimientos para unir el software a la interface GAMS. Sin embargo, si se puede obtener la licencia a través de GAMS Corporation para el uso del software.

Tabla 12. Lista de Solvers - versión 23.3.

SOLVERS				
AECPWRAP	COINGUROBI	DECISM	LMBIGM	NLPEC
ALPHAECP	COINIPOPT	DICOPT	LMCHULL	OQNLP
AMPL	COINMOSEK	EMP	LMLBOA	OSL
BARON	COINOS	EXAMINER	LS	OSLSE
BDMPL	COINSCIP	GAMSCHK	MILES	PATH
BENCH	COINXPRESS	GUROBI	MINOS	PATHNLP
COINBONMIN	CONOPT	KNITRO	MINOS5	SBB
COINCBC	CONVERT	LGO	MOSEK	SNOPT
COINCOUENNE	CPLEX	LINDOGLOBAL	MPECDUMP	XA
COINCPLEX	DEA	LINDOWRAP	MPSGE	XPRESS
COINGLPK	DECISC	LINGO	MSNLP	

Fuente: Autoras del manual.

CPLEX: GAMS/CPLEX permite combinar el lenguaje de modelado de alto nivel de GAMS con el poder del optimizador CPLEX. El software fue desarrollado por ILOG mediante el cual se pueden resolver modelos de programación lineal, programación entera mixta, programación cuadrática. Los algoritmos con los cuales cuenta para resolver los modelos son: primal simplex, dual simplex, optimizador de redes, de barrera o punto interior, capacidad cuadrática, entero mixto y un buscador de inviabilidad. En problemas con variables enteras y enteras mixtas que requieren de mayor complejidad computacional, CPLEX usa el algoritmo Branch and Cut resolviendo una serie de subproblemas de programación lineal.

Mientras el algoritmo Branch an Bound hace la búsqueda en problemas de MIP, CPLEX genera el reporte presentado en la Ilustración 11, en la que se puede observar en el orden correspondiente, el número de nodos, el valor de la función objetivo, las variables que tienen valor fraccionario, la mejor solución encontrada en cada iteración, la mejor solución relajada y el porcentaje de optimalidad gap. CPLEX también marca con un asterisco (*) la fila en la que se encuentre la mejor solución entera factible o un candidato a ser mejor solución.

Ilustración 11. Salida de datos- algoritmo Branch and Bound.

```
   Node  Left     Objective  IInf  Best Integer     Best Node    ItCnt     Gap

      0     0      338.3729    10                    338.3729       19
*     0+    0                         1350.0000      338.3729       19   74.94%
      0     0      357.5000     6     1350.0000     Cuts: 19        36   73.52%
*     0+    0                         1346.0000      357.5000       36   73.44%
      0     0      357.5000     6     1346.0000   Impl Bds: 3       37   73.44%
*     0+    0                         1340.0000      357.5000       37   73.32%
*     0+    0                          430.0000      357.5000       37   16.86%
      0     2      357.5000     6      430.0000      358.0000       37   16.74%
*     3     3      integral     0      377.0000      363.5500       63    3.57%
*     6     4      integral     0      376.0000      363.5500       74    3.31%
*    10     4      integral     0      373.0000      363.5781       89    2.53%
                                                   Impl Bds: 6
*    16     1      integral     0      372.0000      369.9500      105    0.55%

Clique cuts applied:  3
Implied bound cuts applied:  11
MIP status(101): integer optimal solution
Fixing integer variables, and solving final LP...
Tried aggregator 1 time.
LP Presolve eliminated 51 rows and 56 columns.
All rows and columns eliminated.
Presolve time =   -0.00 sec.
Fixed MIP status(1): optimal

Proven optimal solution.

MIP Solution:           372.000000    (105 iterations, 18 nodes)
Final Solve:            372.000000    (0 iterations)

Best possible:          372.000000
Absolute gap:             0.000000
```

FUENTE: GAMS

3.3.1 Matriz de capacidad del solver

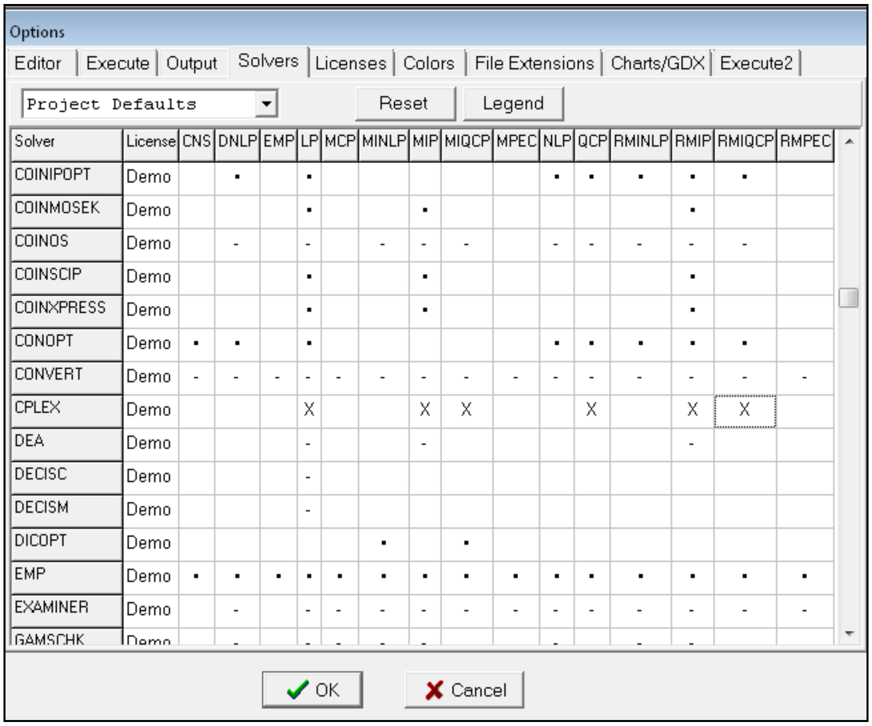
La ilustración 12 mostrada abajo representa la matriz de capacidad de cada solver, esta matriz se obtuvo de la versión de GAMS IDE 22,6 e indica que tipo de modelo puede resolver cada solver. Por otro lado, la ilustración 13 representa la matriz de capacidad de la última versión GAMS IDE 23,3 indicando con una X los modelos que puede resolver el optimizador CPLEX.

Ilustración 12. Matriz de capacidad del solver - versión 22.6

Solver/Model type availability - 22.6 December 24, 2007												
	LP	MIP	NLP	MCP	MPEC	CNS	DNLP	MINLP	QCP	MIQCP	*Stoch.*	*Global*
ALPHAECP								✔		✔		
BARON 8.1	✔	✔	✔				✔	✔	✔	✔		✔
BDMLP	✔	✔										
COIN	✔	✔										
CONOPT 3	✔		✔			✔	✔		✔			
CPLEX 11.0	✔	✔							✔	✔		
DECIS	✔										✔	
DICOPT								✔		✔		
KNITRO 5.1	✔		✔				✔		✔			
LINDOGLOBAL 5.0	✔	✔	✔				✔	✔	✔	✔		
LGO	✔		✔				✔		✔			✔
MILES				✔								
MINOS	✔		✔				✔		✔			
MOSEK 5	✔	✔	✔				✔		✔	✔		
MPSGE												
MSNLP			✔				✔		✔			✔
NLPEC				✔	✔							
OQNLP			✔				✔	✔	✔	✔		✔
OSL V3	✔	✔										
OSLSE	✔										✔	
PATH				✔		✔						
SBB								✔		✔		
SNOPT	✔		✔				✔		✔			
XA	✔	✔										
XPRESS 18.00	✔	✔							✔			
Contributed Plug&Play solvers												
AMPLwrap	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
DEA	✔	✔										
Kestrel	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

FUENTE: McCarl GAMS User Guide

Ilustración 13. Matriz de capacidad del solver - versión 23.3



Solver	License	CNS	DNLP	EMP	LP	MCP	MINLP	MIP	MIQCP	MPEC	NLP	QCP	RMINLP	RMIP	RMIQCP	RMPEC
COINIPOPT	Demo		▪		▪						▪	▪	▪	▪	▪	
COINMOSEK	Demo				▪			▪						▪		
COINOS	Demo		-		-		-	-	-		-	-	-	-	-	
COINSCIP	Demo				▪			▪						▪		
COINXPRESS	Demo				▪			▪						▪		
CONOPT	Demo	▪	▪		▪						▪	▪	▪	▪	▪	
CONVERT	Demo	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CPLEX	Demo				X			X	X			X		X	X	
DEA	Demo				-			-						-		
DECISC	Demo				-											
DECISM	Demo				-											
DICOPT	Demo						▪		▪							
EMP	Demo	▪	▪	▪	▪	▪	▪	▪	▪	▪	▪	▪	▪	▪	▪	▪
EXAMINER	Demo		-		-	-	-	-	-	-	-	-	-	-	-	-
GAMSCHK	Demo		-		-	-	-	-			-		-	-		

Fuente: GAMS

4 COMPILACIÓN DE ERRORES

La ejecución de un programa GAMS pasa a través de una serie de etapas, la primera de las cuales es la compilación. Al ejecutar el programa muchas veces se encuentra con el mensaje: Errores de compilación que indica que un número elevado de errores se están presentando en el diseño del modelo, GAMS cubre el proceso de encontrar y corregir dichos errores. Si el modelo presenta errores, en la ventana de procesos se encontraran unas líneas de texto de color rojo en la que se enuncia el tipo de error con su respectiva explicación.

4.1 CÓMO ENCONTRAR ERRORES

Si al ejecutar el modelo en el navegador se encuentran líneas de color rojo, al hacer doble clic sobre ellas inmediatamente lo lleva al sitio donde se cometió el error. Se recomienda empezar por el primer error suministrado por GAMS, ya que, al corregirlo este ocasionará que algunos de los errores siguientes desaparezcan.

En la tabla 12 se presentan algunos de los errores más comunes que se suelen cometer en GAMS identificando la causa común del error.

Tabla 13. Errores comunes en GAMS.

MENSAJE DE ERROR	CAUSA COMUN DE ERROR	EJEMPLO
8	Falta un paréntesis “(“.	**** 8 ')' expected

36	Faltan elementos en la definición de la ecuación.	**** 36 '=' or '..' or ':=' or '$=' operator expected
37	Falta de especificación en el tipo de ecuación.	**** 37 '=l=' or '=e=' or '=g=' operator expected
51-60	El modelo no se puede solucionar por programación lineal.	**** 51 Endogenous function argument(s) not allowed in linear models
66	No ha sido declarado un conjunto, una tabla o un escalar para una ecuación.	**** 66 The symbol shown has not been defined or assigned
71	La ecuación ha sido declarada pero no con el comando correspondiente	**** 71 The symbol shown has been declared as an equation, but no Symbolic equation (..) was found.
96	Un comando termina y otro empieza y no fue incluido un ";".	****check for missing ';' on previous line)
195	Indica que un nombre ya ha sido definido anteriormente.	**** 195 Symbol redefined with a different type

FUENTE: McCarl GAMS User Guide

En todos los sistemas de modelado es importante detectar fácilmente la ubicación del error de compilación, es por eso que GAMS está diseñado para encontrar rápidamente el tipo de error cometido, dar una explicación e indicar la causa del error, los errores se encuentran en la etapa de corrección del proceso o compilación.

Durante la ejecución de los primeros modelos en GAMS es probable que se cometan errores ya que el modelador no está familiarizado con el lenguaje. Sin Embargo, esto no es motivo de preocupación, solo se necesita habilidad para la programación y un buen manejo del inglés, ya que, los tanto los comandos utilizados en GAMS como los manuales de ayuda se encuentran en dicho idioma.

Tan pronto como se detecta un error, se detiene la ejecución del modelo y nunca será resuelto, la solución está en encontrar el error y solucionarlo. Existen tres grupos en que se pueden agrupar los errores: compilación, ejecución y generación del modelo que serán discutidas a continuación.

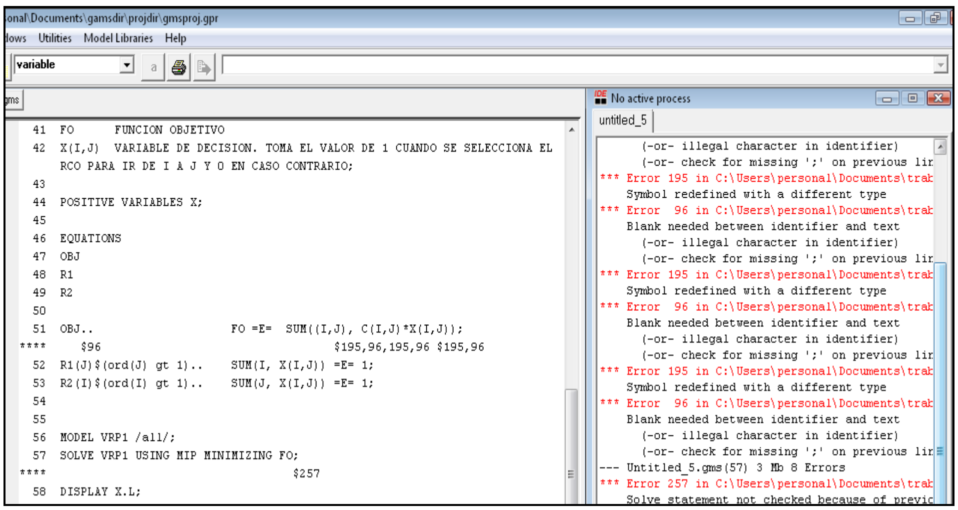
4.2 ERRORES DE COMPILACIÓN

Durante la compilación se pueden detectar una gran cantidad de errores que a menudo se remontan a la entrada de datos en la interface GAMS, algunos de los errores son muy simples: falta de un punto y coma ";", olvidar definir una variable o una ecuación, un error de escritura en un comando, entre otros. Estos errores también ocasionan que se comentan otros errores, para estos casos, el mensaje de error le ayudara a corregirlo. En una línea se pueden encontrar varios errores de compilación, no obstante, GAMS no enumerara más de diez.

Puede ser que el modelo no haga lo que se desea que haga, pero a este tipo de errores no se tratarán en este numeral, se sugiere revisar el modelo, ejecutarlo y contrarrestar las ecuaciones de entrada con las de la compilación para tener una mejor idea del error que se está cometiendo.

En la ilustración 14 se muestran los errores ocasionados por la falta de un punto y coma ";" en línea 49, al terminar de definir las ecuaciones. En este ejemplo se ve claramente que el ";" puede ocasionar un gran número de errores en el modelo y no permite ejecutar el modelo correctamente.

Ilustración 14. Ventana de errores.



FUENTE: GAMS

4.2.1 Características de los errores de compilación.

De los errores de compilación se pueden rescatar las siguientes características.

- En la compilación o listado de salida al comienzo de la línea el error se marca con cuatro asteriscos '****'.
- Los errores de compilación se identifican por el símbolo dólar '$' seguido del número que identifica el error y en una línea independiente que empieza con cuatro asteriscos.
- En la ventana de procesos una lista con todos los números de error encontrados, de igual forma, se mostrará una descripción de la causa probable de cada error.

- Muchas veces un error se detecta hasta la siguiente declaración donde se encuentra, siempre revise cuidadosamente las declaraciones anteriores con el fin de hallar la causa original del problema.

4.3 ERRORES EN TIEMPO DE COMPILACIÓN

El análisis de las declaraciones (comandos) es más complicado que los errores de compilación, ya que, esto implica tiempo en la revisión del modelo. Por ejemplo, las ecuaciones matemáticas deben coincidir correctamente con lo que se espera del modelo matemático original, el tipo de modelo (LP, MIP, NLP) por el que se desea resolver el problema debe ser el adecuado con el fin de evitar obtener resultados equivocados. Otro aspecto importante de este tipo de errores se debe a que los mensajes de erros pueden ser erróneos y confusos porque se está resolviendo un programa que contiene más errores.

El mensaje de error aparece en dos sitios diferentes, primero, como los errores de compilación se muestran debajo de la declaración SOLVE, segundo, al final de la compilación aparece un corto texto explicativo del tipo de error cometido en el modelo. A continuación se presenta un ejemplo de un error que comúnmente se suele presentar al ejecutar el modelo.

```
OBJ..                  FO =E=  prod((I,J), C(I,J)*X(I,J));
R1(J)$(ord(J) gt 1)..   prod(I, X(I,J)) =E= 1;
R2..                    prod(I, X(I,'0')) =E= K;
R3..                    prod(J, X('0',J)) =E= K;


MODEL VRP1 /all/;
SOLVE VRP1 USING MIP MINIMIZING FO;
****                                    $59,256
**** The following MIP errors were detected in model VRP1:
****  59 equation OBJ .. VAR prod smin smax
****  59 equation R1 .. VAR prod smin smax
```

```
****   59 equation R2 .. VAR prod smin smax
****   59 equation R3 .. VAR prod smin smax
  65   DISPLAY X.L;
```

El error 59 que aparece al ejecutar el modelo está indicando que no se puede resolver por MIP, recomienda usar DNLP.

4.4 ERRORES DE EJECUCIÓN

La principal causa por la que se cometen los errores de ejecución se debe al mal uso de las operaciones aritméticas como la división por cero, para evitar esto, se recomienda revisar el capítulo 2 del manual que contiene información acerca comandos para el uso adecuado de las operaciones matemáticas en el lenguaje de modelado GAMS.

Vale la pena recordar que GMAS no ejecuta ni guarda un modelo que presente cualquier tipo de errores, es por eso que el modelador puede durar mucho tiempo perfeccionando el modelo y corrigiendo cualquier tipo de error que se presente durante la compilación y ejecución del mismo.

5 PROBLEMAS MODELADOS EN GAMS

Durante este capítulo se presentaran diferentes tipos de problemas haciendo uso de la teoría antes expuesta, los ejemplos presentados provienen de fuentes como la librería de GAMS, manuales descargados de internet y propios de las autoras. Con el fin de mantener la estructura, se presentarán los ejemplos de la misma manera como se muestran en el programa.

5.1 EJEMPLO 1 - PROBLEMA DE TRANSPORTE Y LOCALIZACIÓN

Problema modelado por las autoras y ejemplo para la redacción del manual. La explicación completa se encuentra en el numeral 1.1 del documento.

* Modelo de localización

```
SETS   i Posible ubicación /BOGOTA, BMANGA, CALI, CARACAS/
       j Clientes /BOGOTA, BRASILIA,MEXICO,CARACAS,LIMA/
       k plantas a ubicar /PLANTA1, PLANTA2/;

TABLE C(i,j) Costos de transporte

             BOGOTA   BRASILIA  MEXICO   CARACAS  LIMA

BOGOTA        0        140       170       90      100

BMANGA       50        150       160       70      110

CALI         60        120       180       100     90

CARACAS      90        130       150        0      80;

VARIABLES            fo,  x(i,j),  y(k,i);
POSITIVE VARIABLES x;
BINARY VARIABLES     y;

PARAMETER b(j) Demanda de los clientes
         /BOGOTA      70000
          BRASILIA    85000
          MEXICO      100000
          CARACAS     20000
          LIMA        60000/;

PARAMETER a(k) Capacidad de las plantas
          /PLANTA1  200000
           PLANTA2  135000/;

EQUATIONS
OBJ  Minimizar el costo total asociado con el transporte
CAPACIDAD(i)    Asociada a la capacidad
DEMANDA(j)      Asociada a la demanda
ALTERNATIVA(k) Asociada a la alternativa de ubicación;

OBJ..             fo =E= SUM((i,j), C(i,j)*x(i,j));
CAPACIDAD(i)..    SUM(j,x(i,j)) =L= SUM(k,a(k)*y(k,i));
DEMANDA(j)..      SUM(i,x(i,j)) =G= b(j);
ALTERNATIVA(k).. SUM(i,y(k,i)) =E= 1;

MODEL Localizacion /ALL/;
SOLVE Localizacion USING MIP MINIMIZING fo;
DISPLAY x.l, y.l;
```

5.2 EJEMPLO 2 - PROBLEMA DE RUTEO DE VEHÍCULOS (VRP)

El ejemplo 1 muestra un problema de ruteo de vehículos con capacidad obtenido de la librería de GAMS y modificado por las autoras de este libro. No se enuncia el problema ni se hacen las respectivas aclaraciones porque GAMS permite describir cada ítem dentro del modelo haciendo uso de comentarios.

```
* PROBLEMA DE RUTEO DE VEHÍCULOS CON CAPACIDAD

$ONTEXT
DESCRIPCIÓN: UNA EMPRESA DE DISTRIBUCIÓN TIENE QUE ENTREGAR A SUS
CLIENTES UNA DEMANDA CONOCIDA EN TON Y LA DISTANCIA ENTRE LE EMPRESA Y
LOS CLIENTES ESTA DADA EN KMS, LOS DATOS SE GENERARON ALEATORIAMENTE
USANDO LA HERRAMIENA EXCEL.  LA EMPRESA USA CAMIONES DE CARGA CON UNA
CAPACIDAD DE 40 TON.  DETERMINAR LAS RUTAS OPTIMAS QUE MINIMICEN EL COSTO
GLOBAL DE TRANSPORTE.
$OFFTEXT

$EOLCOM //

OPTION OPTCR=0;
OPTION LIMROW=3;
OPTION LIMCOL=3;

SET I SITIOS /1*7/;
SET J(I) CLIENTES /2*7/;
ALIAS (I,P);

*DEMANDAS
PARAMETER DEM(I)  DEMANDA DE CADA CLIENTE ;
DEM['2'] =   25 ;
DEM['3'] =   17 ;
DEM['4'] =   15 ;
DEM['5'] =   11 ;
DEM['6'] =   13 ;
DEM['7'] =   9  ;

*DISTANCIAS ENTRE CIUDADES

TABLE C(I,P) MATRIZ DE DISTANCIAS

      1       2       3       4       5       6       7
1     1000    58      54      70      23      82      51
```

```
2    58     1000    89      50      57      22      97
3    52     88      1000    73      89      73      14
4    70     50      73      1000    33      28      26
5    25     57      89      33      1000    18      82
6    82     23      72      28      20      1000    34
7    50     97      14      25      82      34      1000;

*CAPACIDAD DE LOS VEHÍUCLOS
SCALAR  CAP  CAPACIDAD  /40/;  //TODOS  LOS  VEHICULOS  TIENEN  IGUAL
CAPACIDAD
SCALAR K VEHICULOS /3/;     // VEHÍCULOS DISPONIBLES


VARIABLE Q,  // CANTIDAD ENTREGADA HASTA EL CLIENTE I
         F;  // FUNCIÓN OBJETIVO

BINARY VARIABLE X(I,P); //1 SI I PERTENECE A LA SOLUCIÓN 0 EN OTRO CASO


EQUATIONS LLEGAUNAVEZ(I)
          SALEUNAVEZ(I)
          QPRIMERACIUDAD(I)
          QOTRASCIUDADES(I,P)
          FOBJ  ;

* FUNCIÓN OBJETIVO

FOBJ..  F =E=  SUM((I,P), C(I,P)*X(I,P));

* ENTRA Y SALE DE CADA CIUDAD UNA VEZ (EXCEPTO EL DEPÓSITO)

LLEGAUNAVEZ(P)$(ord(P) gt 1)..       SUM(I, X(I,P)) =E= 1;
SALEUNAVEZ(I)$(ord(I) gt 1)..        SUM(P, X(I,P)) =E= 1 ;

*SI J ES EL PRIMER CLIENTE DE UN TOUR, ENTONCES Q(I) = DEM(I)

QPRIMERACIUDAD(I)$J(I)..   Q[I] =l= CAP + (DEM[I]-CAP)*X['1',I] ;

*SI J VIENE JUSTO DESPUÉS DE I EN UN TOUR, ENTONCES Q(J) ES MAYOR *QUE LA
CANTIDAD ENTREGADA DURANTE EL TOUR A I MAS LA CANTIDAD A *SER ENTREGADA A
J.

QOTRASCIUDADES(I,P)$((J(I))  AND  (J(P)) AND  (ORD(I) ne ORD(P)))..
    Q[P]  =g=  Q[I]  +  DEM[P]   -  CAP  +  CAP*X[I,P]  +  (CAP-DEM[P]  -
DEM[I])*X[P,I] ;

Q.up[J] = CAP ;
```

```
Q.lo[J] = DEM[J] ;

* RESOLVER EL PROBLEMA
MODEL CVRP / all / ;

SOLVE CVRP USING MIP MINIMIZING F;

DISPLAY X.l;
```

BIBLIOGRAFÍA

[1] Bruce A. McCarl. McCarl Expanded GAMS User Guide. Version 23.3. Profesor de Economía Agrícola Texas A&M University. 2009.

[2] Richard E. Rosenthal. GAMS - A User's Guide. GAMS Development Corporation, Washington, DC, USA. 2008.

Referencias en Internet

[1] CEPLEX 12 http://www.gams.com/dd/docs/solvers/cplex.pdf [Citado 27 de noviembre de 2009].

[2] Capítulo 14: GAMS http://www.uv.es/~sala/gams/14.PDF [Citado 8 de noviembre de 2008].

[3] Programacion lineal entera con gams http://www.uv.es/~sala/gams/entera.PDF [Citado 8 de noviembre de 2008].

[4] Modelos matemáticos de optimización http://www.gams.com/docs/contributed/modeladoengams.pdf [Citado 31 de octubre de 2008].

[5] http://www.gams.com