CIATEC

REDISEÑO DE LAS RUTAS DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS EN EL MUNICIPIO DE LEÓN, GUANAJUATO, MÉXICO. RECORRIDO MÍNIMO AG-TPS-GOOGLE

Tesis

QUE PARA OBTENER EL GRADO ACADEMICO DE

Maestro en Ciencia y Tecnología en la Especialidad de Ingeniería Industrial y de Manufactura

PRESENTA

JOSÉ LUIS RAMOS GONZÁLEZ

Director

Dr. Javier Yáñez Mendiola

Co director

Dr. Luis Ernesto Mancilla Espinosa

León, Guanajuato, México, Diciembre del 2015

POSGRADO INTERINSTITUCIONAL

Resumen.

A pesar de que la recolección de residuos sólidos urbanos es una función gubernamental muy importante desde los puntos de vista económico, social y ambiental, tradicionalmente las rutas de recolección han sido diseñadas de manera intuitiva por parte de quienes tienen a cargo esa responsabilidad.

Tomando en cuenta aspectos importantes como el modelado del sistema y la optimización del mismo mediante técnicas científicas e ingenieriles que permitan tener rutas de recolección más eficientes, en este trabajo se tiene como objetivo proponer y aplicar una fusión de herramientas con el fin de rediseñar las rutas de recolección. El problema se abordó con el enfoque general para el tratamiento de los problemas de rutas de transporte conocido como el Problema del Agente Viajero (TSP por sus siglas en inglés). Se utilizó la técnica metaheurística Algoritmos Genéticos y otros conceptos de Programación Matemática como es el caso de los grafos. Estos elementos se complementaron con fuentes de información y herramientas informáticas disponibles a cualquier usuario, Google Maps y su herramienta Daft Logic para medir distancias. Se desarrolló un programa de Algoritmos Genéticos en MATLAB. Con base en estos elementos se creó una metodología que se propone para el rediseño de rutas. Se aplicó esta metodología a dos zonas del municipio, gracias a lo cual se determinó que el programa funciona bien para redes pequeñas y medianas, sin embargo al manejar redes de mayores dimensiones se detectó que el programa tarda excesivamente en la generación de la población inicial. Se concluyó que esto es debido a que redes de este tipo generan matrices dispersas por lo cual es muy fácil caer en encasillamientos antes de lograr conformar los individuos completos correspondientes a la población inicial. A la vez, estas redes involucran un gran número de caminos diferentes. Se visualizó como alternativa para mejorar el programa, una modificación a la subrutina de generación de los individuos, mediante un procedimiento iterativo híbrido para la agregación determinística y

aleatoria de los nodos durante la conformación de los cromosomas de la población inicial.

Se comprobó el correcto funcionamiento tanto del algoritmo genético como del programa en el que fue implementado, aplicándolo a una serie de instancias del TSP clásico con matrices densas. Con los resultados que se tuvieron, se encontró que el programa AG trabaja bien, tanto con redes dispersas, como con redes que involucran matrices densas, ya que normalmente realiza la optimización en forma correcta y ágil, convergiendo además a un solo individuo. Se puede afirmar que tanto la metaheurística genética como el programa AG funcionan correctamente, tanto en la integración de los recorridos de los individuos de la población inicial como en la etapa pura de optimización debida al Algoritmo Genético, siendo una limitante el tamaño de la red.

Se contempla un cambio de estrategia para el tratamiento de este problema, mediante la aplicación del enfoque del Problema de Ruteo de Arcos (o ARP, por sus siglas en inglés).

Palabras clave:

Recolección de Residuos Sólidos Urbanos, Algoritmos Genéticos, TSP, Google Maps.

Contenido.

Relación de figuras.

Relación de tablas.

Abreviaciones.

AG: Algoritmo(s) Genético(s).

CARP (Capacitated Arc Routing Problem): Problema de Ruteo de Arcos con Capacidad Definida.

CVRP (Capacitated Vehicule Routing Problem): Problema de Ruteo de Vehículos con Capacidad Definida.

DAFT LOGIC: Herramienta interactiva para medir distancias entre puntos de mapas de Google Maps.

ECARP (Extended Capacitated Arc Routing Problem): Problema Extendido de Ruteo de Arcos con Capacidad Definida.

PLE (ILP por sus siglas en inglés): Programación Lineal Entera.

MATLAB: Software matemático de la compañía Mathworks.

PM: Programación Matemática.

QSB: Software comercial que comprende un sistema de ayuda a la toma de decisiones con herramientas para resolver distintos tipos de problemas en el campo de la Investigación de Operaciones.

RPP (Rural Postman Problem): Problema del Cartero Rural.

RSU: Residuos sólidos urbanos.

SIAP-LEON: Sistema Integral del Aseo Público de León, Guanajuato, México.

tons: toneladas.

TSP (Traveler Salesman Problem): Problema del Agente Viajero.

UMAIP: Unidad Municipal de Acceso a la Información Pública del Municipio de León, Guanajuato, México.

VRP (Vehicule Routing Problem): Problema de Ruteo de Vehículos.

Símbolos.

A : matriz de adyacencias que representa el conjunto de aristas.

c : costo (distancia entre nodos).

E : conjunto de aristas.

G : grafo.

m : metros.

n : número de nodos.

V : vértice o nodo.

x_{ij} : variable binaria para indicar si una arista forma parte de un recorrido.

Glosario.

Arista: es la conexión incidente en dos nodos distintos.

Arista auxiliar: es la arista virtual que representa sub-recorridos que comprenden tramos de aristas no adyacentes y que son necesarios para cerrar un ciclo y terminar el algoritmo de ejecución del programa AG. Se calculan aplicando el criterio del camino más corto y sus distancias totales entre los nodos extremos se introducen en la matriz de datos para efectuar una nueva corrida del programa.

Convergencia: es la situación en la que se presenta el mismo recorrido para todos los individuos o cromosomas de una generación (dos o más de estos individuos pueden iniciar en nodos diferentes, lo cual no es relevante si el recorrido es el mismo). Se puede emplear como condición de terminación del programa.

Corrida inicial: Es la primera de dos fases de ejecución del programa AG. En esta primera fase se genera una propuesta de recorrido que incluye una (o más) aristas no existentes que están representadas en la matriz de adyacencias con valores muy altos *9999's.* Estas aristas no existentes dan origen a otras tantas aristas auxiliares.

Corrida final: Es la segunda fase de ejecución del programa AG en la que se emplean aristas auxiliares cuyas distancias totales se introducen en la matriz de datos para efectuar una nueva corrida del programa, en la que se busca que no hayan tramos con valores muy altos *9999.*

Matriz de adyacencias: es la representación matricial del conjunto de nodos de la red, dada por valores en los elementos respectivos de la matriz que involucran conexiones entre dos nodos distintos.

Matriz de adyacencias ponderada: es el resultado de la conversión de la matriz de adyacencias unitaria, en la que han sido sustituidos los *(1's)* por los valores de las distancias de las aristas entre nodos adyacentes, y el relleno de valores *9999's* en los elementos en los que no existe adyacencia entre nodos.

Matriz de adyacencias unitaria: Es la matriz de adyacencias en la que se han capturado *(1's)* en los elementos de la matriz correspondientes a los nodos donde existe conexión entre nodos consecutivos

Nodo (vértice): es un punto de una red dada. Representa el cruce de dos o más calles.

Recorrido Mínimo AG-TSP-Google: Metodología desarrollada dentro del proyecto Rediseño de las rutas de recolección de residuos sólidos urbanos en el municipio de León, Guanajuato, México, la cual aplica como técnica de solución la heurística Algoritmos Genéticos (AG) bajo el enfoque del Problema del Agente Viajero (TSP) y considera como fuente de datos los mapas de Google Maps.

Problemas NP: son aquellos problemas para los que no se conoce un algoritmo polinomial de resolución (Díaz, A. & Tseng, F.T., 1996)

Valor muy alto: es un valor de *9999* utilizado en el programa AG para el manejo de los casos en los que no existe adyacencia entre nodos. Se agrega, en los elementos respectivos de la matriz de adyacencia.

Capítulo 1. Planteamiento del problema.

1.1. Introducción.

De acuerdo con información oficial, se generan un promedio de 991 toneladas de residuos sólidos urbanos (RSU) diariamente en el Municipio de León, Guanajuato (SIAP León, 2014). El SIAP-LEON es un organismo público descentralizado de la administración municipal denominado “Sistema Integral del Aseo Público de León, Guanajuato” el cual tiene por objeto, entre otros, la prestación del servicio de recolección domiciliaria que involucra la recolección, traslado y disposición final de RSU en el municipio. Este servicio lo tiene concesionado a empresas privadas (SIAP León, 2014).

Se estableció como objetivo del proyecto proponer y aplicar una fusión de herramientas para la solución del problema del diseño de las rutas de recolección, con el fin de rediseñarlas.

Al revisar la literatura y herramientas que se han empleado para caracterizar y resolver el problema de la recolección de residuos sólidos municipales el problema se abordó con el enfoque general para el tratamiento de los problemas de rutas de transporte conocido como el Problema del Agente Viajero (TSP por sus siglas en inglés).

En la literatura se encontraron abundantes ejemplos de aplicaciones de los Algoritmos Genéticos (AG) con resultados positivos. Se eligió esta técnica metaheurística para ser aplicada en este proyecto, por lo que se desarrolló un programa en el software matemático MATLAB.

Adicionalmente se utilizaron conceptos de Programación Matemática como es el caso de los grafos, aprovechando su capacidad de representar la topología de redes correspondientes a problemas de optimización de rutas. Estos elementos se complementaron con fuentes de información y herramientas informáticas disponibles a cualquier usuario tales como Google Maps y su herramienta Daft Logic para medir distancias.

Con base en los elementos anteriores se desarrolló una metodología nombrada Recorrido Mínimo AG-TSP-Google y que se propone para el rediseño de rutas.

1.2. Descripción del problema.

Las rutas de recolección de residuos sólidos urbanos en el Municipio de León, Guanajuato, tradicionalmente han sido diseñadas de manera intuitiva por parte de quienes tienen a su cargo esa responsabilidad sin tomar en consideración aspectos importantes como el modelado del sistema y la optimización del mismo mediante técnicas científicas e ingenieriles que permitan tener rutas de recolección mas eficientes.

1.2.1. Características del modelo actual.

El modelo actual de las rutas de recolección de RSU, presenta las siguientes características principales (SIAP León, 2014).

- Actualmente se tiene dividida el área urbana del municipio en siete sectores.
- En cuatro de los sectores la frecuencia de recolección es tres veces por semana, en dos de ellos seis veces por semana y en uno la frecuencia es diaria.

- Los vehículos recolectan sólo los residuos sólidos urbanos (RSU) generados en casas-habitación como resultado de las actividades domésticas.
- La recolección se realiza en forma continua y en puntos establecidos.
- El grupo de recolección hace el recorrido una vez al día durante el cual se recogen todos los residuos sólidos, respetando el sentido de las calles así como los señalamientos viales.
- Cifras de tonelajes de recolección de residuos sólidos para el período Ene-Ago'13 indican que se generaron un promedio de 926 toneladas diarias de RSU correspondientes a 120 rutas concesionadas y 12 propias del SIAP (SIAP León, 2013).
- De acuerdo con recolectores consultados se recabó el dato de que los recorridos pueden abarcar un lapso de entre siete y ocho horas por jornada. En ocasiones es necesario realizar dos viajes al relleno sanitario para la descarga de los residuos.

Las cifras presentadas indican que la magnitud de los volúmenes de RSU manejados es un tema importante.

1.2.2. Variables importantes.

Se identifican las siguientes variables importantes en el estudio de este problema.

- Número de calles.
- Sentido y restricciones de vialidades.
- Número de cruces de calles.
- Distancia entre cada cruce de calles.

Aunque los tiempos de recorrido pueden ser una variable importante, en este trabajo se adoptó a la distancia total de recorrido como criterio a optimizar, considerando que en última instancia el tiempo total de recorrido resulta ser un efecto de la distancia.

1.3. Estado del arte.

El estado del arte analizado ha sido dividido en dos secciones. La primera se desarrolló enfocada en la búsqueda en bases de datos de publicaciones científicas y académicas sobre las técnicas más recientes que se han aplicado en la solución de problemas de ruteo de vehículos en general y de las relacionadas con la recolección de residuos sólidos en particular. En la segunda parte se hace una breve reseña de antecedentes referentes al Problema del Agente Viajero (TSP, por sus siglas en inglés).

1.3.1 Consulta a base de datos de citas bibliográficas.

Mediante consultas preliminares a las bases de datos sobre uno de los casos más representativos del tema de ruteo de vehículos que es el Ruteo de Vehículos con Capacidad Definida (CVRP por sus siglas en inglés), se detectó que por sobre los métodos exactos y los heurísticos, los procedimientos metaheurísticos se emplean con mayor frecuencia (ver tabla 1).

Tabla 1. Resumen consulta a base de datos de citas bibliográficas. Artículos publicados últimos cinco años 2009-2014. Tópico CVRP. (Nov14).

Tipo de método	No. de artículos	%
Exactos	16	17.8
Heurísticos	22	24.4
Metaheurísticos	52	57.8
Total	90	100

De los 52 casos estudiados mediante técnicas metaheurísticas, siete correspondieron a AG's, siete a Búsqueda Tabú, ocho a Colonias de Hormigas, siete a Enjambres de Partículas, cuatro a Recocido Simulado y el resto a otros.

Adicionalmente, se revisó la literatura sobre el tópico Recolección de residuos sólidos. El análisis del tema permitió ubicar artículos de utilidad con las siguientes observaciones:

Lacomme, Prins & Ramdane-Cherif (2001)**,** mencionan que los algoritmos exactos están aún limitados para problemas pequeños y que los metaheurísticos son requeridos para instancias de gran escala. Publican el primer algoritmo genético (AG) para el Problema de Ruteo de Arcos con Capacidad Definida (CARP, por sus siglas en inglés) planteando un híbrido que ataca extensiones realistas como vueltas prohibidas y grafos mixtos.

Posteriormente, continuando en la misma línea, Lacomme, Prins & Sevaux, (2003), buscan minimizar la longitud del viaje más largo además de la longitud de la ruta total (el único criterio minimizado en el problema académico). Presentan un AG bi-objetivo para este CARP más realista nunca estudiado en la literatura hasta ese momento. Sobre esta base incluyen dos características clave: uso de heurísticas constructivas para sembrar la población inicial y la hibridación con un procedimiento de búsqueda local.

Viotti, Polettini & Pomi (2003), argumentan que hasta la fecha, las rutas óptimas se han desarrollado de acuerdo con metodologías intuitivas y experiencia de campo, y que para analizar estas complejidades se usan aplicaciones que normalmente se basan en procedimientos heurísticos que permiten soluciones de alta calidad pero que en el tema computacional tienen una complejidad limitante por la calidad de las soluciones calculadas, y para la representación exacta de las zonas urbanas. Para resolver el problema emplean una metodología alternativa basada en un AG.

Zhu, Xia & Yang (2008), presentan un AG mejorado para resolver una versión extendida del CARP (ECARP) con vueltas prohibidas y problemas de encharcamientos. El nuevo algoritmo mejora la estructura de la población y el modo de organización del cromosoma del AG tradicional, elaborando operadores evolutivos simples y de alta eficacia que evitan el fenómeno de la convergencia prematura. La mejora propuesta puede resolver el ECARP eficazmente, y la

comparación entre la mejora del AG y un algoritmo memético clásico (el cual como los AG también es un tipo de algoritmo evolutivo), muestra que este algoritmo es más eficaz y puede conseguir mejores resultados en la resolución del CARP básico de gran tamaño.

Por su parte, Bonomo, Durán & Larumbe (2012), proponen un método que utiliza técnicas de investigación de operaciones para optimizar las rutas de vehículos de recolección de residuos de contenedores. El problema se reduce al TSP clásico. Además del criterio de distancia mínima típico, el problema de optimización incorpora el objetivo de reducir el uso y desgaste del vehículo. El enfoque de solución, emplea la teoría de grafos y herramientas de programación matemática. También discuten el proceso de corrección de datos.

Dado que en la literatura sobre el tema de ruteo de vehículos son abundantes los ejemplos, con resultados positivos, de aplicaciones de los AG's en los casos en que el recorrido implica un número elevado de puntos de visita y tomando en cuenta que el SIAP en los inicios del proyecto contemplaba el concepto de macrorutas para fusionar algunas de las rutas, se seleccionó esta técnica metaheurística para ser aplicada en este proyecto.

1.3.2 Antecedentes del Problema del Agente Viajero (TSP).

El TSP ha sido uno de los problemas más estudiados en Investigación de Operaciones. En 1972, Karp probó que pertenece a la clase de problemas difíciles NP-Hard (Martí, 2001).

Existe el antecedente remoto del problema de la exploración de los grafos de Euler, en el que cada nodo tiene el mismo número de aristas entrantes y salientes, mismo que fue resuelto con un algoritmo creado por Hierholzer en 1873 (Russell & Norvig, 2010).

En el siglo XIX, problemas matemáticos relacionados con el TSP fueron tratados por el matemático irlandés William **Hamilton** y el matemático inglés Thomas

Kirkman. La forma general del TSP parece haber sido estudiada por primera vez por el matemático K. Menger en Viena y Harvard en los 1930's (U. Waterloo 2015)

A principios de los 50's del siglo pasado se inicia la Teoría de Flujos en Redes, relacionándose con la Teoría de Grafos mediante los trabajos de Ford & Fulkerson en 1954. La Programación Entera se inicia con Gomory con sus trabajos sobre cortes automáticos publicados en 1958 (Salazar, 2001).

Dantzig, Fulkerson, y Johnson, publican uno de los primeros artículos sobre el TSP que es un referente, pues muchas ideas son usadas para resolver problemas de PLE. Reportan la solución de un TSP de 49 ciudades (Univ. Waterloo, 2015).

En la línea del modelado matemático, algunos de los métodos exactos que han sido utilizados son los algoritmos de Ramificación y Acotamiento (Laporte et al. 1987), Ramificación y Corte (Lysgaard et al. 2004), Programación Dinámica (Magnanti 1981) y el procedimiento de Relajación Lagrangiana (Stewart & Golden 1984) (Anbuudayasankar, Ganesh, Mohapatra, 2014).

Dentro de los enfoques heurísticos, los procedimientos constructivos de ruta han manejado el procedimiento del Vecino más Cercano (Rosenkrantz et al. 1977), el procedimiento de Ahorros (Clarke & Wright 1964), los procedimientos de Inserción y el enfoque de Árbol de Expansión Mínima (Christofides 1976). Los procedimientos de mejora de ruta generalmente utilizan el enfoque de Intercambio de Aristas, tal como el procedimiento k-opt que fue presentado por Lin & Kernighan (1973). Por su parte, Ball (2011) ha hecho un amplio estudio sobre la heurística basada en modelos y métodos de PM (Anbuudayasankar et al., 2014).

Ball (2011) al realizar un estudio de las heurísticas que hacen uso de los métodos y modelos basados en Programación Matemática (PM), menciona que ésta envuelve el estudio de técnicas que pueden generar soluciones óptimas demostrables a problemas de optimización. Desde un punto de vista práctico, las heurísticas tienen un objetivo muy similar, es decir generar "soluciones" a problemas de optimización. La diferencia es que las soluciones deberían ser

“buenas” pero no necesariamente óptimas demostrables. En muchas aplicaciones prácticas la distinción puede ser menor. Concluye que cuándo un modelo de PM es empleado en el estudio de un problema real, ya sea que se llame método heurístico o exacto, se trata de una cuestión de interpretación. Por tanto, hasta cierto punto, cualquier investigación que involucre la aplicación de la PM, contribuye al estudio de las heurísticas basadas en PM.

Los métodos metaheurísticos se pueden clasificar como de poca memoria y basados en memoria. En 1993 Osman propuso un algoritmo de Recocido Simulado el cual es uno de los metaheurísticos más importantes que emplea poca memoria. Posteriormente, como uno de sus principales expositores, Tarantilis presentó variantes (2001, 2002 y 2004) (Anbuudayasankar et al., 2014).

Dentro de los metaheurísticos basados en memoria Anbuudayasankar et al. (2014) comentan que el término memoria fue utilizado de manera explícita por los algoritmos de Búsqueda Tabú (TS) (Glover 1989) y los Algoritmos Basados en Memoria Adaptativa (AMBA) (Rochat & Taillard 1995). Sin embargo, otros meta-heurísticos como los Algoritmos Genético**s** (Holland1975), Colonia de Hormigas (Dorigo 1992), Enjambre de Partículas (Kennedy & Eberhart 1995) y los Algoritmos Meméticos (Moscato 1989), usan la mecánica y estructuras que pueden ser considerados como recuerdos.

Anbuudayasankar et al. (2014), en la revisión que hacen sobre el TSP mencionan que los AG’s fueron explorados por Ulder et al. (1991), que Cotta et al. (1995) lo hibridaron con técnicas Branch and Bound, Larranaga et al. (1999) usaron un AG híbrido y Marinakis et al. (2007) probaron un método de solución utilizando AG‘s.

1.4. Justificación.

De acuerdo con información oficial, se generan un promedio de 991 toneladas de RSU diariamente en la Ciudad de León, Guanajuato (SIAP León, 2014).

El desarrollo económico, los cambios en los patrones de consumo y el crecimiento de la mancha urbana, entre otros factores, imponen serios retos a los gobiernos municipales en lo que respecta a la prestación de servicios públicos, en particular en los sistemas de gestión de los RSU. El componente más importante en estos sistemas de gestión es el que corresponde a la recolección de los residuos, de ahí que un mal diseño de las rutas impacte en mayores **tiempos y costos** y contribuya a que no se cumpla con las expectativas del servicio requerido.

1.5. Objetivos.

1.5.1 Objetivo general.

Proponer y aplicar la técnica metaheurística de solución Algoritmos Genéticos, y las fuentes y herramientas de Google Maps, al problema de diseño de las rutas de recolección de residuos sólidos urbanos, con el fin de rediseñarlas, bajo el enfoque del TSP.

1.5.2 Objetivos específicos.

1. Modelar el diseño actual de las rutas de recolección de RSU para conocer sus características.
2. Determinar una técnica de solución para el rediseño de las rutas de recolección de RSU.
3. Analizar el rediseño para mejorar el sistema de rutas.

1.6. Hipótesis.

El Recorrido Mínimo AG-TSP–Google minimiza la distancia para la recolección de los residuos sólidos, con respecto a las rutas que se tienen actualmente.

Capítulo 2. Marco teórico.

2.1 Programación Matemática.

La Programación Matemática (PM) es una parte de las Matemáticas Aplicadas que trata de resolver problemas de decisión en los que se deben determinar acciones que optimicen un determinado objetivo satisfaciendo limitaciones en los recursos disponibles (Salazar, 2001).

La finalidad de la PM es resolver los problemas alcanzando el valor mínimo (o máximo) de un objetivo. Los problemas de este tipo se llaman problemas de optimización. Su estudio nace de un contexto aplicado más amplio: la Investigación de Operaciones. Esta parte de la Ciencia puede ser entendida como un puente entre Realidad y Matemática que pretende resolver problemas de aquélla con técnicas de ésta. Este enlace no es simple y directo. La clave es el concepto de modelo (matemático). Los modelos son representaciones simplificadas de sistemas reales, que tratan de destacar las relaciones principales entre ciertos objetos. Los elementos controlables en la realidad se traducen en variables en el modelo, y mediante ecuaciones e inecuaciones se modelan las limitaciones de recursos. Algunos de estos modelos proceden directamente de la realidad, pero muchos otros son planteados por otras ramas, como son los Grafos, la Teoría de Juegos, y otras (Salazar, 2001, pp. 3-5, 7).

La PM se divide en varias ramas según las características de las variables y de las ecuaciones o inecuaciones que describen los modelos matemáticos. En particular, cuando estas expresiones matemáticas son lineales en sus variables, se habla de Programación Lineal; y cuando sus variables asumen sólo valores enteros se trata de Programación Entera, que son campos que se abordan en este trabajo para

conocer el tipo de resultados que se obtienen al resolver el TSP con ellos (Salazar, 2001, p.7).

En esta investigación, con la aplicación de los Algoritmos Genéticos se busca asegurar que los resultados que se obtengan proporcionen soluciones de variables enteras.

Otra rama de interés es la Optimización Combinatoria que busca resolver problemas caracterizados por tener un número finito, aunque muy grande, de posibles soluciones (Salazar, 2001, p. 8).

Díaz, A. & Tseng, F.T. (1996) mencionan al respecto que:

> "En los problemas de tipo combinatorio existe siempre un procedimiento elemental para determinar la solución óptima buscada: realizar una explosión exhaustiva del conjunto de soluciones (enumeración completa). Es decir, generar todas las soluciones factibles (o sea, las que satisfacen las restricciones), calcular para cada una el costo asociado y elegir finalmente la que haya dado lugar al mejor de ellos. … aunque este método teóricamente lleva siempre a la solución óptima buscada, no es eficiente, pues el tiempo de cálculo necesario crece exponencialmente con el número de ítems del problema… Por tanto, es patente que existen problemas combinatorios para los que no se conocen algoritmos combinatorios de resolución más que aquellos en los cuales se produce una explosión del tiempo de cálculo (exponencialmente) al aumentar el tamaño del problema. Son problemas computacionalmente difíciles de tratar" (p.22).

Los mismos autores, Díaz, A. & Tseng, F.T. (1996) profundizan en la clasificación de este tipo de problemas:

> "Matemáticamente se trató de caracterizar…(este) tipo de problemas…se comprobó que pertenecen a (la clase) denominada NP, en la cual están incluidos aquellos problemas para los que no se conoce un algoritmo polinomial de resolución, aunque sí sea posible, dada una solución, comprobar en tiempo polinomial si su coste es mejor que un determinado valor,…en 1971 Cook

> demostró que hay problemas en NP que son “especialmente difíciles”. Son los denominados NP-completos…aquellos que además son NP-difíciles (NP-Hard), es decir tienen la peculiaridad de que todos los problemas en NP pueden ser “reducidos” polinomialmente a ellos, o lo que es lo mismo, que si se puede dar una solución en tiempo polinomial para uno de ellos, se podría dar para todos los de NP” (p.23).

En este sentido se deriva de manera natural la necesidad y justificación de emplear procedimientos heurísticos y/o metaheurísticos para resolver el tipo de problemas en los que el número de nodos sea elevado.

Varios problemas de optimización pueden ser modelados mediante unas herramientas de abstracción alternativas a los modelos matemáticos basados en variables e inecuaciones. Estas herramientas son los grafos, los cuales se emplean para describir algoritmos que aprovechan la estructura combinatoria de los problemas (Salazar, 2001, p.259).

Formalmente, un grafo se define como una pareja $G = (V, A)$, donde V es un conjunto de puntos llamados vértices o nodos, y A es un conjunto de pares de nodos distintos, llamados aristas. A su vez, A se puede definir como una matriz de adyacencias entre nodos.

El término grafo se refiere tanto a la definición matemática anterior, como a la representación gráfica de una red. En general, permiten representar la topología de redes, y en particular, las correspondientes a problemas de optimización de rutas. En combinación con otras herramientas de PM es posible utilizarlos para la resolución de problemas.

En este trabajo los grafos se emplean para representar gráficamente las zonas de recolección, así como las rutas de las soluciones encontradas.

Las anteriores ideas llevan a utilizar diferentes técnicas de solución bajo el tema de lo que se conoce como híbridos, mediante los cuales se emplean herramientas

o conceptos de PM, relacionados con procedimientos heurísticos o metaheurísticos para solucionar el tipo de problemas que nos ocupa.

2.2. Métodos exactos.

Dentro de los métodos exactos para resolver problemas de Optimización Combinatoria se distinguen los analíticos y los no analíticos; ejemplos de los analíticos son el método simplex para modelos de Programación Lineal, los Planos de Corte y la Programación Dinámica, los Grafos y Árboles. Entre los no analíticos destacan los basados en la enumeración de soluciones como los de la Programación Entera (Algoritmos de Ramificación y Acotamiento, y de Ramificación y Corte) (Salazar, 2001).

Dada su dificultad computacional, generalmente los Métodos Exactos se emplean para resolver problemas con pocos nodos.

2.3. Métodos heurísticos.

Díaz et al. (citados por Martí, 2001) proporcionan la siguiente definición general, “Un método heurístico es un procedimiento para resolver un problema de optimización bien definido mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución.” (p.2).

Salazar (2001) por su parte presenta una definición relacionada con el concepto del algoritmo:

> Se llama algoritmo heurístico (o simplemente heurístico) a cualquier algoritmo que con poco esfuerzo computacional proporcione una solución factible cuyo valor

> objetivo normalmente esté próximo al valor objetivo óptimo del problema original. Las soluciones obtenidas mediante un heurístico se denominan soluciones heurísticas. (p.374)

Martí (2001) aclara que “En contraposición a los métodos exactos que proporcionan una solución óptima del problema, los métodos heurísticos se limitan a proporcionar una buena solución no necesariamente óptima.” (p.2).

Los métodos heurísticos se emplean como parte de un procedimiento global para lograr el óptimo de un problema. Existen dos grandes posibilidades:

- Proporcionan una buena solución inicial de partida.
- Participan en un paso intermedio del procedimiento.

Los algoritmos heurísticos dependen en gran medida del problema concreto para el que se han diseñado. En otros métodos de resolución de propósito general como son los exactos, existe un procedimiento conciso y preestablecido, independientemente del problema abordado. En los métodos heurísticos no es así. Las técnicas e ideas aplicadas a la solución de un problema son específicas de este y aunque, en general, pueden ser trasladadas a otros problemas, se particularizan en cada caso (Martí, 2001).

Existen muchos métodos heurísticos de naturaleza muy diferente. Atendiendo a la manera en que buscan y construyen sus soluciones, Martí (2001) maneja una clasificación de donde se retoman las siguientes categorías más representativas, en donde se pueden ubicar los heurísticos más conocidos:

- Métodos constructivos.- consisten en construir paso a paso la solución del problema. Usualmente son deterministas y están basados en la mejor elección en cada iteración. En este apartado se pueden encontrar los conocidos como: El Vecino más Próximo; los de Inserción; los basados en Árboles Generadores (incluyendo el Algoritmo de Christofides); los basados en Ahorros (también nombrados como de Clarke y Wrigth).

- Métodos de búsqueda (o de mejora) local.- a diferencia de los anteriores comienzan con una solución del problema y la mejoran progresivamente. Se basan en explorar el entorno o vecindad de una solución y quedan determinados al especificar el criterio de selección de una solución dentro del entorno. Se pueden emplear diferentes criterios para seleccionar una nueva solución. Uno de los más simples y usuales es el llamado greedy (o voraz) el cual consiste en tomar la solución con mejor evaluación de la función objetivo, siempre que sea mejor que la anterior. Esto permite ir mejorando la solución mientras sea posible. Aquí se pueden incluir todos los procedimientos denominados de k-Intercambio y el de Lin-Kernighan.

A esta clasificación resulta conveniente agregar una categoría más que conjuga las características de unas técnicas con otras para lograr mejores resultados. La importancia de este grupo consiste en que se pueden considerar como un punto de partida para los métodos metaheurísticos (Martí, 2001):

- Métodos combinados.- surgen de la combinación de los constructivos al proporcionar una solución inicial, a partir de la cual los métodos de búsqueda local tratan de obtener una de mejor valor.
    - Procedimientos aleatorizados. Aplican una modificación al algoritmo de construcción consistente en sustituir una elección greedy por una elección al azar de entre un conjunto de buenos candidatos. De esta forma se evalúan todos los elementos que pueden ser añadidos y se selecciona un subconjunto con los mejores. La elección se realiza al azar sobre ese subconjunto de buenos candidatos. Posteriormente cada solución puede mejorarse con un algoritmo de búsqueda local.
    - Multiarranque. Generalizan el esquema anterior. Tienen dos fases: la primera genera una solución y la segunda en la que normalmente, pero no necesariamente, esta es mejorada. Cada iteración global produce una solución, frecuentemente un óptimo local. La mejor de ellas se toma como la salida del algoritmo. Básicamente se trata de

almacenar información relativa a soluciones ya generadas y utilizarla para la construcción de nuevas soluciones.

2.4. Métodos metaheurísticos.

Los métodos metaheurísticos surgieron con el propósito de obtener mejores resultados que los alcanzados por los heurísticos tradicionales (constructivos y de mejora). El término fue introducido por Fred Glover en 1986 (Martí, 2001, p.5). Osman y Kelly (1989) (citados por Martí, 2001) los definen de la siguiente manera:

> Los procedimientos metaheurísticos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria en los que los heurísticos clásicos no son efectivos. Proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de la inteligencia artificial, la evolución biológica y los mecanismos estadísticos (entre otros). (Martí, 2001, p.5)

Al abundar sobre la importancia de estos métodos Martí (2001) opina que, “Se sitúan conceptualmente por “encima de” los heurísticos en el sentido que guían el diseño de éstos. Así, al enfrentarse a un problema de optimización, podemos escoger cualquiera de estos métodos para diseñar un algoritmo específico que lo resuelva aproximadamente” (p.5).

En particular, para atacar los problemas de recorridos en los que se visita un número elevado de puntos se recomiendan las técnicas metaheurísticas dado que este tipo de problemas encuentran grandes dificultades para ser resueltos con los métodos exactos.

La extensa literatura sobre el tema muestra que ha habido un gran crecimiento y desarrollo de los métodos metaheurísticos. Entre los más conocidos se enlistan aquellos relativamente consolidados y que han probado su eficacia sobre una

colección amplia de problemas. Cada uno de ellos tiene sus propias características específicas.

- Algoritmos Genéticos (AG).
- Búsqueda Tabú.
- Colonias de Hormigas.
- Enjambres de Partículas.
- Recocido Simulado.

En la siguiente sección se describen los Algoritmos Genéticos, relacionándolos con los conceptos del TSP.

2.5. Algoritmos Genéticos (AG) y el Problema del Agente Viajero (TSP).

Para hacer una descripción general de los AG, Díaz & Tseng (1996) comienzan refiriéndose a los procesos evolutivos en la naturaleza:

> …los algoritmos genéticos son “técnicas de búsqueda basadas en la mecánica de la selección natural y la genética” [Goldberg, 1989]. Su estructura se ha diseñado basándose en un intento de abstracción artificial del “algoritmo” de selección propio de la naturaleza, con la esperanza de que así se consigan éxitos similares en relación a la capacidad de adaptación a un amplio número de ambientes diferentes. Es decir, los algoritmos genéticos son flexibles y de ámbito general, y pueden ser aplicados en un amplio número de problemas.
>
> En los organismos biológicos, la información hereditaria es pasada a través de los cromosomas que contienen la información de todos esos factores, es decir, los genes, los cuales a su vez están compuestos por un determinado número de valores (alelos). Varios organismos se agrupan formando una población, y aquellos que mejor se adaptan son los que más probabilidades tienen de sobrevivir y reproducirse. Algunos de los supervivientes son seleccionados por la naturaleza para ser cruzados y así producir una nueva generación de organismos.

> Esporádicamente, los genes de un cromosoma pueden sufrir ligeros cambios (las denominadas mutaciones).
>
> Con esto, los conceptos básicos de los algoritmos genéticos ya pueden ser presentados: por lo general, los alelos son binarios (cero o uno), representando valores de las variables de decisión, que se corresponderán con los genes. Los cromosomas representan pues las soluciones. En realidad, los cromosomas no son más que tiras de bits, y en su forma más simple, un organismo puede estar formado por un simple cromosoma.
>
> Las tres operaciones antes mencionadas (reproducción, cruzamiento y mutación) han sido frecuentemente utilizadas en este tipo de algoritmos. Un algoritmo genético sencillo podría aplicar operaciones de reproducción, cruzamiento y mutación a una población y generar nuevos organismos de modo iterativo (p. 31).

Dentro de los enfoques generales para el tratamiento de los problemas de rutas de transporte se encuentra el Problema del Agente Viajero (TSP por sus siglas en inglés). Es el caso más sencillo de plantear aunque de compleja solución.

En el TSP se dispone de un sólo vehículo y una única estación. En un sólo viaje el viajero debe visitar n ciudades de forma que la distancia recorrida o sus costos c sean mínimos. Se dispone de una matriz de datos que corresponde a las distancias entre puntos o nodos (Maroto, Alcaraz & Ruiz, 2002).

Al hablar de las razones por las cuales el TSP ha recibido especial atención entre la comunidad científica, Martí, R. (2009), menciona que: “Sus soluciones admiten una doble interpretación: mediante grafos y mediante permutaciones, dos herramientas de representación muy habituales en problemas combinatorios por lo que las ideas y estrategias empleadas son, en gran medida, generalizables a otros problemas.” (p.4).

2.5.1. Modelo matemático del TSP.

Retomando la descripción que Díaz & Tseng (1996) hacen de los Algoritmos Genéticos con respecto al proceso de reproducción, mencionan que:

> …se guía a través de una función que mide el grado de adaptación del cromosoma, siendo el proceso de selección del cromosoma por reproducir dependiente de los valores que esa función le asigne: a mayor valor (para maximización, o menor valor para minimización), mayor probabilidad de selección y supervivencia. Los miembros de la nueva generación de cromosomas son de nuevo emparejados aleatoriamente (p. 32).

Este proceso de reproducción que a su vez depende del proceso de selección, enfocado al TSP, se puede implementar mediante la interpretación de un modelo matemático. Para el efecto consideremos el empleado por Maroto et al. (2002, pp.198-199):

-Sea:

Un grafo $G = (V, E)$ con etiquetas (distancias) en los arcos o aristas c_{ij}, y lo que se desea encontrar es un único viaje o "tour" de mínima longitud, o lo que es lo mismo, un ciclo dirigido hamiltoniano (es decir, que visite todos los puntos sólo una vez) y que contenga los n nodos del grafo.

-Variables.

Para cada arista posible del grafo se define una variable,

$x_{ij} = 1$, si la arista (i, j) está en el ciclo,

$x_{ij} = 0$, en caso contrario,

dónde $i, j = 1, \ldots, n$

-Función objetivo.

Minimizar los costos del viaje:

$$Minimizar \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \left(c_{i,j}\, x_{i,j}\right) \quad (1)$$

-Restricciones.

Llegadas: a cada ciudad llega el visitante (cada nodo es visitado):

$$\sum_{i=1, i\neq j}^{n} \left(x_{i,j}\right) = 1, \;\; j = 1, \dots, n \quad (2)$$

Salidas: de cada ciudad parte el visitante (cada nodo es abandonado):

$$\sum_{j=1, j\neq i}^{n} \left(x_{i,j}\right) = 1, \;\; i = 1, \dots, n \quad (3)$$

Complementando la explicación sobre el proceso de los AG, Díaz & Tseng (1996) aclaran en qué consisten las operaciones de cruce o cruzamiento y de mutación.

> Mediante la operación de cruzamiento se intercambian genes entre la pareja. Un modo sencillo de hacerlo consiste en dividir cada una de las dos tiras de bits en dos sub-tiras y luego cambiar los valores de las últimas sub-tiras, obteniendo así dos nuevos cromosomas de una nueva generación. Las mutaciones consisten en alterar un bit de un cromosoma (de 0 a 1 ó viceversa), con una probabilidad relativamente pequeña. Todo este proceso se repite hasta que algún crtiterio de finalización se cumpla (p. 32).

Finalmente, Díaz & Tseng (1996) mencionan algunas características de los AG, de donde destaca el que esta técnica metaheurística genera poblaciones de soluciones en forma aleatoria, y no una solución simple en cada iteración.

> Los algoritmos genéticos, al menos en sus versiones más clásicas, manipulan la información en tiras de bits. Mientras la mayoría de las otras metaheurísticas generan una solución simple en cada iteración, los algoritmos genéticos trabajan

sobre una población de soluciones, generando una nueva en cada iteración. Por lo general son algoritmos neutrales al contexto y no consideran ni explotan la información del problema dentro del proceso de búsqueda. La generación de nuevas soluciones es, pues, aleatoria por naturaleza. Los algoritmos genéticos utilizan una elección al azar para guiar la búsqueda, repitiendo durante un número relativamente largo de iteraciones operaciones muy simples que dan lugar a un volumen masivo de soluciones, dentro de la búsqueda de las que se consideran buenas soluciones (p. 32).

2.5.2. Pseudocódigo del Algoritmo Genético.

El funcionamiento general de un AG, puede ser esquematizado mediante pseudocódigo. Se presenta el propuesto por Araujo & Cervigón (2009, p.22):

```
función Algoritmo_Genético()
{
        TPoblacion pob;         // población
        TParametros parámetros// tamaño población

        obtener_parametros(parametros);
        pob = poblacion_inicial();
        evaluación(pob, tam_pob, pos_mejor, sumadaptacion);

        // bucle de evolución
        mientras no se alcanza condición de terminación hacer
        {
                seleccion(pob, parámetros);
                reproducción(pob, parámetros);
                evaluación(pob, parámetros, pos_mejor, sumadaptacion);
        }
        devolver pob[pos_mejor]
}
```

Normalmente la operación *(obtener_parametros)* incluye el tamaño de la población y la longitud del cromosoma. Las líneas *selección, reproducción y evaluación* representan procesos que se implementan por medio de funciones o subrutinas. El proceso de reproducción considera el cruce y la mutación. [1]

Los mismos autores Araujo & Cervigón (2009) explican el funcionamiento del algoritmo descrito:

> El algoritmo procesa un conjunto de individuos que forman la población *pob.* Al comienzo del algoritmo se obtienen los datos de entrada al problema *(obtener_parametros)* y se genera la población inicial, cuyos individuos se evalúan mediante la función de adaptación del algoritmo. El resto del algoritmo consiste en un bucle, cada una de cuyas iteraciones es una generación en la que se produce un proceso de selección, que da mayores probabilidades de tener copias en la nueva población a los individuos más adaptados, seguido de un proceso de reproducción en el que se generan nuevos individuos a partir de los de la población mediante operaciones de mezcla y pequeñas alteraciones, y finalmente una evaluación de la nueva población. En muchas ocasiones se utilizan pequeñas variantes de este esquema. Así, por ejemplo, a veces se selecciona un subconjunto de la población, que es el único que participa en las operaciones de reproducción. (pp. 22-23)

Este esquema general puede ser aplicable a cualquier algoritmo evolutivo (v.gr. AG's, AG's de codificación real, Programas Evolutivos y otros).

> Siguiendo este esquema general se han desarrollado distintas variantes de algoritmos evolutivos, cuya principal diferencia se encuentra en la representación de los individuos. Lógicamente, los operadores genéticos que se utilizan para la reproducción en cada caso dependen de la representación adoptada. (Araujo & Cervigón, 2009, pp. 23)

---

[1] Para mayor explicación consultar el texto de Araujo & Cervigón (2009), Algoritmos Evolutivos. Un enfoque práctico.

En este sentido, cabe hacer la reflexión de que si bien la representación de los individuos o cromosomas de la población tienen un papel muy importante, evidentemente el problema que se estudie va a dar lugar a un algoritmo en específico.

> …dentro de cada una de esas clases, cada algoritmo es un caso particular, no sólo en función del problema concreto al que se aplica, sino también dependiendo de la elección concreta de los métodos que se aplican en los distintos procesos involucrados, como la selección y la reproducción. (Araujo & Cervigón, 2009, pp. 28).

La conclusión de esto último, es que es necesaria la utilización de un algoritmo orientado al TSP en específico.

Capítulo 3. Desarrollo.

El desarrollo del proyecto se llevó a cabo cubriendo las siguientes etapas:

- Planteamiento general.
- Descripción del modelo actual de las rutas de recolección de RSU en León, Gto.
- Determinación de la técnica de solución para el problema del diseño de las rutas de recolección.
- Desarrollo de metodología para el rediseño de rutas de recolección - Recorrido Mínimo AG-TSP-Google.

3.1. Planteamiento general.

El planteamiento general del problema se estableció mediante la obtención de información oficial y la revisión del estado del arte y del marco teórico de temas relacionados.

3.1.1. Obtención de información de las rutas de recolección del municipio.

- Entrevista con Director del SIAP.

- Trámite ante la Unidad Municipal de Acceso a la Información Pública del Municipio de León, Gto. (UMAIP), para obtención de documentación de las rutas del municipio:

  - Nombres de calles**,** frecuencias y horarios de recolección (archivos Excel).

  - Mapas (fotocopias y archivos pdf).

  - Cifras de tonelajes de recolección de residuos sólidos por ruta, período Ene-Ago13.

3.1.2. Revisión del estado del arte y del marco teórico.

- Extracción y análisis de artículos de bases de datos bibliográficos bajo los tópicos Recolección de residuos, AG's, TSP y VRP **(**Problema de Ruteo de Vehículos).

- Revisión de marco teórico de técnicas metaheurísticas en general, los AG's y el TSP.

3.2. Descripción del modelo actual de las rutas de recolección de RSU en León, Gto.

El modelo de las rutas de recolección de RSU del municipio se caracterizó con base en la información del SIAP y se complementó con información directa proporcionada por un operador de un vehículo de recolección.

3.2.1. Características del modelo actual.

En el modelo actual de las rutas de recolección de RSU, se identificaron los aspectos principales descritos en la sección 1.2. Descripción del problema, con base en la información obtenida mediante el trámite de solicitud de información al SIAP a través de la UMAIP y de la consulta de la información disponible en la página web del SIAP León, 2014.

3.2.2. Tiempos y movimientos en el modelo actual.

Se tuvo la facilidad de contactar a un operador de un vehículo de recolección de residuos del municipio. A continuación se presenta de manera específica información sobre los horarios e itinerario de la ruta 16 vigentes a principios del año 2014.

3.2.2.1. Horarios semanales de recorrido ruta 16.

- Datos generales:

  Lugar de encierro del vehículo: Bombay 818. Col. Escondida.

  Encendido del vehículo: 19:00 hrs.

  Salida de pensión: 19:15 hrs.

  Traslado a zona de trabajo: vía Blvd. Timoteo Lozano, continuación por Blvd. Cervantes Saavedra y finalmente Blvd. Manuel de Austri.

- Tarea: Carga de todos los residuos (no peligrosos) de la vía pública.
- Límites de zona de trabajo (ver mapa de la figura 1)
  - Blvd. Manuel de Austri.
  - Blvd. San Juan Bosco.
  - Calle San Juan de los Lagos.
  - Calle Guatemala/Nicaragua.
  - Blvd. Campeche.
  - Calle Chiapas norte.

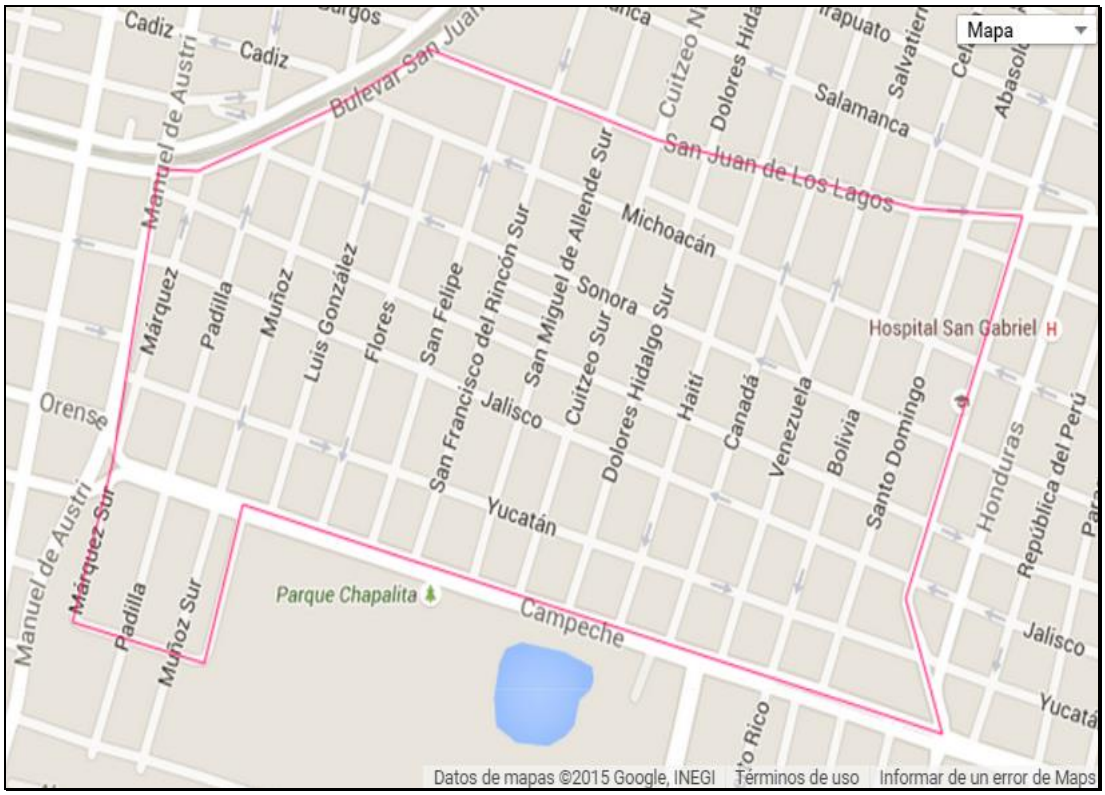


Fig. 1. Mapa ruta 16 (Daft Logic, 2015).

Derivado del análisis del mapa, se identificaron 119 nodos y 209 aristas. No se tomaron en cuenta calles muy estrechas y algunas privadas.

- Horarios de recorrido:

  Los recorridos durante los días de la semana se llevan a cabo aproximadamente en los horarios descritos en la tabla 2.

Tabla 2. Horarios de recorrido ruta 16.

Día	Inicio	Suspensión para tiro	Carga aprox. (tons.)	Número calles pendientes	Reinicio (sig. día)	Fin recole-cción	Fin des-carga
Domingo*	19:45	22:30	8 - 9	8	00:00	2:00	3:00
Lunes	19:45	23:30	8 - 9	4	01:00	2:00	
Martes	19:45	23:15	8 - 9	6	00:45	2:00	
Miércoles	19:45	23:15	8 - 9	6	00:45	2:00	
Jueves	19:45	23:15	8 - 9	6 – 8	00:45	2:00	
Viernes *	19:45	22:30	8 - 9	10	00:00	2:00	3:00
Sábado	No hay servicio						

* Domingo y viernes, se realizan 2 viajes, el segundo es para recolectar en calles faltantes.

De acuerdo con la tabla 2, cuatro de los días de la semana requieren 6 horas con 15 minutos de labores y dos días necesitan una hora más.

- Inicio de recorrido:
    - 19:45 hrs.
    - Calle Chiapas Norte (Secundaria # 5 “Emperador Cuauhtémoc).
- Criterio para el manejo de contratiempos por obras o festividades:
    - Saltar calles cerradas y regreso posterior.
- Primera descarga (aproximadamente con ocho toneladas): a las 23:00 hs.

  La capacidad de carga ha llegado hasta doce toneladas (capacidad vol. 6 x 2.5 x 2.25 m^3).
- El relleno sanitario se ubica en la Col. Lagunilla, pasando la *“Y”* de la salida a Lagos de Moreno. Se siguen los siguientes pasos:
    - Acceso controlado.
    - Paso a báscula.
    - Paso a zona de tiro.

3.2.2.2. Itinerario de recorrido ruta 16.

El siguiente recorrido correspondiente al lunes 3 de marzo'14 se llevó a cabo de acuerdo con el itinerario descrito en la tabla 3.

Tabla 3. Itinerario de recorrido ruta 16, lunes 03 de marzo 2014.

Cruce de calles	Hora
Chiapas norte /Manuel de Austri	8:10 p.m.
Manuel de Austri /Campeche	8:11 p.m.
Campeche / Padilla	8:12 p.m.
Padilla / San Juan Bosco	8:19 p.m.
San Juan Bosco / Muñoz	8:20 p.m.
Muñoz /Campeche	8:27 p.m.
Campeche /Padilla Sur	8:28 p.m.
Padilla Sur /Chiapas Norte	8:32 p.m.
Chiapas Norte / Muñoz Sur	8:33 p.m.
Muñoz sur / Campeche	8:36 p.m.
Campeche / Luis González	8:37 p.m.
Luis Gonzalez / Veracruz	8:39 p.m.
Veracruz / Márquez Sur	8:40 p.m.
Márquez Sur / Campeche	8:41 p.m.
Campeche / San Miguel	8:43 p.m.
San Miguel / San Juan de los Lagos	8:53 p.m.
San Juan de los Lagos / Cuitzeo	8:54 p.m.
Cuitzeo / Campeche	9:03 p.m.
Campeche / Canadá	9:05 p.m.
Canadá /Zacatecas	9:09 p.m.
Zacatecas / Nicaragua	9:12 p.m.
Nicaragua / Jalisco	9:14 p.m.
Jalisco/ Márquez	9:28 p.m.
Márquez / Zacatecas	9:29 p.m.
Zacatecas / Canadá	9:39 p.m.
Canadá / San Juan de los Lagos	9:44 p.m.

Tabla 3. Itinerario de recorrido ruta 16, lunes 03 de marzo 2014 (continuación).

Cruce de calles	Hora
San Juan de los Lagos / Bolivia	9:45 p.m.
Bolivia / Campeche	9:56 p.m.
Campeche / Venezuela	9:57 p.m.
Venezuela / San Juan de los Lagos	10:07 p.m.
San Juan de los Lagos / Nicaragua	10:09 p.m.
Nicaragua / Michoacán	10:10 p.m.
Michoacán / Dolores Hidalgo	10:14 p.m.
Dolores Hidalgo / San Juan de los Lagos	10:15 p.m.
San Juan de los Lagos / Haití	10:16 p.m.
Haití / Campeche	10:27 p.m.
Campeche / Dolores Hidalgo	10:28 p.m.
Dolores Hidalgo / Michoacán	10:34 p.m.
Michoacán / San Juan Bosco	10:41 p.m.
San Juan Bosco / San Juan de los Lagos	10:43 p.m.
San Juan de los Lagos / San Fco. del Rincón	10:46 p.m.
San Fco. del Rincón / Campeche	11:02 p.m.
Campeche / San Felipe	11:04 p.m.
San Felipe / San Juan de los Lagos	11:20 p.m.
Acomodo de objetos, colocación de lona, revisión de llantas. Se parte rumbo al relleno.	11:26 p.m.
Campeche / Luis González	1.00 a.m.
Luis González / Michoacán	1:07 a.m.
Michoacán / Flores	1:08 a.m.
Flores / Campeche	1:14 a.m.
Campeche / Manuel de Austri	1:16 a.m.
Manuel de Austri / San Juan Bosco	1:18 a.m.
San Juan Bosco / Márquez	1:18 a.m.
Márquez / Yucatán (reversa)	1:20 a.m.
Yucatán / Sto. Domingo	1:31 a.m.
Sto. Domingo / Sonora	1:32 a.m.
Sonora/ San Juan Bosco	1:42 a.m.
Fin (acomodo de objetos, regreso a pensión).	1:43 a.m.

Del análisis del recorrido se detectaron algunas aristas y nodos faltantes de visitar, con base en ello se calcularon los siguientes indicadores (tabla 4):

Tabla 4. indicadores del recorrido.

Aristas		
Recorridas	172	82.2%
Faltantes de recorrer	**29**	**14.0%**
Recorridas en sentido contrario	**3**	**1.4%**
Redundantes (repetidas)	5	2.4%
Total	209	100.0%
Nodos		
Visitados	114	95.8%
Faltantes de visitar	**5**	**4.2%**
Total	119	100.0%

Del total de las 209 aristas, las faltantes de recorrer 29 representan el 14%; las que se cubrieron en **sentido contrario** 3 representan el 1.4%. Con relación a los nodos, del total de 119, faltaron ser visitados 5 o sea el 4.2%.

Se observa que el recorrido se inició a las 8:10 p.m., o sea 25 minutos después de la hora normal (7:45 p.m.) y se finalizó a la 1:43, es decir, 17 minutos antes de las 2:00 (hora normal de término), originando un tiempo total de recorrido de 5 horas con 33 minutos.

3.3. Determinación de la técnica de solución para el problema del diseño de las rutas de recolección.

Esta etapa se llevó a cabo en tres partes:

Con el propósito de conocer las características del TSP y tener una noción de sus métodos de solución y sus limitaciones, para saber el tipo de problemáticas que se pueden presentar durante su resolución, inicialmente se decidió tener la experiencia de resolver el TSP mediante un método exacto, así como mediante procedimientos heurísticos, antes de entrar de lleno en las técnicas metaheurísticas (AG's).

De acuerdo con lo anterior, se buscó un ejemplo de solución del TSP en la literatura de Investigación de Operaciones. Concretamente por su sencillez y debido a que analiza el problema del surgimiento de sub-toures al resolver el TSP por Programación Lineal Entera (PLE) , así como la complejidad computacional de lo que este método exacto involucra, se empleó el de Maroto et al. (2002). Complementariamente se utilizó el software comercial QSB con su herramienta PLE.

Gracias a lo anterior, efectivamente se obtuvieron conocimientos sobre la dificultad que se presenta al resolver el TSP con este método, el surgimiento de sub-ciclos en los recorridos propuestos y la manera de resolverlo (con variables y restricciones adicionales), así como la complejidad desde el punto de vista computacional que conlleva.

Posteriormente se resolvió también el ejemplo del TSP de Maroto mediante las técnicas heurísticas disponibles en el mismo software comercial QSB. Habiendo llevado a cabo esta experiencia se adquirió el conocimiento sobre la frecuente necesidad de alimentar datos (distancias) correspondientes a nodos no adyacentes, que involucran sub-recorridos que comprenden tramos de aristas consecutivas necesarias para cerrar un ciclo y terminar el algoritmo de ejecución

del programa en turno. A la postre, hemos dado en nombrar a estos sub-recorridos aristas auxiliares.

Adicionalmente se aplicaron las mismas herramientas heurísticas a dos instancias de diferente tamaño de una ruta del municipio.

Derivado de estas experiencias se observó que QSB es muy dependiente de la manera en que está estructurada la matriz de adyacencias, es decir, es muy rígido respecto al uso de los datos de la matriz de adyacencias, ya que los recorridos de las soluciones que genera siempre inician en el nodo 1. Así mismo, para proporcionar la solución de las instancias de la ruta uno que se le alimentaron, siempre requirió las mismas aristas auxiliares en reiteradas ejecuciones del paquete.

La segunda parte de esta etapa estuvo enfocada a buscar entre diferentes fuentes, tratando de localizar elementos en general que auxiliasen en la implementación de la solución del TSP mediante la aplicación de los AG's.

Finalmente, en la tercera parte, de entre los elementos localizados, se eligió la propuesta de Taha (2009) para estructurar el programa de Algoritmos Genéticos para la solución del TSP. Se completó esta parte presentando una comparación vs. el AG de carácter general planteado por Goldberg.

3.3.1. Análisis de propuestas científicas para la solución de problemas TSP.

Se resolvió el TSP por PLE y se analizó su complejidad computacional. Así mismo se resolvió el TSP por técnicas heurísticas.

3.3.1.1. Solución del TSP por Programación Lineal Entera. Ejemplo del problema TSP de Maroto et al. (2002).

Para obtener un conocimiento práctico inicial sobre el tema se replicó el ejemplo del problema TSP, El Viajante de comercio de Maroto et al. (2002), el cual se

encuentra resuelto en el libro por PLE. En este caso se resolvió con el mismo método del software comercial QSB.

Normalmente al resolver este tipo de problemas por PLE la solución óptima que se encuentra no proporciona una trayectoria continua a través de todos los nodos, sino que se forma de varios lazos desconectados (o sub-toures), por lo cual es necesario agregar otro tipo de restricciones al modelo. Cabe referir a Martí (2001) sobre el comentario que agrega a la descripción del modelo matemático que presenta:

> Las restricciones que aparecen…reciben el nombre de *restricciones de eliminación de subtours* y garantizan que la solución sea un tour. El problema es que aparecen en una cantidad del orden de $2^{(n/2)}$, lo cual hace inmanejable tal formulación. Se han encontrado restricciones alternativas para evitar la formación de subtours que suponen la incorporación de una cantidad polinómica de restricciones (Miller, Tucker y Zemlin, 1960). Aun así la resolución óptima del problema ha resultado poco eficiente, salvo para ejemplos relativamente pequeños, dado el elevado tiempo de computación requerido por cualquier método exacto. p. 10.

Planteamiento del problema del ejemplo del TSP de Maroto et al. (2002):

Un viajante de una conocida empresa cerámica debe visitar cinco importantes ciudades españolas, y desea conocer el orden en que debe hacer las visitas de manera que la distancia total recorrida sea mínima.

Las distancias que separan las cinco ciudades aparecen a continuación (tabla 5):

Tabla 5. Matriz de adyacencias cinco ciudades, ejemplo TSP Maroto et al. (2012).

	Alicante	Barcelona	Burgos	Madrid	Valencia
Alicante	0	552	661.4	424.6	178.3
Barcelona	594.4	0	629	628.7	363.2
Burgos	662.2	628.3	0	247.2	598
Madrid	424.1	628.2	246.7	0	360
Valencia	178.1	364.7	595.5	358.7	0

La figura 2 muestra el grafo del problema:

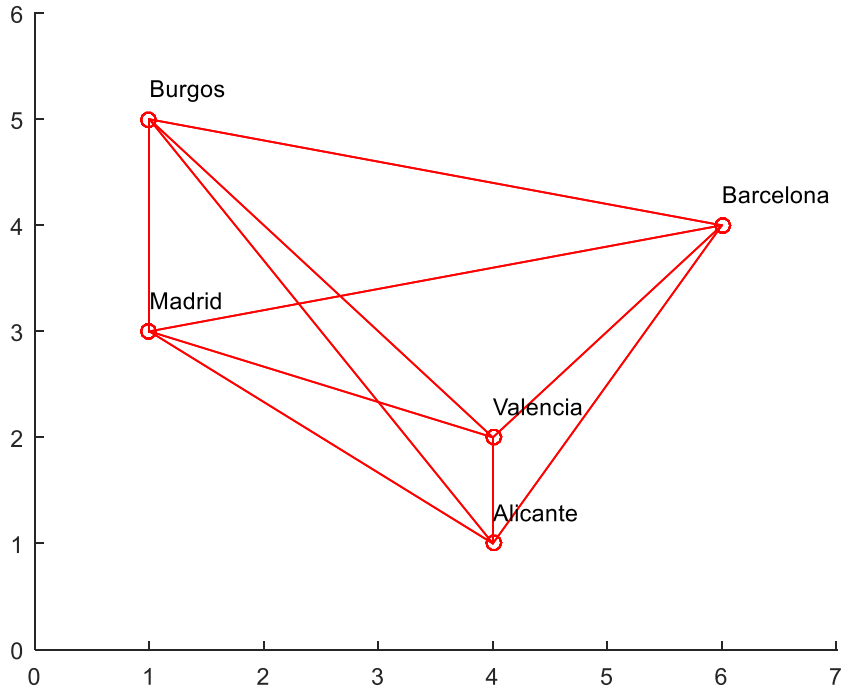


Fig. 2. Grafo de cinco ciudades, ejemplo TSP Maroto et al. (2012).

Solución:

Como se puede apreciar, la matriz de adyacencias (tabla 5), no es simétrica.

De acuerdo con los datos, originalmente se tienen 20 variables y 10 restricciones. Sin embargo, para evitar la formación de sub-ciclos o sub-toures, se incorporaron 5 variables auxiliares y 16 restricciones más.

Con las adiciones anteriores, se creó el siguiente modelo de datos en la sección PLE en QSB (tabla 6).

Tabla 6. Modelo de datos en QSB sección PLE. Cinco ciudades, ejemplo TSP Maroto et al. (2012).

Variable -->	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X17	X18	X19	X20	X21	X22	X23	X24	X25	Direction	R. H. S.
Minimize	552	661.4	424.6	179.3	549.4	629	628.7	363.2	662.2	629.3	247.2	598	424.1	628.2	246.7	360	179.1	364.7	595.5	358.7	1	1	1	1	1		
C1	1	1	1	1																						>=	1
C2					1	1	1	1																		>=	1
C3									1	1	1	1														>=	1
C4													1	1	1	1										>=	1
C5																	1	1	1	1						>=	1
C6					1				1				1				1									>=	1
C7	1									1				1				1								>=	1
C8		1				1									1				1							>=	1
C9			1				1				1									1						>=	1
C10				1				1				1				1										>=	1
C11	-5																				-1	1				>=	-4
C12										-5												1	-1			>=	-4
C13														-5								1		-1		>=	-4
C14																		-5				1			-1	>=	-4
C15		-5																			-1		1			>=	-4
C16						-5																-1	1			>=	-4
C17															-5								1	-1		>=	-4
C18																			-5				1		-1	>=	-4
C19			-5																		-1			1		>=	-4
C20							-5															-1		1		>=	-4
C21											-5												-1	1		>=	-4
C22																				-5				1	-1	>=	-4
C23				-4																	-1				1	>=	-4
C24								-4														-1			1	>=	-4
C25												-4											-1		1	>=	-4
C26																-4								-1	1	>=	-4
LowerBound	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
UpperBound	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	M	M	M	M	M		

Y se obtuvo la siguiente solución (tabla 7):

Tabla 7. Resultados de QSB sección PLE. Cinco ciudades, ejemplo TSP Maroto et al. (2012).

	20:11:28		Tuesday	January	13	2015
	Decision	Solution	Unit Cost or	Total	Reduced	Basis
1	X1	0	552,00	0	0	basic
2	X2	0	661,40	0	487,40	at bound
3	X3	0	424,60	0	424,60	at bound
4	X4	1,00	179,30	179,30	0	basic
5	X5	0	549,40	0	372,00	at bound
6	X6	1,00	629,00	629,00	629,00	at bound
7	X7	0	628,70	0	627,80	at bound
8	X8	0	363,20	0	357,90	at bound
9	X9	0	662,20	0	233,50	at bound
10	X10	0	629,30	0	0	basic
11	X11	1,00	247,20	247,20	0	basic
12	X12	0	598,00	0	341,40	at bound
13	X13	1,00	424,10	424,10	0	basic
14	X14	0	628,20	0	3,50	at bound
15	X15	0	246,70	0	0	basic
16	X16	0	360,00	0	108,00	at bound
17	X17	0	179,10	0	0	basic
18	X18	1,00	364,70	364,70	0	basic
19	X19	0	595,50	0	593,80	at bound
20	X20	0	358,70	0	356,10	at bound
21	X21	0	1,00	0	1,00	at bound
22	X22	1,00	1,00	1,00	0	basic
23	X23	2,00	1,00	2,00	0	basic
24	X24	3,00	1,00	3,00	0	basic
25	X25	0	1,00	0	4,00	at bound
	Objective	Function	(Min.) =	1850,30		

A la que, descontando el valor de las variables auxiliares X22, X23 y X24, corresponde el resultado numérico que a continuación se muestra (tabla 8).

Tabla 8. Resultados netos de QSB sección PLE. Cinco ciudades, ejemplo TSP Maroto et al. (2012).

De	A	Distancia
Alicante	Valencia	179.3
Valencia	Barcelona	364.7
Barcelona	Burgos	629
Burgos	Madrid	247.2
Madrid	Alicante	424.1
	Total	1,844.3

Se incluye el grafo de la solución anterior (figura 3):

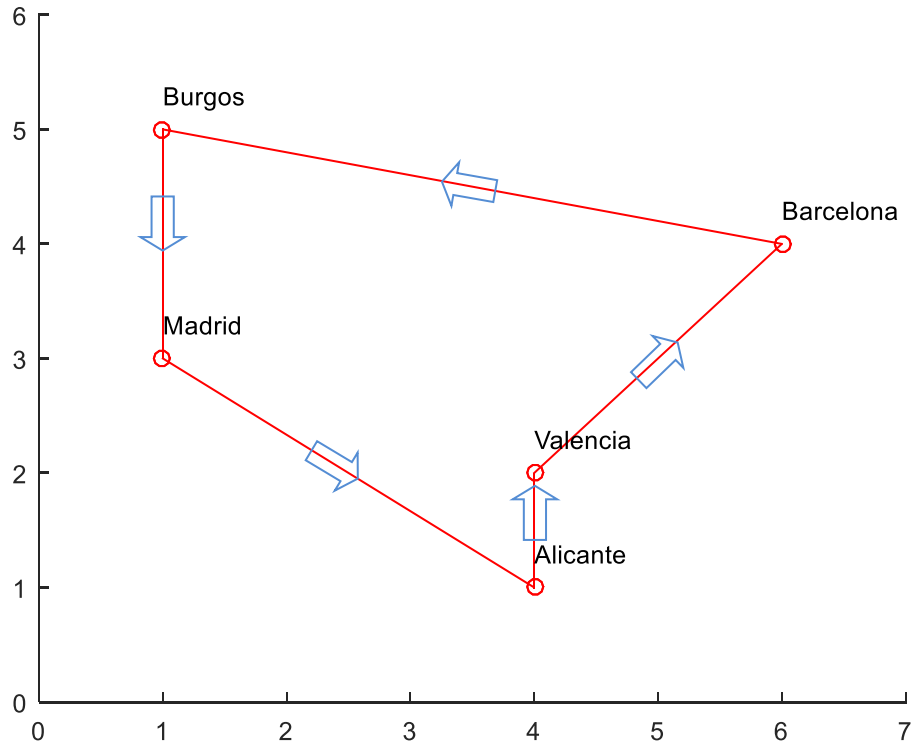


Fig. 3. Grafo de la solución cinco ciudades, ejemplo TSP Maroto et al. (2012).

El resultado del libro de Maroto et al. (2002) se muestra en la tabla 9 y corresponde al mismo recorrido, en sentido inverso.

Tabla 9. Resultados del libro. Cinco ciudades, ejemplo TSP Maroto et al. (2012).

De	A	Distancia
Alicante	Madrid	424.6
Madrid	Burgos	246.7
Burgos	Barcelona	629.3
Barcelona	Valencia	363.2
Valencia	Alicante	179.1
	Total	1,842.9

Cabe reiterar el hecho de que para evitar la formación de sub-ciclos fue necesario incorporar cinco variables auxiliares y 16 restricciones más.

3.3.1.2. Análisis de complejidad computacional del problema TSP resuelto por PLE.

Para evitar el problema de la formación de sub-ciclos se pueden agregar variables y/o restricciones al modelo matemático de PLE. Existen dos alternativas. En la primera se añadirían n variables y $(n-1)^2$ restricciones al modelo original. En la segunda alternativa serían añadidas 2^n restricciones (Maroto et al., 2002).

Eligiendo la primera alternativa, en la tabla 10 se muestra la progresión del número de variables y restricciones necesarias para resolver el problema TSP mediante PLE.

Tabla 10. Número de variables y restricciones necesarias para resolver el TSP con PLE.

Número de ciudades	**n**	**5**	**10**	**15**	**29**
Número inicial					
Variables	$n^2 - n$	20	90	210	812
Restricciones	2n	10	20	30	58
Auxiliares adicionales para evitar subciclos					
Variables	n	5	10	15	29
Restricciones	$(n-1)^2$	16	81	196	784
Totales					
Variables	n^2	25	100	225*	841*
Restricciones	$n^2 + 1$	26	101	226*	842*

***** QSB no tiene la capacidad de manejar estas cantidades con su herramienta PLE.

3.3.1.3. Solución del TSP con métodos heurísticos. Ejemplo del problema TSP de Maroto et al. (2002).

Al aplicar al mismo ejemplo de Maroto et al. (2002) el método exacto no analítico Ramificación y Acotamiento y las heurísticas Inserción más Barata y Mejoramiento por Intercambio Dos-Vías del software comercial QSB, se obtiene el resultado correcto sin manejar variables ni restricciones adicionales.

Planteamiento del problema del ejemplo:

Un viajante de una conocida empresa cerámica debe visitar cinco importantes ciudades españolas, y desea conocer el orden en que debe hacer las visitas de manera que la distancia total recorrida sea mínima.

Las distancias que separan las cinco ciudades aparecen en la tabla 11:

Tabla 11. Matriz de adyacencias cinco ciudades, ejemplo TSP Maroto et al. (2012).

	Alicante	Barcelona	Burgos	Madrid	Valencia
Alicante	0	552	661.4	424.6	178.3
Barcelona	594.4	0	629	628.7	363.2
Burgos	662.2	628.3	0	247.2	598
Madrid	424.1	628.2	246.7	0	360
Valencia	178.1	364.7	595.5	358.7	0

La figura 4 muestra el grafo del problema:

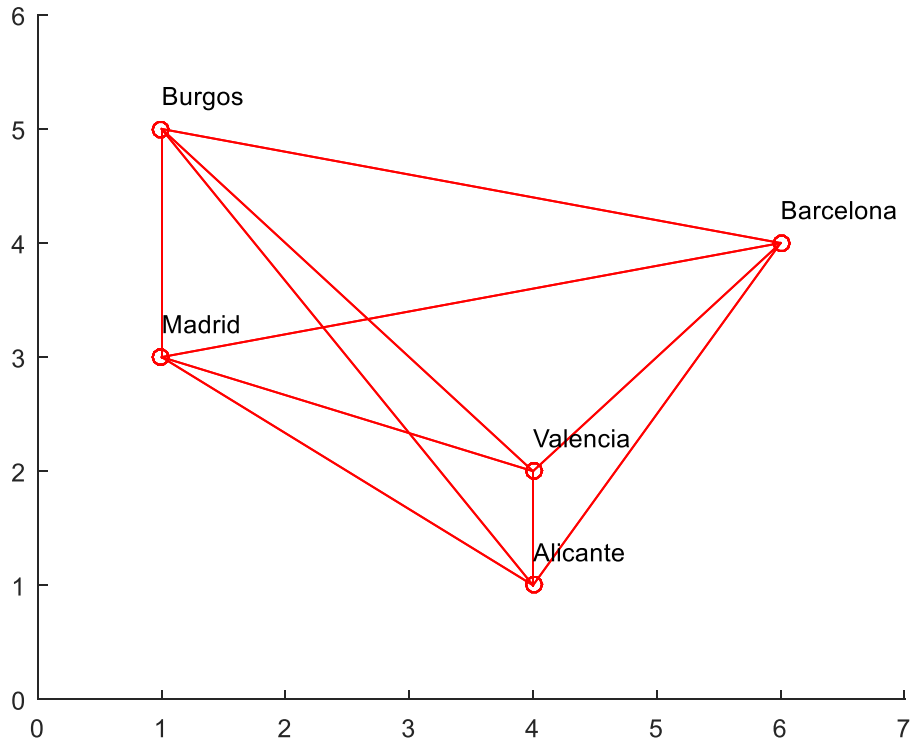


Fig. 4. Grafo de cinco ciudades, ejemplo TSP Maroto et al. (2012).

Solución:

Como se puede apreciar la matriz de datos (matriz de adyacencia) no es simétrica.

Se creó el siguiente modelo en QSB en la sección modelado de redes (Network Modeling), tipo de problema TSP (tabla 12).

Tabla 12. Modelo de datos QSB (Network Modeling). Cinco ciudades, ejemplo TSP Maroto et al. (2012)

From \ To	Alicante	Barcelona	Burgos	Madrid	Valencia
Alicante	0	552	661,4	424,6	179,3
Barcelona	594,4	0	629	628,7	363,2
Burgos	662,2	629,3	0	247,2	598
Madrid	424,1	628,2	246,7	0	360
Valencia	179,1	364,7	595,5	358,7	0

Y se obtuvo la solución por el método de Ramificación y Acotamiento y las heurísticas Inserción más Barata y Mejoramiento por Intercambio Dos-Vías. Los tres métodos proporcionan los mismos resultados, sin embargo por cuestiones de espacio se incluyen sólo los correspondientes a Inserción más Barata, mismos que se muestran en la tabla 13.

Tabla 13. Resultados de QSB (Network Modeling). Cinco ciudades, ejemplo TSP Maroto et al. (2012).

01-14-2015	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Alicante	Valencia	179,3	4	Burgos	Madrid	247,2
2	Valencia	Barcelona	364,7000	5	Madrid	Alicante	424,1
3	Barcelona	Burgos	629				
	Total	Minimal	Traveling	Distance	or Cost	=	1.844,30
	(Result	from	Cheapest	Insertion	Heuristic)		

Cabe reiterar que este resultado es igual al obtenido con PLE del mismo software QSB.

Se muestra el grafo de la solución anterior (figura 5):

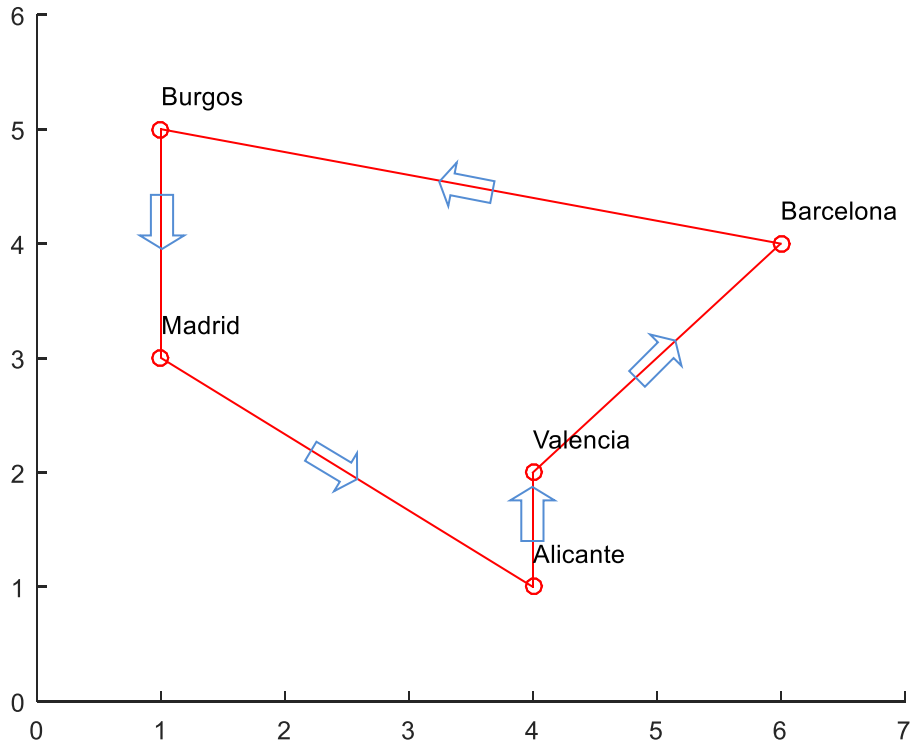


Fig. 5. Grafo de la solución cinco ciudades, ejemplo TSP Maroto et al. (2012).

El resultado del libro de Maroto et al. (2002) es el siguiente y corresponde al mismo recorrido, en sentido inverso (tabla 14).

Tabla 14. Resultados del libro. Cinco ciudades, ejemplo TSP Maroto et al. (2012).

De	A	Distancia
Alicante	Madrid	424.6
Madrid	Burgos	246.7
Burgos	Barcelona	629.3
Barcelona	Valencia	363.2
Valencia	Alicante	179.1
	Total	1,842.9

Con el fin de replicar el mismo resultado del libro fue necesario crear el siguiente modelo (tabla 15), cargando sólo los datos de los eslabones o adyacencias indicadas en la matriz de datos siguiente. Es decir, con esto se asegura el recorrido en el sentido indicado en la solución del libro.

Tabla 15. Modelo con datos en un sólo sentido en QSB (Network Modeling). Cinco ciudades, ejemplo TSP Maroto et al. (2012).

From \ To	ALICANTE	BARCELONA	BURGOS	MADRID	VALENCIA
ALICANTE				424,6	
BARCELONA					363,2
BURGOS		629,3			
MADRID			246,7		
VALENCIA	179,1				

Gracias a lo cual se obtienen resultados idénticos al libro, por los tres métodos, Ramificación y Acotamiento y las heurísticas Inserción más Barata y Mejoramiento por Intercambio Dos-Vías (nuevamente por razones de espacio se muestra sólo los correspondientes a Inserción más Barata, tabla 16).

Tabla 16. Resultados de QSB (Network Modeling) con datos en un sólo sentido. Cinco ciudades, ejemplo TSP Maroto et al (2012).

01-14-2015	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	ALICANTE	MADRID	424,6	4	BARCELONA	VALENCIA	363,2000
2	MADRID	BURGOS	246,7	5	VALENCIA	ALICANTE	179,1
3	BURGOS	BARCELONA	629,3000				
	Total	Minimal	Traveling	Distance	or Cost	=	1.842,90
	(Result	from	Cheapest	Insertion	Heuristic)		

Se muestra el grafo de la solución anterior (figura 6):

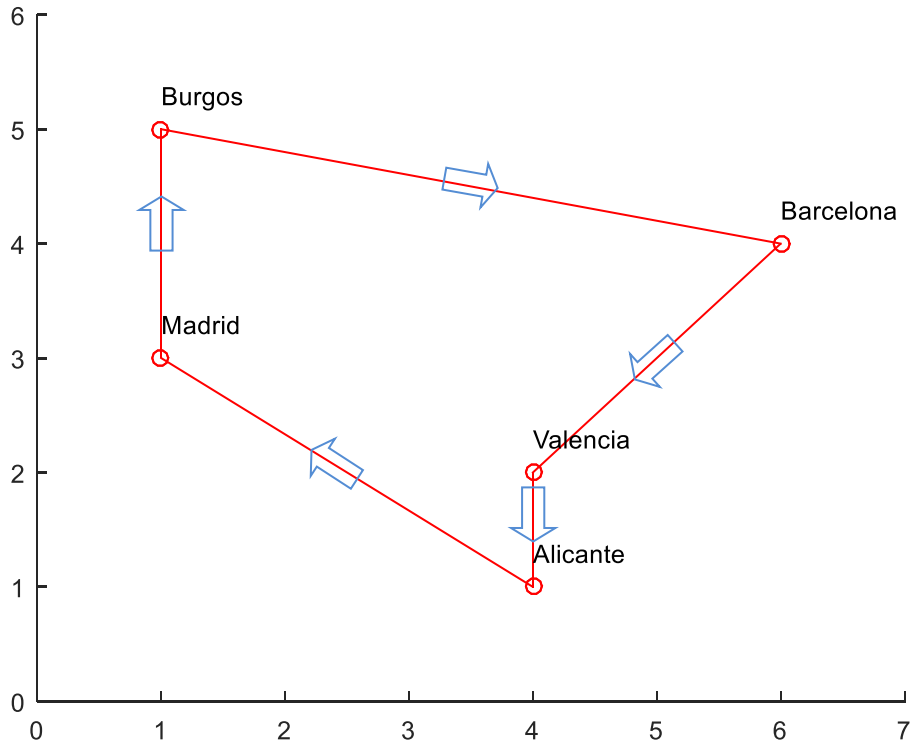


Fig. 6. Grafo de la solución cinco ciudades, ejemplo TSP Maroto et al. (2012).

3.3.1.4. Solución del TSP con métodos heurísticos, ruta uno; 16 nodos.

A manera de referencia se ejecutó el problema TSP para 16 y 29 nodos de la ruta uno del municipio, aplicando el mismo método exacto no analítico, Ramificación y Acotamiento, y las heurísticas Inserción más Barata y Mejoramiento por Intercambio Dos-Vías del software comercial QSB.

Planteamiento del problema TSP ruta uno; 16 nodos:

El planteamiento de esta ruta se realiza a través de su grafo y matriz de adyacencias correspondiente.

El dibujo del grafo de la ruta es el siguiente (figura 7):

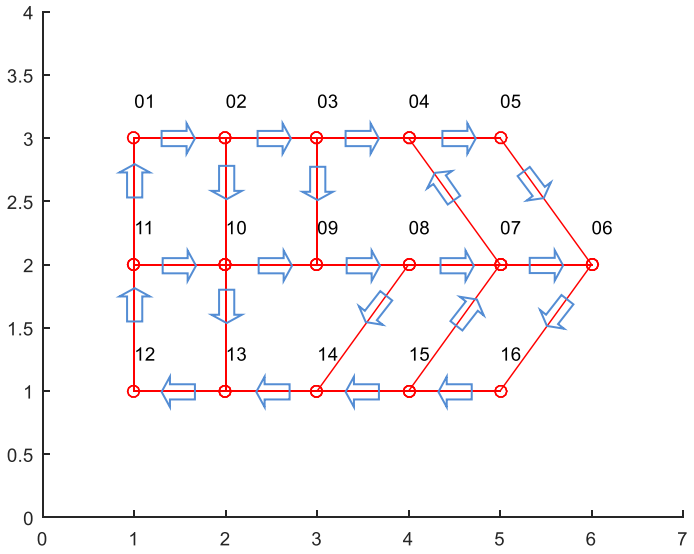


Fig. 7. Grafo de la ruta uno; 16 nodos.

Su matriz de adyacencias ponderada es la mostrada en la tabla 17:

Tabla 17. Matriz de adyacencias ponderada. Ruta uno, 16 nodos.

9999	0150	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	0100	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999	9999
9999	9999	9999	0210	9999	9999	9999	9999	0110	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	0200	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120
9999	9999	9999	0160	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	0180	9999	9999	9999	9999	9999	9999	0120	9999	9999
9999	9999	9999	9999	9999	9999	9999	0020	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	0110	9999	9999	9999
0100	9999	9999	9999	9999	9999	9999	9999	9999	0160	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0170	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120	9999	9999	9999
9999	9999	9999	9999	9999	9999	0120	9999	9999	9999	9999	9999	9999	0180	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0180	9999

A partir de la cual se creó el modelo de datos que se alimentó a QSB en la sección modelado de redes (Network Modeling), tipo de problema TSP. Originalmente se capturó la matriz de adyacencias simple correspondiente al grafo dibujado. Sin embargo, para proporcionar un resultado aceptable, el programa requirió la distancia entre los nodos 7-1. El recorrido que une estos dos nodos es: 7, 6, 16, 15, 14, 13, 12, 11, 1, y suma una distancia de *1,160 metros*. La cual fue agregada al modelo de datos que se capturó (tabla 18).

Tabla 18. Modelo de datos QSB (Network Modeling). Ruta uno; 16 nodos.

From \	Node1	Node2	Node3	Node4	Node5	Node6	Node7	Node8	Node9	Node10	Node11	Node12	Node13	Node14	Node15	Node16
Node1		150														
Node2			100							100						
Node3				210					110							
Node4					200											
Node5						190										
Node6																120
Node7	1160			160		190										
Node8							180							120		
Node9								20								
Node10									100				110			
Node11	100									160						
Node12											100					
Node13												170				
Node14													120			
Node15							120							180		
Node16															180	

El programa arrojó los siguientes resultados. Se puede apreciar que se produjo el mismo resultado en los tres métodos, distancia total igual a *3,340 metros* (tablas 19, 20 y 21).

Tabla 19. Resultados de QSB (Network Modeling) heurística Inserción más Barata. Ruta uno; 16 nodos.

11-14-2014	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Node1	Node2	150	9	Node14	Node13	120
2	Node2	Node3	100	10	Node13	Node12	170
3	Node3	Node4	210	11	Node12	Node11	100
4	Node4	Node5	200	12	Node11	Node10	160
5	Node5	Node6	190	13	Node10	Node9	100
6	Node6	Node16	120	14	Node9	Node8	20
7	Node16	Node15	180	15	Node8	Node7	180
8	Node15	Node14	180	16	Node7	Node1	1160
	Total	Minimal	Traveling	Distance	or Cost	=	3340
	(Result	from	Cheapest	Insertion	Heuristic)		

Tabla 20. Resultados de QSB (Network Modeling) heurística Mejoramiento por Intercambio Dos-Vías. Ruta uno; 16 nodos.

11-14-2014	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Node1	Node2	150	9	Node14	Node13	120
2	Node2	Node3	100	10	Node13	Node12	170
3	Node3	Node4	210	11	Node12	Node11	100
4	Node4	Node5	200	12	Node11	Node10	160
5	Node5	Node6	190	13	Node10	Node9	100
6	Node6	Node16	120	14	Node9	Node8	20
7	Node16	Node15	180	15	Node8	Node7	180
8	Node15	Node14	180	16	Node7	Node1	1160
	Total	Minimal	Traveling	Distance	or Cost	=	3340
	(Result	from	Two-way	Exchange	Improvement	Heuristic)	

Tabla 21. Resultados QSB (Network Model.) método Ramificación Acotamiento. Ruta uno;16 nodos

11-14-2014	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Node1	Node2	150	9	Node14	Node13	120
2	Node2	Node3	100	10	Node13	Node12	170
3	Node3	Node4	210	11	Node12	Node11	100
4	Node4	Node5	200	12	Node11	Node10	160
5	Node5	Node6	190	13	Node10	Node9	100
6	Node6	Node16	120	14	Node9	Node8	20
7	Node16	Node15	180	15	Node8	Node7	180
8	Node15	Node14	180	16	Node7	Node1	1160
	Total	Minimal	Traveling	Distance	or Cost	=	3340
	(Result	from	Branch	and	Bound	Method)	

El grafo que corresponde a esta solución es el siguiente (figura 8):

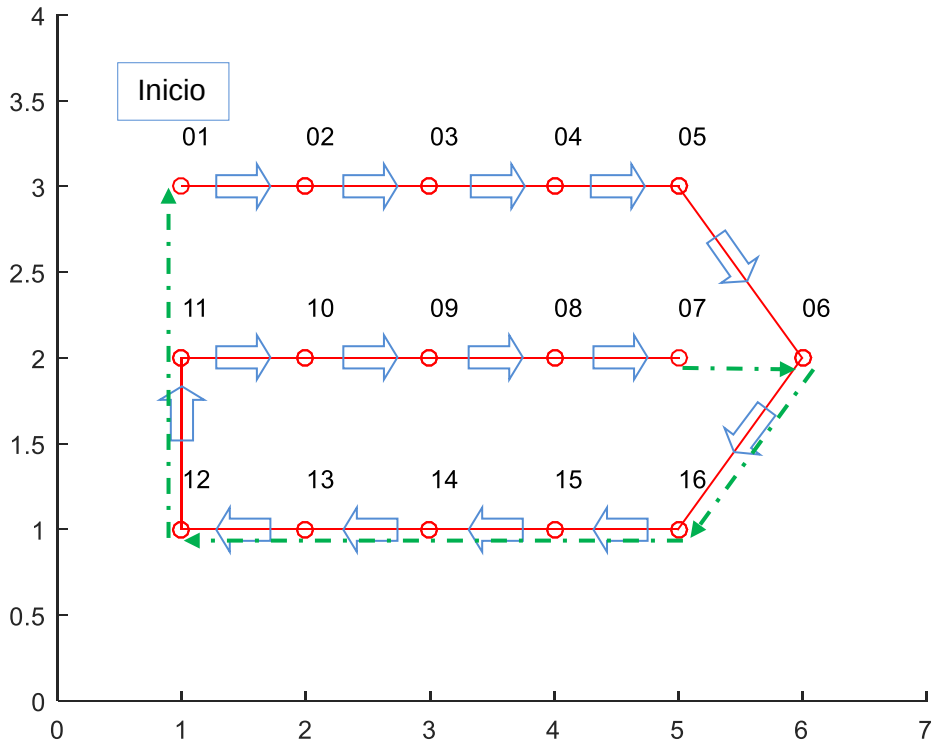


Fig. 8. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 7-1.

Se observa que el recorrido se inicia en el nodo 1. La solución corresponde a un ciclo hamiltoniano, no euleriano, es decir se han recorrido todos los nodos pero no todas las aristas.

Se aclara que el sistema no produjo una solución práctica con la heurística del Vecino más Cercano (tabla 22), ya que requirió adicionalmente la distancia entre los nodos 10 y 4 que no son adyacentes.

Tabla 22. Resultados de QSB (Network Mod.) heurística Vecino más Cercano. Ruta uno; 16 nodos.

11-14-2014	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Node1	Node2	150	9	Node11	Node10	160
2	Node2	Node3	100	10	Node10	Node4	M
3	Node3	Node9	110	11	Node4	Node5	200
4	Node9	Node8	20	12	Node5	Node6	190
5	Node8	Node14	120	13	Node6	Node16	120
6	Node14	Node13	120	14	Node16	Node15	180
7	Node13	Node12	170	15	Node15	Node7	120
8	Node12	Node11	100	16	Node7	Node1	1160
	Total	Minimal	Traveling	Distance	or Cost	=	M
	(Result	from	Nearest	Neighbor	Heuristic)		

3.3.1.5. Solución del TSP con métodos heurísticos, ruta uno; 29 nodos.

Planteamiento problema TSP ruta uno; 29 nodos:

La siguiente es la matriz de adyacencias de la ruta uno con 29 nodos (tabla 23).

Tabla 23. Matriz de adyacencias unitaria ruta uno; 29 nodos.

Nod	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
1		1																											
2			1							1																			
3				1					1																				
4					1																								
5						1																							
6																1													
7				1		1																							
8							1							1															
9								2																					
10									1				1																
11	1									1																			
12											1																		
13												1							1										
14													1																
15							1							1															
16															1		1												
17																		1						1					
18															1														
19																				1		1							
20												1																	
21																				1		1							
22																												1	
23																		1						1					
24																									1				
25																										1			
26																							1				1		
27																												1	
28																													1
29																					1								

El grafo respectivo es el de la figura 9:

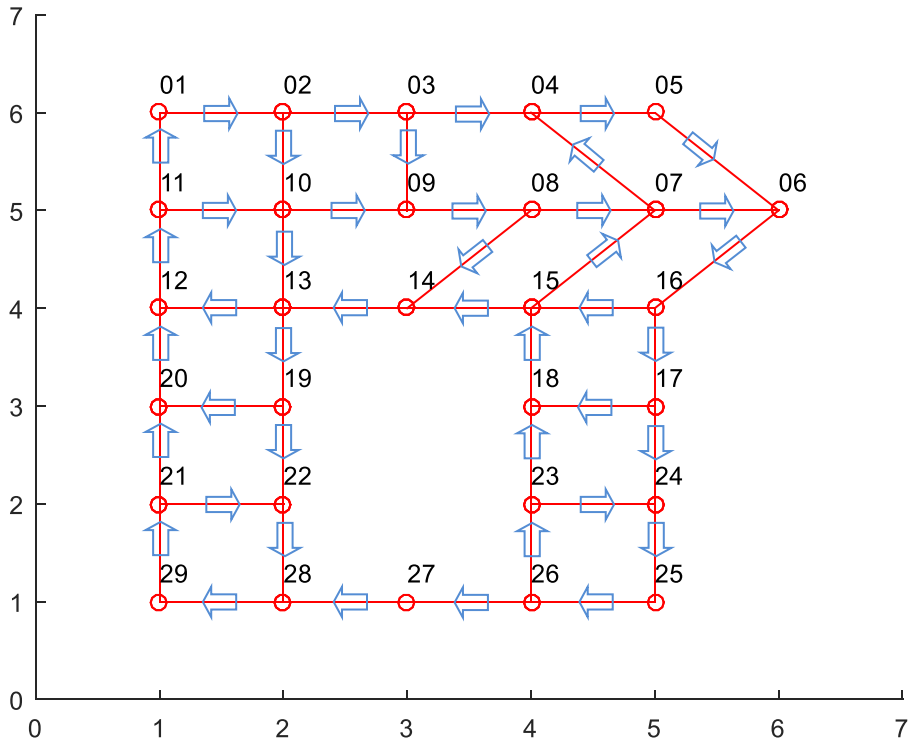


Fig. 9. Grafo de la ruta uno; 29 nodos.

Se muestra a continuación el modelo de datos creado en QSB en la sección modelado de redes (Network Modeling), tipo de problema TSP. Originalmente se capturó la matriz de adyacencias simple correspondiente al grafo dibujado. Sin embargo, para proporcionar un resultado aceptable, el programa requirió la distancia entre los nodos 27-23 y 7-1. Los recorridos que unen estos dos pares de nodos son: 27, 28, 29, 21, 20, 12, 11, 10, 9, 8, 7, 6, 16, 17, 24, 25, 26, 23, y, 7, 6, 16, 15, 14, 13, 12, 11, 1. Los cuales suman distancias totales de *2,170 metros* y *1,160 metros* respectivamente, mismos que fueron alimentados en el modelo de datos que sigue a continuación (tabla 24).

Tabla 24. Modelo de datos QSB (Network Modeling). Ruta uno; 29 nodos.

From \	Node1	Node2	Node3	Node4	Node5	Node6	Node7	Node8	Node9	Node10	Node11	Node12	Node13	Node14	Node15	Node16	Node17	Node18	Node19	Node20	Node21	Node22	Node23	Node24	Node25	Node26	Node27	Node28	Node29
Node1		150																											
Node2			100							100																			
Node3				210					110																				
Node4					200																								
Node5						190																							
Node6																120													
Node7	1160			160		190																							
Node8							180							120															
Node9								20																					
Node10									100				110																
Node11										160																			
Node12											100																		
Node13												170							120										
Node14													120																
Node15							120							180															
Node16															180		120												
Node17																		170						120					
Node18															120														
Node19																				170		120							
Node20												120																	
Node21																				120		180							
Node22																												120	
Node23																		120						170					
Node24																									110				
Node25																										160			
Node26																							120				170		
Node27																							2170					120	
Node28																													190
Node29																					120								

El programa arrojó los siguientes resultados. Se puede observar que se produjo el mismo resultado en los tres métodos, distancia total *6,990* metros (tablas 25 a 27).

Tabla 25. Resultados QSB (Network Model.) heurística Inserción más Barata. Ruta uno; 29 nodos.

01-14-2015	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Node1	Node2	150	16	Node14	Node13	120
2	Node2	Node3	100	17	Node13	Node19	120
3	Node3	Node4	210	18	Node19	Node22	120
4	Node4	Node5	200	19	Node22	Node28	120
5	Node5	Node6	190	20	Node28	Node29	190
6	Node6	Node16	120	21	Node29	Node21	120
7	Node16	Node17	120	22	Node21	Node20	120
8	Node17	Node24	120	23	Node20	Node12	120
9	Node24	Node25	110	24	Node12	Node11	100
10	Node25	Node26	160	25	Node11	Node10	160
11	Node26	Node27	170	26	Node10	Node9	100
12	Node27	Node23	2170	27	Node9	Node8	20
13	Node23	Node18	120	28	Node8	Node7	180
14	Node18	Node15	120	29	Node7	Node1	1160
15	Node15	Node14	180				
	Total	Minimal	Traveling	Distance	or Cost	=	6990
	(Result	from	Cheapest	Insertion	Heuristic)		

Tabla 26. Resultados de QSB (Network Modeling) heurística Mejoramiento por Intercambio Dos-Vías. Ruta uno; 29 nodos.

01-14-2015	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Node1	Node2	150	16	Node14	Node13	120
2	Node2	Node3	100	17	Node13	Node19	120
3	Node3	Node4	210	18	Node19	Node22	120
4	Node4	Node5	200	19	Node22	Node28	120
5	Node5	Node6	190	20	Node28	Node29	190
6	Node6	Node16	120	21	Node29	Node21	120
7	Node16	Node17	120	22	Node21	Node20	120
8	Node17	Node24	120	23	Node20	Node12	120
9	Node24	Node25	110	24	Node12	Node11	100
10	Node25	Node26	160	25	Node11	Node10	160
11	Node26	Node27	170	26	Node10	Node9	100
12	Node27	Node23	2170	27	Node9	Node8	20
13	Node23	Node18	120	28	Node8	Node7	180
14	Node18	Node15	120	29	Node7	Node1	1160
15	Node15	Node14	180				
	Total	Minimal	Traveling	Distance	or Cost	=	6990
	(Result	from	Two-way	Exchange	Improvement	Heuristic)	

Tabla 27. Resultados de QSB (Network Modeling) método Ramificación y Acotamiento. Ruta uno; 29 nodos.

01-14-2015	From Node	Connect To	Distance/Cost		From Node	Connect To	Distance/Cost
1	Node1	Node2	150	16	Node14	Node13	120
2	Node2	Node3	100	17	Node13	Node19	120
3	Node3	Node4	210	18	Node19	Node22	120
4	Node4	Node5	200	19	Node22	Node28	120
5	Node5	Node6	190	20	Node28	Node29	190
6	Node6	Node16	120	21	Node29	Node21	120
7	Node16	Node17	120	22	Node21	Node20	120
8	Node17	Node24	120	23	Node20	Node12	120
9	Node24	Node25	110	24	Node12	Node11	100
10	Node25	Node26	160	25	Node11	Node10	160
11	Node26	Node27	170	26	Node10	Node9	100
12	Node27	Node23	2170	27	Node9	Node8	20
13	Node23	Node18	120	28	Node8	Node7	180
14	Node18	Node15	120	29	Node7	Node1	1160
15	Node15	Node14	180				
	Total	Minimal	Traveling	Distance	or Cost	=	6990
	(Result	from	Branch	and	Bound	Method)	

El grafo que corresponde a esta solución es el de la figura 10:

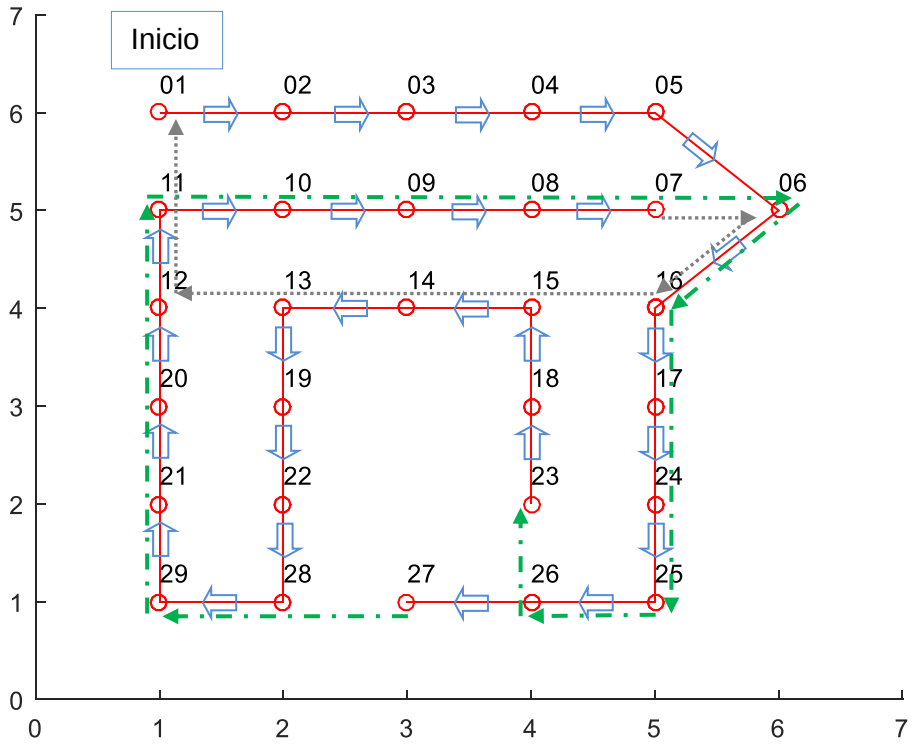


Fig. 10. Grafo de la solución TSP ruta uno; 29 nodos con aristas auxiliares 27-23 y 7-1.

Nuevamente se aprecia que el recorrido inicia en el nodo 1 y que la solución corresponde a un ciclo hamiltoniano, no euleriano, es decir se han recorrido todos los nodos pero no todas las aristas.

3.3.2. Búsqueda de herramientas, algoritmos y códigos referentes a la solución del TSP mediante AG's.

Se revisó el tema AG's en títulos de los autores Araujo & Cervigón (2009); Martí (2001); Taha (2012); Díaz, Laguna & Moscato en Díaz (1996); Maroto (2012); Russell & Norvag (2010); Srinivasan (2010); entre otros.

Asimismo se revisaron y probaron programas de AG's (tool kits de Matlab; Ison, Sitt & Trevisan; y otros en internet).

3.3.3. Algoritmos Genéticos aplicados al TSP. Bases para elaboración de software para solución del problema.

La siguiente es una descripción de los elementos del AG propuesto por Taha (2012) para el TSP, los cuales fueron tomados como base para codificar el programa en MATLAB:

1. Codificación de genes (nodos o vértices) del cromosoma (individuos).
   Esto se realiza mediante la representación numérica directa de los nodos del recorrido (por ejemplo *1-2-5-4-3-1*).
2. Población inicial.
   El primer paso es identificar las aristas que salen de cada nodo en la red, lo cual se hace a partir de la matriz de adyacencias.
   El siguiente paso es la construcción del recorrido, lo que se lleva a cabo comenzando en un nodo origen específico, agregando en la posición extrema a la derecha un nodo único no redundante, seleccionado de entre todos los que salen del último nodo agregado.
3. Creación de un hijo.
   Primeramente se seleccionan los dos mejores padres, mediante la evaluación de su función de su aptitud (longitud de recorrido).
   A continuación se realiza un intercambio de genes (cruce) para la creación de dos hijos mediante el procedimiento de cruce ordenado.
4. Mutación en los genes de los hijos.
   Se lleva a cabo mediante el intercambio de nodos de dos posiciones seleccionadas al azar (con excepción del nodo inicial u origen), con pequeña probabilidad de ocurrencia *(0.1)*. (pp. 423-424)

Esta descripción se complementa agregando un bucle que incluye la sustitución de los dos peores individuos de la generación anterior por los nuevos hijos creados, así como la evaluación de la nueva población para repetir los pasos 3 y 4 hasta que se cumpla una condición de terminación.

Goldberg (citado por Araujo & Cervigón, 2009) propuso el llamado Algoritmo Genético Simple, el cual utilizó para explicar con claridad el funcionamiento de los AG's en general. Se toma como referencia para presentar una comparación contra el algoritmo propuesto por Taha (2012) para el TSP (Tabla 28), mismo que se empleó para estructurar el programa en MATLAB de este trabajo:

Tabla 28. Comparativo AG Simple (Goldberg) vs. AG para el TSP (Taha).

Elemento	**AG Simple (Goldberg)**	**TSP (Taha)**
Representación de individuos	Binaria	Numérica directa (ejemplo *1-4-5-2-3*)
Generación de la población inicial	Aleatoria	Inicio en un nodo elegido al azar. Continuación del recorrido agregando sucesivamente un nodo de entre los que son adyacentes al último.
Selección	Por ruleta (proporcional a la adaptación relativa de los individuos)	Directa (los dos individuos mejor adaptados de la generación anterior).
Cruce	Monopunto	Cruce ordenado. Incluye armar el recorrido preferentemente con nodos adyacentes.
Mutación	Aleatoria bit a bit	Intercambio de dos genes (nodos) elegidos al azar.
Reemplazo	De los padres	Los dos individuos peor adaptados de la generación anterior.

3.4. Metodología para el rediseño de rutas de recolección - Recorrido Mínimo AG-TSP-Google.

La metodología desarrollada consta de las siguientes etapas:

1. Preparación de datos.
2. Medición de aristas con Daft Logic.
3. Generación de la matriz de adyacencias ponderada.
4. Ejecución del programa AG-TSP.
5. Verificación de la ruta obtenida.

Tomando en cuenta los elementos participantes en la metodología desarrollada se decidió nombrarla Recorrido Mínimo AG-TSP-Google.

La figura 11 esquematiza las etapas de esta metodología.

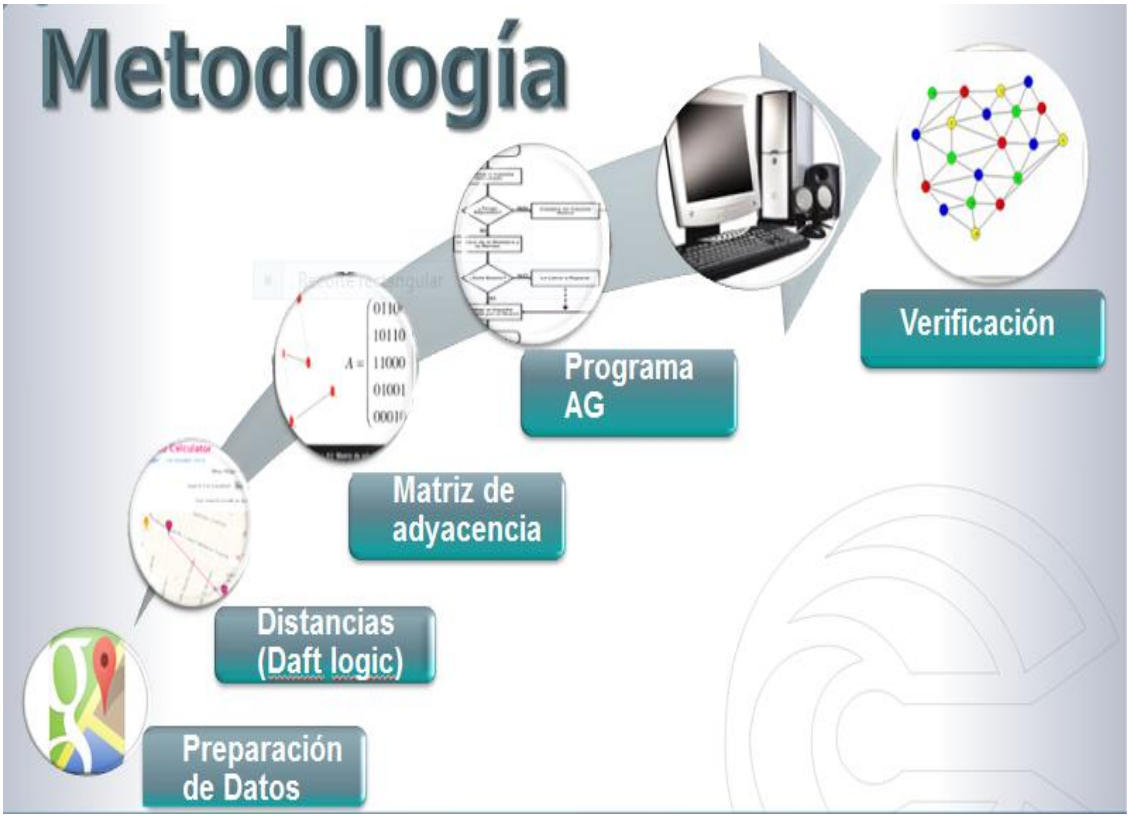


Fig. 11. Metodología para el rediseño de rutas de recolección.

A continuación se explican sus detalles.

3.4.1. Preparación de datos.

3.4.1.1 Empleando como referencia la información de la zona correspondiente a la ruta que se desea estudiar, básicamente sus delimitaciones, se descarga de Google Maps (2015) el mapa respectivo.

3.4.1.2 Se puede dibujar o marcar el grafo dirigido de la red en una copia del mapa obtenido de Google Maps (2015).

3.4.1.3 Se elabora la relación de calles, se determinan sus cruces y se etiquetan los nodos correspondientes.

3.4.1.4 Tomando en cuenta los sentidos de las calles se elabora la matriz de adyacencias unitaria. Esta matriz de adyacencias se genera en un archivo Excel, capturando *(1's)* en los cruces de las filas y columnas correspondientes a los nodos donde existe conexión entre nodos consecutivos. En la primera hoja de este archivo debe ir la matriz de adyacencias sin columnas, renglones ni ningún dato adicional a los que corresponden exclusivamente a las adyacencias. La matriz se puede desarrollar previamente en hojas internas del archivo, ya que normalmente para su elaboración son muy útiles, al menos, una columna y un renglón de identificación de los nodos.

3.4.1.5 Se nombra el archivo identificando la ruta correspondiente, asignándole un nombre, por ejemplo: *nombre de archivo con ruta.xlsx,* el cual va a ser empleado posteriormente en la etapa de ejecución del programa AG.

3.4.2. Medición de aristas con Daft Logic.

3.4.2.1 Se accede a la herramienta Daft Logic de Google Maps y se descarga el mapa correspondiente a la zona.

3.4.2.2 Se miden una a una las aristas del grafo para obtener las distancias correspondientes.

3.4.3. Generación de la matriz de adyacencias ponderada.

3.4.3.1 Con los datos anteriores se convierte la matriz de adyacencias unitaria en la matriz de adyacencias ponderada, sustituyendo los *(1's)* por los valores de las distancias previamente obtenidos.

3.4.3.2 Para el manejo de los casos en los que no hay adyacencia entre nodos, se agrega un valor muy alto, *9999*, en los elementos respectivos de la matriz.

3.4.4. Ejecución del programa AG.

3.4.4.1 Se alimenta la matriz de adyacencias ponderada registrando el nombre del archivo Excel en la parte superior del programa. La instrucción de lectura del archivo es:

d = xlsread('nombre de archivo con ruta.xlsx')

3.4.4.2 Se confirman o redefinen los parámetros. Esto se realiza en las líneas correspondientes al inicio del programa. Para fines ilustrativos se muestran valores pequeños en el caso del número de generaciones y el tamaño de la población. Generalmente se recomiendan valores bajos para la tasa de mutación:

Número de generaciones = 10

Tamaño de la población = 6 individuos (recorridos)

Tasa de probabilidad de mutación = 0.1

Estos parámetros se pueden modificar en función de los resultados que arroje el programa, tomando en cuenta que sus variaciones influirán en

la convergencia del programa así como en sus tiempos de ejecución. Esto dependerá naturalmente del tamaño de la red alimentada. Se probó que al menos para redes pequeñas y medianas (menos de 30 nodos) estos parámetros funcionan eficientemente para este programa.

3.4.4.3 El programa AG produce el número de generaciones especificado. Presenta los cromosomas o individuos formados en cada una de sus generaciones (iteraciones) mostrando el recorrido propuesto a través de la secuencia de genes o nodos para cada uno de ellos.

La 1ª generación presenta la población inicial creada aleatoriamente por el AG con la característica de que los nodos en cada individuo no se repiten salvo el nodo inicial u origen, condicionando el armado del recorrido a que exista adyacencia entre nodos, con excepción del último eslabón ya que muy probablemente no existe adyacencia entre el último nodo del recorrido construido y el nodo inicial y que, sin embargo, es necesario para cerrar el ciclo.

En general, el programa AG busca formar los recorridos usando sólo eslabonamientos dados por la adyacencia entre nodos.

Específicamente la operación de cruce se lleva a cabo incluyendo solamente nodos que cumplen con esta condición.

La operación de mutación puede generar aristas no existentes al hacer el intercambio de nodos de dos posiciones seleccionadas al azar (de un individuo), que no tengan adyacencia entre sí (llamadas aristas auxiliares), integrando, por tanto, valores muy altos *9999's* en la función objetivo (distancia). El programa en su desempeño recompone de manera natural la conformación correcta de los individuos al optimizar (minimizar) la función objetivo, lo cual se logra integrando sólo aristas correspondientes a nodos adyacentes que involucran valores pequeños en comparación con los *9999's.*

En la última generación, por las razones mencionadas más arriba, es posible que el programa mantenga eslabonamientos correspondientes a aristas no existentes. Esta es la causa por la cual el programa AG puede requerir su ejecución en dos fases. En la primera se genera una propuesta de recorrido que incluye una (o más) aristas no existentes (o auxiliares), es decir con valores muy altos *9999's.*

3.4.4.4 Lo anterior, en una segunda fase, da lugar al manejo de aristas auxiliares que se calculan por inspección aplicando el criterio del camino más corto, y cuyas distancias totales se introducen en la matriz de datos para efectuar una nueva corrida del programa.

El programa AG entrega el resultado final cuando se cumple la condición de terminación (generalmente el número de iteraciones o generaciones definido). El programa puede converger o no dependiendo de si presenta el mismo recorrido para todos sus individuos o cromosomas en la última generación (algunos de estos individuos pueden iniciar en nodos diferentes, lo cual no es relevante si el recorrido es el mismo). Si se requiere se puede volver a ejecutar el programa con un número mayor de generaciones para lograr su convergencia.

Finalmente, lo que se busca en esta segunda fase, es que no existan tramos con valores muy altos *9999.*

3.4.5. Verificación de la ruta obtenida.

3.4.5.1 Se puede marcar el grafo dirigido de la red correspondiente a la solución, en una copia del mapa de Google Maps, verificando los sentidos.

3.4.5.2 Con base en el recorrido obtenido y los datos de la matriz de adyacencias ponderada se confirma la distancia total. Esto puede realizarse fácilmente en una hoja de cálculo.

Capítulo 4. Resultados.

Basados en la metodología descrita en la sección 3.4, en este capítulo se presentan los resultados de su aplicación.

Primeramente se muestra su uso en una red de dimensiones pequeñas lo cual es de utilidad para indicar la metodología en detalle.

En segundo término se presenta una red mayor que sirve para evidenciar algunos aspectos observados durante la investigación.

Posteriormente se realiza un análisis y discusión de los aspectos detectados en las secciones anteriores.

Al final del capítulo se informa sobre los entregables producidos a lo largo de la investigación.

4.1. Aplicación de la metodología para el rediseño de rutas de recolección - Recorrido mínimo con AG-TSP-Google. Ruta uno; 16 nodos.

En esta sección se indica la aplicación de la metodología a 16 nodos (cruces de calles) de la ruta uno del municipio.

4.1.1. Preparación de datos.

A partir de la información de las delimitaciones de la zona se descargó de Daft Logic (2015) el mapa correspondiente a los 16 nodos de la ruta uno (Fig. 12).

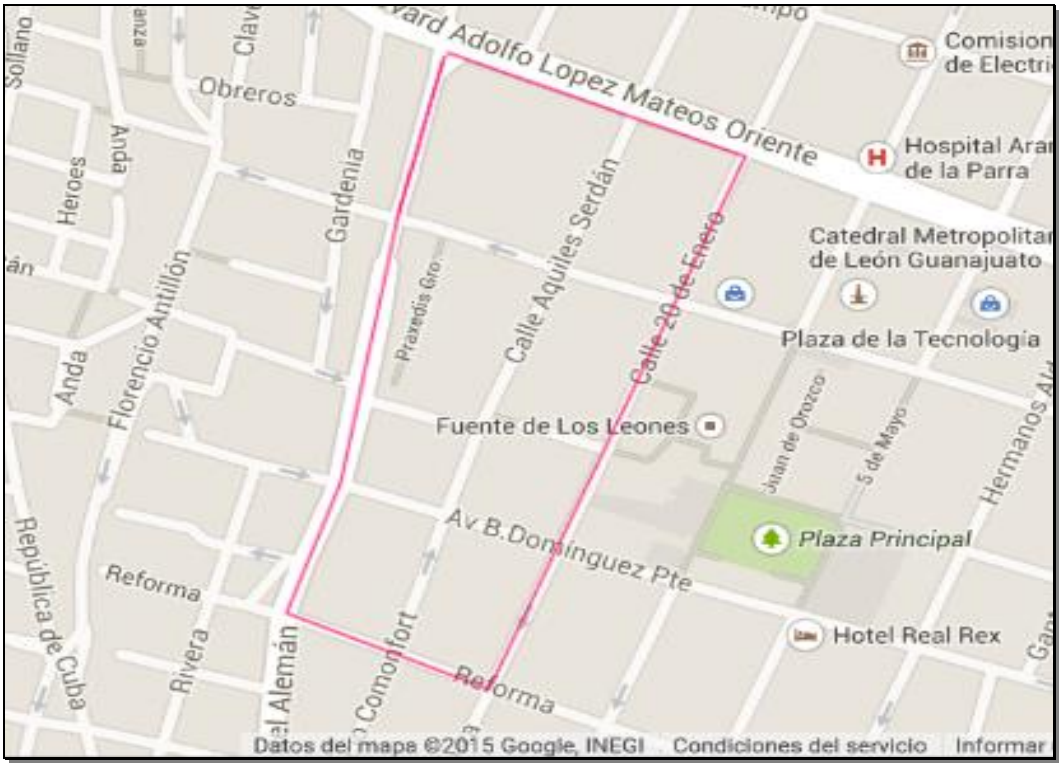


Fig. 12. Mapa de 16 nodos de la ruta uno (Daft Logic, 2015).

Se dibujó el grafo correspondiente de esta red (Fig. 13).

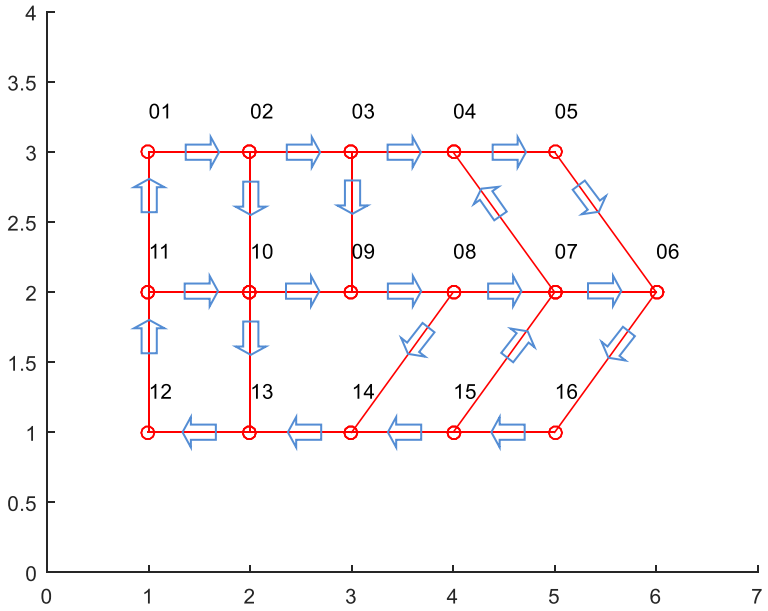


Fig. 13. Grafo de la ruta uno; 16 nodos.

Se elaboró la relación de calles y se determinaron los sentidos, restricciones y otras características de las vialidades (ver tabla 29).

Tabla 29. Relación de calles ruta uno.

	Calle	Sentido
1	Blvd. Miguel Alemán	Sur – Norte
2	Ignacio Comonfort / Aquiles Serdán	Sur - Norte
3	Justo Sierra / 20 de Enero	Norte - Sur
4	Reforma	Oriente - Poniente
5	Belisario Domínguez	Poniente - Oriente
6	Josefa Ortiz de Domínguez	Poniente - Oriente
7	Alvaro Obregón	Oriente - Poniente
8	Blvd. Adolfo López Mateos	Poniente – Oriente

Se definieron los cruces de calles y se etiquetaron los nodos (tabla 30).

Tabla 30. Etiquetado de nodos ruta uno.

Nodo	Calle	Esquina con
1	Blvd. M. Alemán	Reforma
2	Blvd. M. Alemán	Belisario Domínguez
3	Blvd. M. Alemán	Josefa Ortiz de Domínguez
4	Blvd. M. Alemán	Alvaro Obregón
5	Blvd. M. Alemán	Bvd. Adolfo López Mateos
6	I. Comonfort / Aquiles.Serdán	Bvd. Adolfo López Mateos
7	I. Comonfort / Aquiles.Serdán	Alvaro Obregón
8	I. Comonfort / Aquiles.Serdán	Josefa Ortiz de Domínguez
9	I. Comonfort / Aquiles.Serdán	Josefa Ortiz de Domínguez
10	I. Comonfort / Aquiles.Serdán	Belisario Domínguez
11	I. Comonfort / Aquiles.Serdán	Reforma
12	Justo Sierra / 20 Enero	Reforma
13	Justo Sierra / 20 Enero	Belisario Domínguez
14	Justo Sierra / 20 Enero	Josefa Ortiz de Domínguez
15	Justo Sierra / 20 Enero	Alvaro Obregón
16	Justo Sierra / 20 Enero	Bvd. Adolfo López Mateos

Se generó la matriz de adyacencias unitaria (tabla 31) a partir de los cruces de calles, tomando en cuenta los sentidos. Se utilizó un archivo Excel, el cual se nombró como: *MatrizAdyRuta1_16nodos.xlsx.*

Tabla 31. Matriz de adyacencias unitaria ruta uno.

		N			o			d			o			s			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
N	1		1														
	2			1							1						
	3				1					1							
o	4					1											
	5						1										
	6																1
d	7				1		1										
	8							1							1		
	9								1								
o	10									1				1			
	11	1									1						
	12											1					
s	13												1				
	14													1			
	15							1							1		
	16															1	

4.1.2. Medición de aristas con Daft Logic.

Con la herramienta Daft Logic (2015) de Google Maps se midieron las aristas del grafo para obtener las distancias. En la figura 14 se muestra la medición de la arista incidente en los nodos 1 y 2. El valor obtenido 149.823 se redondeó a $150\ m$. el cual se capturó en el elemento de la matriz que corresponde a la adyacencia entre los dos nodos (fila 1, columna 2), ver tabla 32. Así se continuó sucesivamente para la medición de todas las aristas.

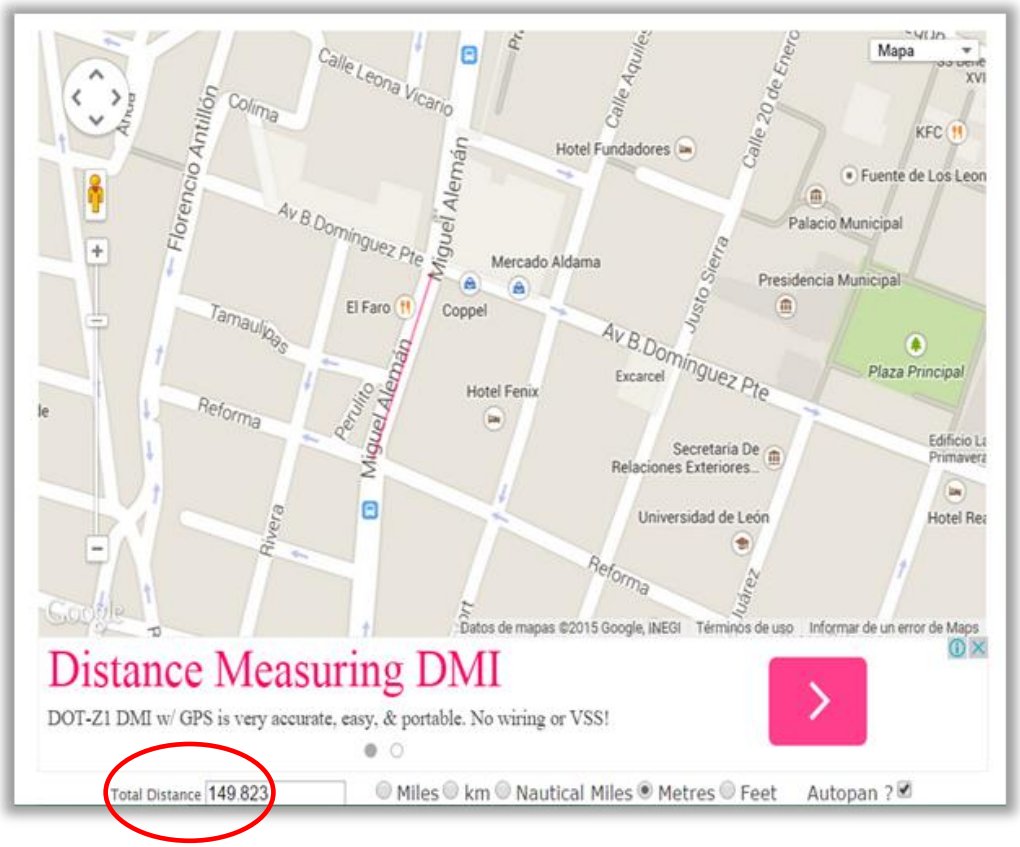


Fig. 14. Medición de la arista incidente entre los nodos *1* y *2,* ruta 1 (Daft Logic, 2015).

4.1.3. Generación de la matriz de adyacencias ponderada.

Con los datos anteriores se convirtió la matriz de adyacencias unitaria en matriz de adyacencias ponderada (tabla 32). Para el efecto se duplicó la matriz unitaria en otra hoja del archivo *MatrizAdyRuta1_16nodos.xlsx* dónde se substituyeron los *(1's)* por los valores de las distancias de las aristas en metros. Se incluyó el valor *9999* en los elementos de la matriz en los que no hay adyacencia entre nodos. Finalmente se copió esta hoja al principio del archivo, eliminando la primera fila y la primera columna, las cuales sirvieron para identificación de los nodos.

Tabla 32. Matriz de adyacencias ponderada ruta uno.

9999	0150	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	0100	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999	9999
9999	9999	9999	0210	9999	9999	9999	9999	0110	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	0200	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120
9999	9999	9999	0160	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	0180	9999	9999	9999	9999	9999	9999	0120	9999	9999
9999	9999	9999	9999	9999	9999	9999	0020	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	0110	9999	9999	9999
0100	9999	9999	9999	9999	9999	9999	9999	9999	0160	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0170	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120	9999	9999	9999
9999	9999	9999	9999	9999	9999	0120	9999	9999	9999	9999	9999	9999	0180	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0180	9999

4.1.4. Ejecución del programa AG.

Se alimentó la matriz de adyacencias ponderada al programa mediante el registro del nombre del archivo *MatrizAdyRuta1_16nodos.xlsx*, en la instrucción:

$$d = xlsread('MatrizAdyRuta1\_16nodos.xlsx')$$

Se definieron los siguientes parámetros del programa. Dado el tamaño de la red se establecieron valores pequeños en el número de generaciones y en el tamaño de la población. El correspondiente a la tasa de mutación se dejó también con un valor bajo como generalmente se maneja:

$$Número\ de\ generaciones = 10$$

$$Tamaño\ de\ la\ población = 6\ individuos\ (recorridos)$$

$$Tasa\ de\ probabilidad\ de\ mutación = 0.1$$

La corrida inicial del programa AG presentó los individuos formados en cada una de sus iteraciones. A continuación se analiza en detalle cada una de las diez generaciones.

En la tabla 33 se presenta el resultado correspondiente a la 1ª generación. En ella se aprecian los seis individuos generados por el programa los cuales corresponden a la población inicial creada aleatoriamente por el AG con la característica de que los nodos no se repiten salvo el nodo inicial. Podemos apreciar que hay dos pares de individuos idénticos el 1 y el 3, y el 5 y 6. Los individuos 2 y 4 son diferentes a los demás.

Para fines de análisis del desempeño del AG nos enfocaremos en el **individuo 6.**

En esta primera generación, el AG consideró al final del recorrido, para cerrar el ciclo, el arista 7-1 que no existe, ya que no hay adyacencia entre esos nodos. Por lo anterior, incluyó el valor 9999 en el acumulado, dando una longitud de recorrido de $12{,}179\ m$. Este valor es uno de los dos más altos de entre los seis individuos,

por lo que se convierte en un candidato para ser substituido en la siguiente generación, de acuerdo con los lineamientos manejados por el programa.

Tabla 33. Resultado de ejecución inicial del programa AG. 1ª generación.

Generación 1																		
Individuo	R e c o r r i d o																	distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	1	2	3	4	5	6	16	15	14	13	12	11	10	9	8	7	1	12,179
6	**1**	**2**	**3**	**4**	**s5**	**6**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**1**	**12,179**

En la 2ª generación (tabla 34), al aplicarse los procesos de selección y reproducción (cruce y mutación) se produjo un nuevo individuo 6 con características más deficientes que el inicial, al pasar de $12,179$ a $21,898\ m.$ en su longitud de recorrido, ya que incluyó dos aristas no existentes, la 15-10 y la 10-14. Nuevamente éste es candidato a ser sustituido dentro del proceso del AG, ya que presenta el valor más alto de entre los seis individuos.

Tabla 34. Resultado de ejecución inicial del programa AG. 2ª generación.

Generacion 2																		
Individuo	R e c o r r i d o																	distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**14**	**13**	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**15**	**10**	**14**	**21,898**

La 3ª generación (tabla 35), después de llevar a cabo el proceso del AG, presenta un individuo 6 con un peor resultado, *41,566 m.*, ya que en esta ocasión se incluyeron cuatro aristas inexistentes, la 13-2, la 16-12, la 1-15 y la 14-10. De nueva cuenta este individuo va a ser substituido en la siguiente iteración del AG, dado que presenta uno de los dos valores más grandes dentro del conjunto de individuos.

Tabla 35. Resultado de ejecución inicial del programa AG. 3ª generación.

generacion 3																		
Individuo	R e c o r r i d o																	Distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	16	15	14	13	12	11	1	5	6	4	2	10	3	51,445
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**10**	**13**	**2**	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**12**	**11**	**1**	**15**	**14**	**10**	**41,566**

El valor que presenta la 4ª generación, correspondiente a la función de adaptación del individuo 6, *21,828 m.*, es mejor que la anterior (ver tabla 36). Considera sin embargo las aristas 14-7 y 15-10 que son inexistentes. El mismo individuo 6 continua formando parte del par de individuos que van a ser sustituidos en la siguiente generación del proceso.

Tabla 36. Resultado de ejecución inicial del programa AG. 4ª generación.

generacion 4																		
Individuo	R e c o r r i d o																	distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**10**	**13**	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**14**	**7**	**4**	**5**	**6**	**16**	**15**	**10**	**21,828**

En la nueva iteración del proceso (5ª generación, tabla 37) se produjo un individuo con un nuevo deterioro en su función de adaptación (*31,627 m.*) ya que incluye tres aristas inexistentes, la 8-5, la 4-14 y la 14-10, por lo que el mismo individuo 6 continúa como candidato a mejorar en el siguiente proceso.

Tabla 37. Resultado de ejecución inicial del programa AG. 5ª generación.

Generacion 5																		
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	4	5	6	16	15	2	3	9	8	14	10	13	12	11	1	7	4	31,677
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**1**	**2**	**3**	**9**	**8**	**5**	**6**	**16**	**15**	**7**	**4**	**14**	**10**	**13**	**12**	**11**	**1**	**31,627**

Según la tabla 38 (6ª generación) el proceso del AG produjo un nuevo individuo 6 con peores características que el anterior ya que incluye cuatro aristas inexistentes (9-5, 14-10, 10-8 y 4-13) totalizando una longitud de recorrido de *41,736 m*. Por consiguiente, este individuo va a ser sustituido en la siguiente iteración.

Tabla 38. Resultado de ejecución inicial del programa AG. 6ª generación.

Generación 6																		
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	11	1	2	3	9	8	7	4	15	14	10	5	6	16	13	12	11	41,576
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**3**	**9**	**5**	**6**	**16**	**15**	**14**	**10**	**8**	**7**	**4**	**13**	**12**	**11**	**1**	**2**	**3**	**41,736**

El sustituto del individuo 6 permaneció en la 7ª generación con un nivel similar de longitud de recorrido (*41,576* m.) ya que consideró las aristas no existentes 4-14, 14-10, 10-5 y 15-13 (tabla 39). Junto con el individuo 1, el individuo 6 va ser substituido por otro en la siguiente ocasión.

Tabla 39. Resultado de ejecución inicial del programa AG. 7ª generación.

Generacion 7																		
Individuo	R e c o r r i d o																	Distancia
1	3	9	8	5	6	16	15	14	10	7	4	13	12	11	1	2	3	41,576
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**7**	**4**	**14**	**10**	**5**	**6**	**16**	**15**	**13**	**12**	**41,576**

En la octava generación, la nueva función de adaptación del individuo 6 desciende a *31,737 m.,* e incluye las aristas no existentes 14-12, 7-10 y 13-4 (tabla 40). El individuo 6 vuelve a quedar como candidato a ser removido y substituido en el siguiente proceso.

Tabla 40. Resultado de ejecución inicial del programa AG. 8ª generación.

Generacion 8																		
Individuo	R e c o r r i d o																	distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**15**	**14**	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**7**	**10**	**13**	**4**	**5**	**6**	**16**	**15**	**31,737**

En la siguiente ocasión (9ª generación) el programa produjo un individuo con mejores características para sustituir al seis, ya que sólo incluyó la arista inexistente (14-10), arrojando una longitud de recorrido de sólo $12{,}069\ m$. De hecho, este mismo valor se repitió en otros cuatro individuos. El individuo 6 ya no es candidato a priori para ser substituido en el siguiente ciclo, ya que al igual que los otros cuatro individuos tiene el mismo valor en su función de adaptación o de aptitud (tabla 41).

Tabla 41. Resultado de ejecución inicial del programa AG. 9ª generación.

Generacion 9																		
Individuo	R e c o r r i d o																	distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
6	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**15**	**14**	**10**	**13**	**12**	**11**	**1**	**2**	**3**	**12,069**

Finalmente el programa converge en la 10ª generación (tabla 42) entregando los mismos cromosomas (individuos). Aun cuando el individuo 2 inició en un nodo diferente no tiene relevancia ya que el recorrido es el mismo. El valor de la función de adaptación quedó en $12{,}069\ m$. incluyendo el arista inexistente 14-10.

Tabla 42. Resultado de ejecución inicial del programa AG. 10ª generación.

generacion 10																		
Individuo	R e c o r r i d o																	distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
5	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
6	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**15**	**14**	**10**	**13**	**12**	**11**	**1**	**2**	**3**	**12,069**

Dada la topología de la red, fue necesario alimentar en el programa la distancia de la arista entre los nodos 14 y 10 que no siendo adyacentes, fue requerida para cerrar el recorrido y terminar la ejecución del algoritmo. Se observa por inspección que este sub-recorrido que une estos dos nodos es: 14, 13, 12, 11 y 10, el cual suma una distancia total de $550\ m.$, misma que se agregó a la matriz de datos (tabla 43).

Tabla 43. Matriz de adyacencias ponderada ruta uno con arista auxiliar 14-10.

9999	0150	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	0100	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999	9999
9999	9999	9999	0210	9999	9999	9999	9999	0110	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	0200	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120
9999	9999	9999	0160	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	0180	9999	9999	9999	9999	9999	9999	0120	9999	9999
9999	9999	9999	9999	9999	9999	9999	0020	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	0110	9999	9999	9999
0100	9999	9999	9999	9999	9999	9999	9999	9999	0160	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0170	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	**0550**	9999	9999	0120	9999	9999	9999
9999	9999	9999	9999	9999	9999	0120	9999	9999	9999	9999	9999	9999	0180	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0180	9999

Con la matriz de adyacencias actualizada se ejecutó nuevamente el programa AG. El programa convergió en ocho generaciones ya que entregó los mismos cromosomas (individuos) desde esa iteración. El resultado de la ejecución final (10ª generación) fue $2{,}620\ metros$ y se observa el recorrido correspondiente en la tabla 44.

Tabla 44. Resultado de ejecución final del programa AG. 10ª generación.

Generación 10																		
Individuo	R e c o r r i d o																	distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
2	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
4	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
5	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
6	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620

4.1.5. Verificación de la ruta obtenida.

Se representó la solución obtenida mediante el siguiente grafo (Fig. 15).

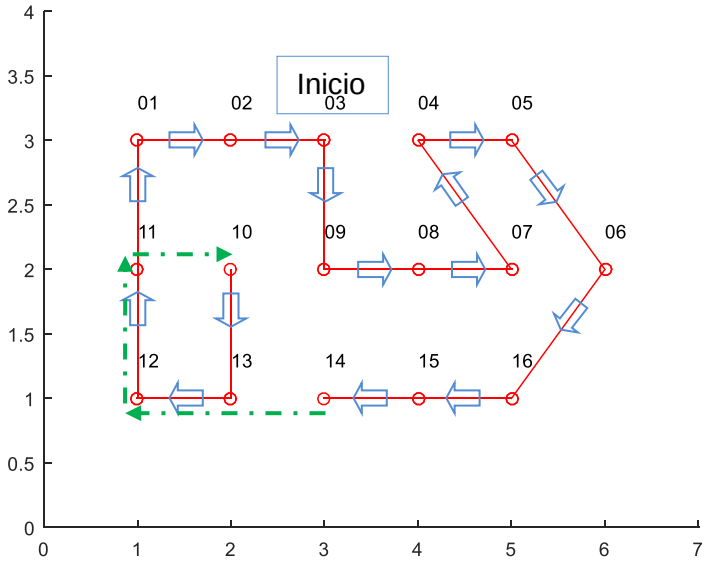


Fig. 15. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 14-10.

Como se puede apreciar, de acuerdo con la definición del TSP, el recorrido es cerrado. Sin embargo, para que esto pudiera lograrse fue necesario considerar el arista auxiliar 14-10 representada con línea punteada.

Siguiendo el recorrido obtenido y con los datos de la matriz de adyacencias ponderada se verifica la distancia total (tabla 45).

Tabla 45. Distancia total del recorrido.

Arista	Delnodo	Al nodo	Distancia
1	3	9	110
2	9	8	20
3	8	7	180
4	7	4	160
5	4	5	200
6	5	6	190
7	6	16	120
8	16	15	180
9	15	14	180
10	14	10	550
11	10	13	110
12	13	12	170
13	12	11	100
14	11	1	100
15	1	2	150
16	2	**3**	100
		suma	2620

Al calcularse y agregarse el dato correspondiente a la arista auxiliar 14-10 se obtuvo el resultado del recorrido igual a $2{,}620\ m$. el cual se considera correcto, mismo que fue validado mediante varias ejecuciones del programa, lo cual se discute en la siguiente sección.

Se puede considerar válido iniciar el recorrido en un nodo diferente al que originalmente propuso el programa AG en su solución, lo cual dependerá de aspectos operativos que así lo recomienden. Por ejemplo, el mismo recorrido puede ser iniciado y terminado en el nodo 1 que se encuentra ubicado en una esquina. Esto nos ejemplifica los casos en los que iniciar en un nodo diferente podría conducir a un recorrido más práctico y/o menos costoso.

4.1.6. Otros resultados del recorrido – Casos básicos.

La ejecución reiterada del programa arroja diferentes resultados para el recorrido propuesto. El análisis de estos recorridos permite concluir que existen sólo algunos recorridos básicos para esta red, aun cuando se presentan variantes relativas al nodo inicial.

Además del que se analizó en la subsección anterior, al cual enumeraremos caso 1, a continuación se presentan los que se identificaron como recorridos básicos adicionales.

La tabla 46 muestra un recorrido inicial generado por el programa, nombrado aquí como **caso básico 2**. En este caso la propuesta de recorrido incluye el arista inexistente 7-10. Si esta arista se calcula por fuera y se introduce el valor correspondiente (que asciende a $1{,}220$ m.) en la matriz de datos del programa, el resultado final del recorrido resulta ser de $3{,}220\ m.$, el cual es mayor que el obtenido como resultado final ($2{,}620\ m.$) del apartado anterior.

Tabla 46. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 2.

Generacion 10																		
Individuo	R	e	c	o	r	r	i	d	o									distancia
1	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
2	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
3	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
4	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
5	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
6	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999

Se presenta el grafo correspondiente al caso básico 2 en la figura 16.

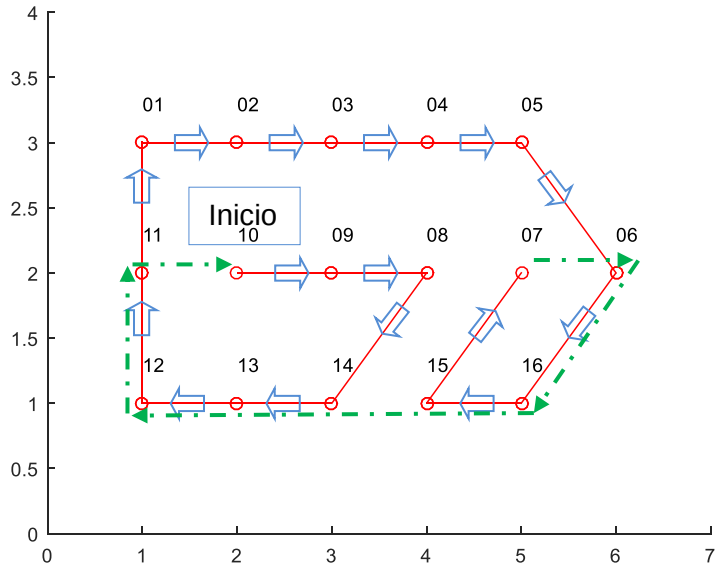


Fig. 16. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 7-10.

Para entregar un circuito cerrado, el programa AG consideró el arista auxiliar 7-10 al final (línea punteada).

La tabla 47 muestra otro recorrido inicial generado por el programa (**caso básico 3**). Incluye el arista inexistente 3-10. Cabe resaltar al individuo 4 que muestra un recorrimiento en la posición de sus genes pero que sin embargo representa al mismo individuo de los otros renglones. Si se calcula el arista auxiliar 3-10 que corresponde a $800\ m.$ y se introduce en la matriz de datos del programa, el resultado final es de $2{,}870\ m.$ también mayor que el obtenido como mejor resultado final ($2{,}620\ m$).Se observa el grafo del caso básico 3 (fig. 17).

Tabla 47. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 3.

Generacion 10																		
Individuo	R e c o r r i d o																	distancia
1	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
2	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
3	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
4	10	9	8	7	4	5	6	16	15	14	13	12	11	1	2	3	10	12,069
5	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
6	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069

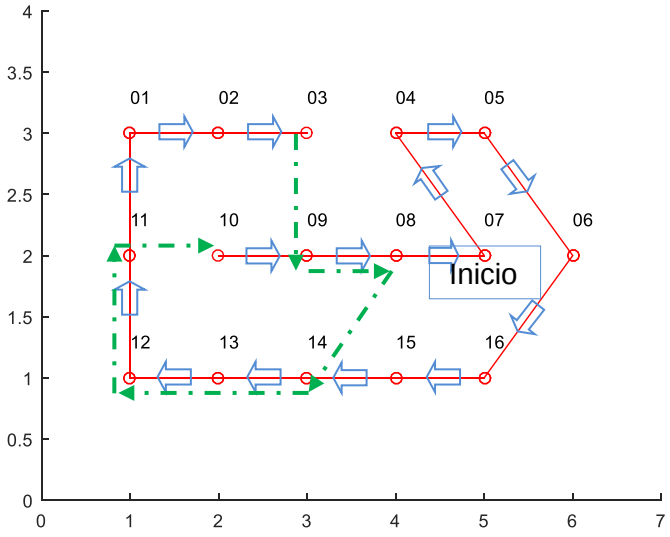


Fig. 17. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 3-10.

En la tabla 48 se presenta otro resultado inicial diferente (**caso básico 4**). En este caso el arista inexistente viene siendo la 7-3, la cual vale $1{,}410\ m.$, de tal manera que el nuevo resultado final sería $3{,}530\ m.$, mayor que el mejor ($2{,}620\ m.$).

Tabla 48. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 4.

Generacion 10																		
Individuo	R	e	c	o	r	r	i	d	o									distancia
1	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
2	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
3	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
4	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
5	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
6	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119

La figura 18 muestra el grafo correspondiente al caso básico 4. En este caso el cierre del recorrido nuevamente se da al final, precisamente con el arista auxiliar 7-3.

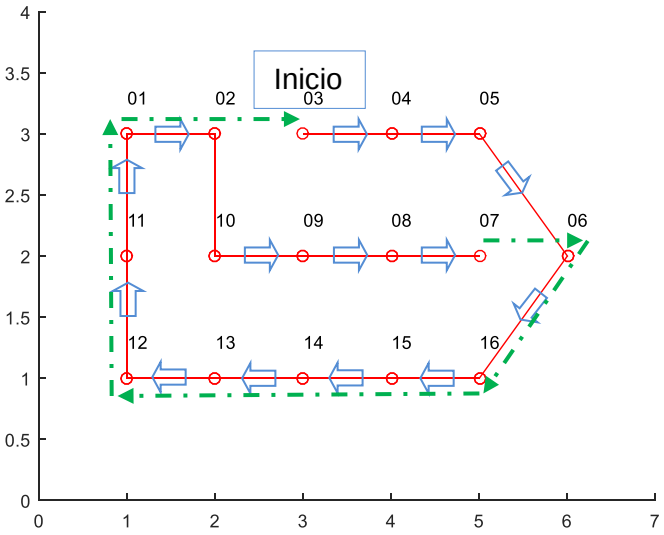


Fig. 18. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 7-3.

El **caso básico 5** se muestra en la tabla 49. Contempla el arista inexistente 10-3, con valor de *730 m.* El nuevo resultado final viene siendo *3,170 m.* Se puede observar el grafo correspondiente en la figura 19.

Tabla 49. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 5.

Generacion 10																		
Individuo	R e c o r r i d o																	distancia
1	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
2	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
3	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
4	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
5	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
6	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079

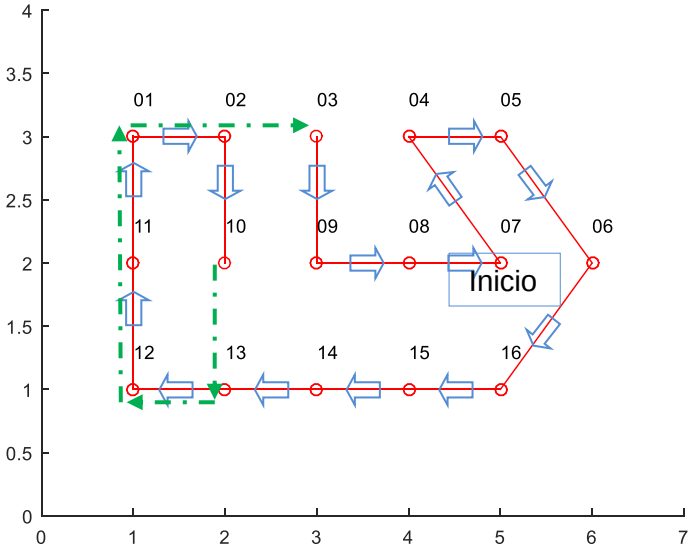


Fig. 19. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 10-3.

Resumiendo lo anterior en la tabla 50, se puede discernir que el recorrido inicial que produce el recorrido mínimo corresponde al caso 1 presentado en la subsección 4.1.

Tabla 50. Diferentes resultados del recorrido. Casos básicos.

Caso Básico	Recorrido																	Distancia inicial *m.*	Arista aux. *m.*	Recorr. final *m.*
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	**12,069**	**550**	**2,620**
2	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999	1,220	3,220
3	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069	800	2,870
4	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119	1,410	3,530
5	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079	730	2,810

Sin embargo, hay que considerar la posibilidad de elegir alguno de los otros recorridos de acuerdo con criterios operativos que sea necesario tomar en cuenta. Por ejemplo, que la primera recolección deba efectuarse en un punto (nodo) específico de la ruta. O bien algunas de las recolecciones deban respetar un cierto orden por diversos motivos (v.gr. horarios pactados, aspectos viales, etc.), que hagan recomendable elegir una ruta diferente a aquella con el recorrido mínimo.

4.2. Aplicación de la metodología para el rediseño de rutas de recolección - Recorrido mínimo con AG-TSP-Google. Ruta 16; 35 nodos.

Se presenta la aplicación de la metodología a la ruta 16 del municipio. Esta zona además de ser de grandes proporciones, de acuerdo con su descripción presentada en la sección 3.2, presenta algunas singularidades. Por lo anterior, y debido a la complejidad computacional (tiempo de ejecución), se presentan en esta sección los resultados correspondientes a su aplicación a 35 nodos de ella. Esto involucra el manejo de 58 aristas incidentes entre los cruces de las calles correspondientes.

Cabe hacer mención de que se logró correr, con éxito, una versión de 49 nodos. Sin embargo, se considera que gracias a la aleatoriedad el resultado fue encontrado muy rápidamente, lo cual no es usual, por lo que no se tomó como base para hacer la descripción que viene a continuación al no ser práctico el hacer las réplicas necesarias.

También es importante mencionar que se obtuvo una corrida con una ruta de 104 nodos, pero para ello se numeraron los nodos de una manera adecuada que facilitara el armado de los cromosomas de la población inicial, lo cual en una situación práctica no es factible poder hacer. Este ejercicio tuvo como propósito específico mostrar la capacidad del programa de generar un recorrido inicial con redes que faciliten su integración sin importar su tamaño, sino más bien la disposición de los elementos correspondientes en la matriz de adyacencias.

4.2.1. Preparación de datos.

En la ilustración 20 se muestran los límites correspondientes a los 35 nodos de la zona estudiada (ruta 16) (Daft Logic, 2015).

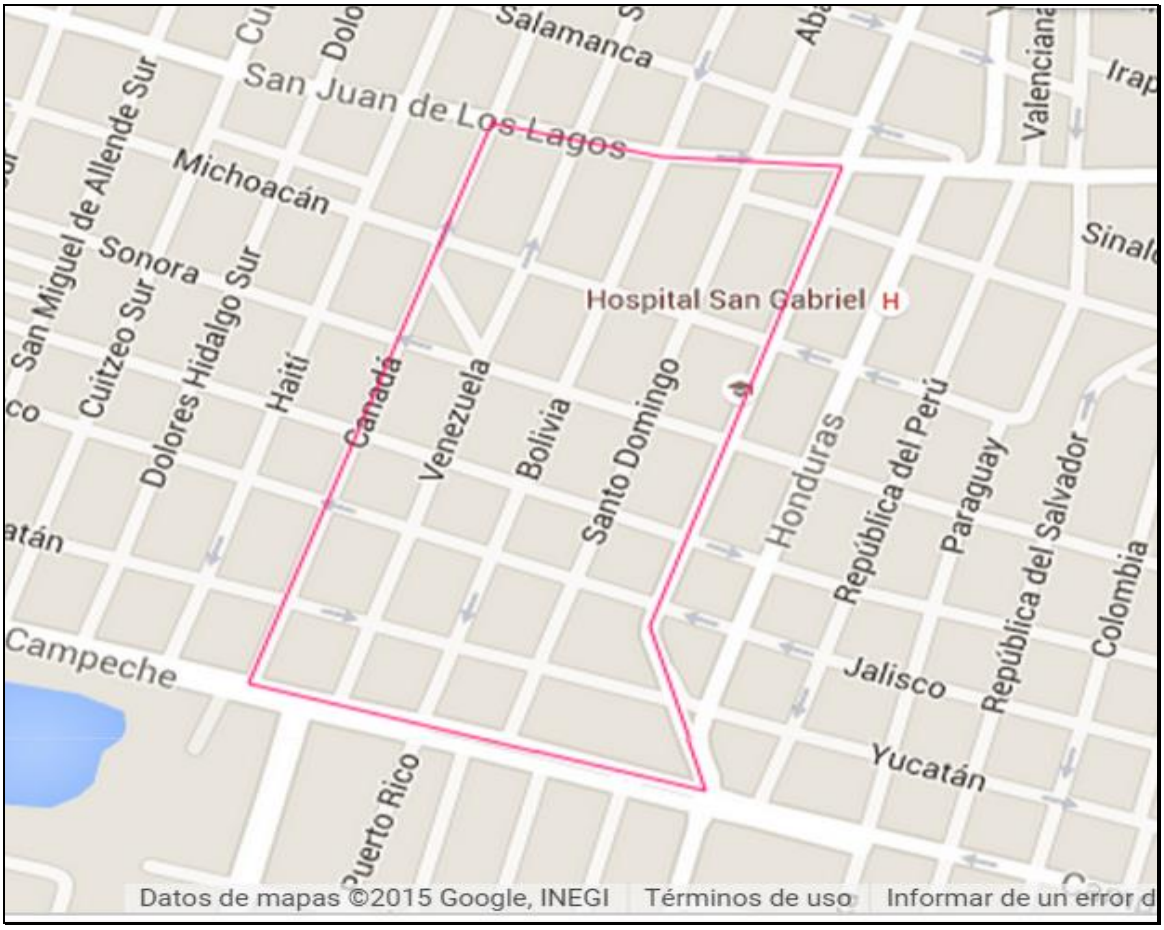


Fig. 20. Mapa ruta 16, 35 nodos (Daft Logic, 2015).

Se dibujó el grafo de la red (Fig. 21).

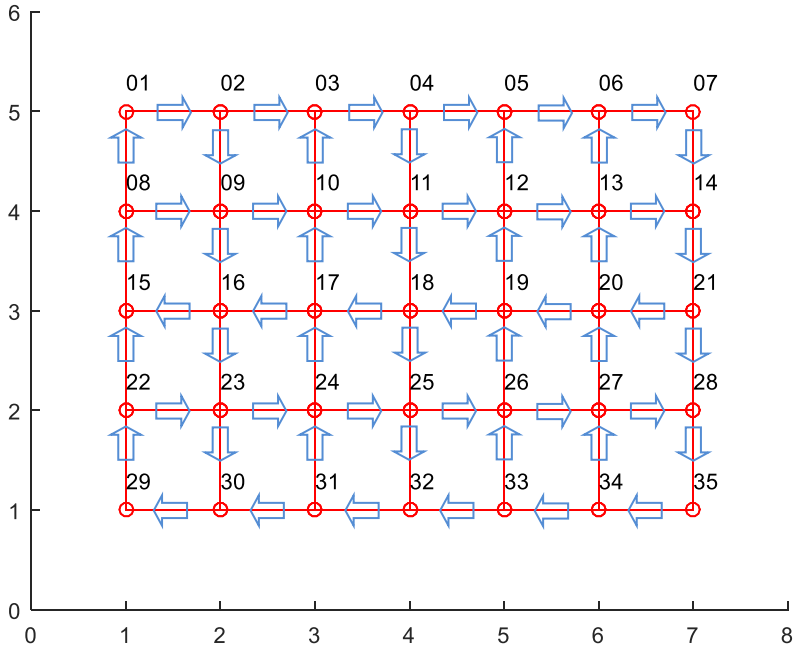


Fig. 21. Grafo de la ruta16, 35 nodos.

Se elaboró la relación de calles y se determinaron los sentidos, restricciones y otras características de las vialidades (tabla 51).

Tabla 51. Relación de calles. Ruta 16, 35 nodos.

Calle	Sentido
Canadá	Sur - Norte
Venezuela	Sur - Norte
Bolivia	Norte - Sur
Santo Domingo	Sur - Norte
Guatemala	Norte - Sur
Campeche	Oriente - Poniente
Yucatán	Poniente - Oriente
Jalisco	Oriente - Poniente
Zacatecas	Poniente - Oriente
Sonora	Oriente - Poniente
Michoacán	Oriente - Poniente
San Juan de los L.	Poniente - Oriente

Se definieron los cruces de calles y se etiquetaron los nodos (tabla 52).

Tabla 52. Etiquetado de nodos. Ruta 16, 35 nodos.

Nodo	Calle	Esquina con
1	Canadá	Campeche
2	Canadá	Yucatán
3	Canadá	Jalisco
4	Canadá	Zacatecas
5	Canadá	Sonora
6	Canadá	Michoacán
7	Canadá	San Juan de los L.
8	Venezuela	Campeche
9	Venezuela	Yucatán
10	Venezuela	Jalisco
11	Venezuela	Zacatecas
12	Venezuela	Sonora
13	Venezuela	Michoacán
14	Venezuela	San Juan de los L.
15	Bolivia	Campeche
16	Bolivia	Yucatán
17	Bolivia	Jalisco
18	Bolivia	Zacatecas
19	Bolivia	Sonora
20	Bolivia	Michoacán
21	Bolivia	San Juan de los L.
22	Santo Domingo	Campeche
23	Santo Domingo	Yucatán
24	Santo Domingo	Jalisco
25	Santo Domingo	Zacatecas
26	Santo Domingo	Sonora
27	Santo Domingo	Michoacán
28	Santo Domingo	San Juan de los L.
29	Guatemala	Campeche
30	Guatemala	Yucatán
31	Guatemala	Jalisco
32	Guatemala	Zacatecas
33	Guatemala	Sonora
34	Guatemala	Michoacán
35	Guatemala	San Juan de los L.

Se generó la matriz de adyacencias unitaria (tabla 53) a partir de los cruces de calles, tomando en cuenta los sentidos.

Tabla 53. Matriz de adyacencias unitaria. Ruta 16, 35 nodos.

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

4.2.2. Medición de aristas con Daft Logic.

Con la herramienta Daft Logic (2015) se midieron las aristas del grafo para obtener las distancias entre cruces de calles. Los valores obtenidos se capturaron en los elementos de la matriz que corresponden a las adyacencias entre nodos.

4.2.3. Generación de la matriz de adyacencias ponderada.

Con los datos anteriores se convirtió la matriz de adyacencias unitaria en matriz de adyacencias ponderada (tabla 54). Se incluyó el valor 9999 en los elementos de la matriz en los que no hay adyacencia entre nodos.

Tabla 54. Matriz de adyacencias ponderada. Ruta 16, 35 nodos. *

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2	3	4	5
999	82	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	95	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	72	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	93	999	999	999	999	999	68	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	100	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	119	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
72	999	999	999	999	999	999	999	74	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	95	999	999	999	999	999	71	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	71	999	999	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	95	999	999	999	999	999	72	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	72	999	999	999	999	999	999	999	98	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	68	999	999	999	999	999	999	999	126	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	70	999	999	999	999	999	999	69	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	69	999	999	999	999	999	90	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	70	999	999	999	999	999	999	68	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	66	999	999	999	999	95	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	73	999	999	999	999	999	98	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	135	999	999	999	999	999	999	72	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	88	999	999	999	999	107	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	69	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	95	999	999	999	999	74	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	73	999	999	999	999	999	999	999	98	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	157	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	73
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	157	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	85	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	70	999	999	999	999	999	91	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	68	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	66	999	999	999	999	96	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	68	999	999	999	98	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	176	999	999	999	999

* Para mejorar la visibilidad en esta tabla, se sustituyeron los valores *9999* por *999.*

4.2.4. Ejecución del programa AG.

Se alimentó la matriz de adyacencias ponderada al programa.

Se definieron los parámetros del programa. Se establecieron 20 generaciones para este tamaño de red. Se mantuvo en 6 el tamaño de la población. El valor correspondiente a la tasa de mutación se dejó bajo, como generalmente se maneja (variando entre 0.1% y 0.2%):

$$Número\ de\ generaciones = 20$$

$$Tamaño\ de\ la\ población = 6\ individuos\ (recorridos)$$

$$Tasa\ de\ probabilidad\ de\ mutación = 0.2$$

La corrida inicial del programa AG presentó los individuos formados en cada una de sus iteraciones. En la tabla 55 se presenta el resultado correspondiente a la 1ª generación. En ella se aprecian los seis individuos generados por el programa los cuales corresponden a la población inicial creada aleatoriamente por el AG. Todos los individuos son diferentes.

Tabla 55. Resultado ejecución inicial programa AG. 1ª generación.

Generación 1																		
Individuo	Recorrido (nodos)																	
1	13	14	21	28	35	34	33	26	27	20	19	18	25	32	31	24	17	16
2	5	6	7	14	21	20	19	18	17	16	23	24	25	26	27	28	35	34
3	5	6	7	14	21	28	35	34	33	26	27	20	19	18	25	32	31	24
4	11	12	13	6	7	14	21	20	19	18	25	26	27	28	35	34	33	32
5	5	6	7	14	21	28	35	34	33	32	31	30	29	22	15	8	1	2
6	19	18	25	32	31	30	29	22	23	24	17	16	15	8	1	2	9	10

(continuación)																	
Recorrido																	Distancia
23	30	29	22	15	8	1	2	9	10	3	4	11	12	5	6	7	13	12,959
33	32	31	30	29	22	15	8	1	2	9	10	3	4	11	12	13	5	13,132
17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	12	13	5	12,930
31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	5	11	13,123
9	16	23	24	25	26	27	20	19	18	17	10	3	4	11	12	13	5	12,938
3	4	11	12	5	6	7	14	21	28	35	34	33	26	27	20	13	19	12,881

En esta primera generación que corresponde a la población inicial, el AG consideró, para cerrar ciclos, aristas que no existen, ya que no hay adyacencia entre algunos nodos indicados. Por lo anterior, incluyó valores 9999 en los acumulados, generando longitudes de recorrido del orden de entre $12{,}000$ y $13{,}000\,m$.

Aun cuando esta corrida del programa se definió para 20 generaciones, el programa convergió desde la 5ª generación. Se muestran los resultados de la 20ª generación en la tabla 56.

Tabla 56. Resultado de ejecución inicial del programa AG. 20ª generación.

Generación 20

Individuo	Recorrido (nodos)																																				Distancia
1	19	18	25	32	31	30	29	22	23	24	17	16	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	33	26	27	20	13	19	12,881
2	19	18	25	32	31	30	29	22	23	24	17	16	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	33	26	27	20	13	19	12,881
3	19	18	25	32	31	30	29	22	23	24	17	16	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	33	26	27	20	13	19	12,881
4	19	18	25	32	31	30	29	22	23	24	17	16	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	33	26	27	20	13	19	12,881
5	19	18	25	32	31	30	29	22	23	24	17	16	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	33	26	27	20	13	19	12,881
6	19	18	25	32	31	30	29	22	23	24	17	16	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	33	26	27	20	13	19	12,881

El valor de la función de adaptación quedó en *12,881 m*. Para cerrar ciclo el programa consideró la arista inexistente 13-19.

Se alimentó al programa la distancia de la arista auxiliar entre los nodos 13 y 19. Para el efecto, se observó por inspección que el sub-recorrido que une estos dos nodos bajo el criterio de la distancia mínima, es: 13, 14, 21, 20 y 19, el cual representa una distancia de $430\ m.$, misma que se agregó a la matriz de datos (tabla 57).

Tabla 57. Matriz de adyacencias ponderada con arista auxiliar **13 -19.** Ruta 16, 35 nodos. *

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2	3	4	5
999	82	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	95	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	72	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	93	999	999	999	999	999	68	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	100	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	119	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
72	999	999	999	999	999	999	999	74	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	95	999	999	999	999	999	71	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	71	999	999	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	95	999	999	999	999	999	72	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	72	999	999	999	999	999	999	98	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	68	999	999	999	999	999	999	126	999	999	999	999	430	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	70	999	999	999	999	999	999	69	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	69	999	999	999	999	999	90	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	70	999	999	999	999	999	999	68	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	66	999	999	999	999	95	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	73	999	999	999	999	999	98	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	135	999	999	999	999	999	999	999	72	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	999	70	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	88	999	999	999	999	107	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	69	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	95	999	999	999	999	74	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	73	999	999	999	999	999	999	98	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	71	999	999	999	999	999	999	157	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	73
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	157	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	85	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	70	999	999	999	999	999	91	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	68	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	66	999	999	999	999	96	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	68	999	999	999	999	98	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	176	999

* Para mejorar la visibilidad en esta tabla, se sustituyeron los valores *9999* por *999.*

Con la matriz de adyacencias actualizada se ejecutó nuevamente el programa AG. El resultado de la ejecución final arrojó un valor de recorrido total de *3,309 m*., según se observa en la tabla 58.

Tabla 58. Resultado de ejecución final del programa AG. 20ª generación.

Generación	20																	
Individuo	Recorrido (nodos)																	
1	11	18	25	26	27	20	13	19	12	5	6	7	14	21	28	35	34	33
2	11	18	25	26	27	20	13	19	12	5	6	7	14	21	28	35	34	33
3	11	18	25	26	27	20	13	19	12	5	6	7	14	21	28	35	34	33
4	11	18	25	26	27	20	13	19	12	5	6	7	14	21	28	35	34	33
5	11	18	25	26	27	20	13	19	12	5	6	7	14	21	28	35	34	33
6	11	18	25	26	27	20	13	19	12	5	6	7	14	21	28	35	34	33

(continuación)																		
Recorrido																		Distancia
32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	3,309
32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	3,309
32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	3,309
32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	3,309
32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	3,309
32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	3,309

Se representó la solución obtenida mediante el siguiente grafo (Fig. 22).

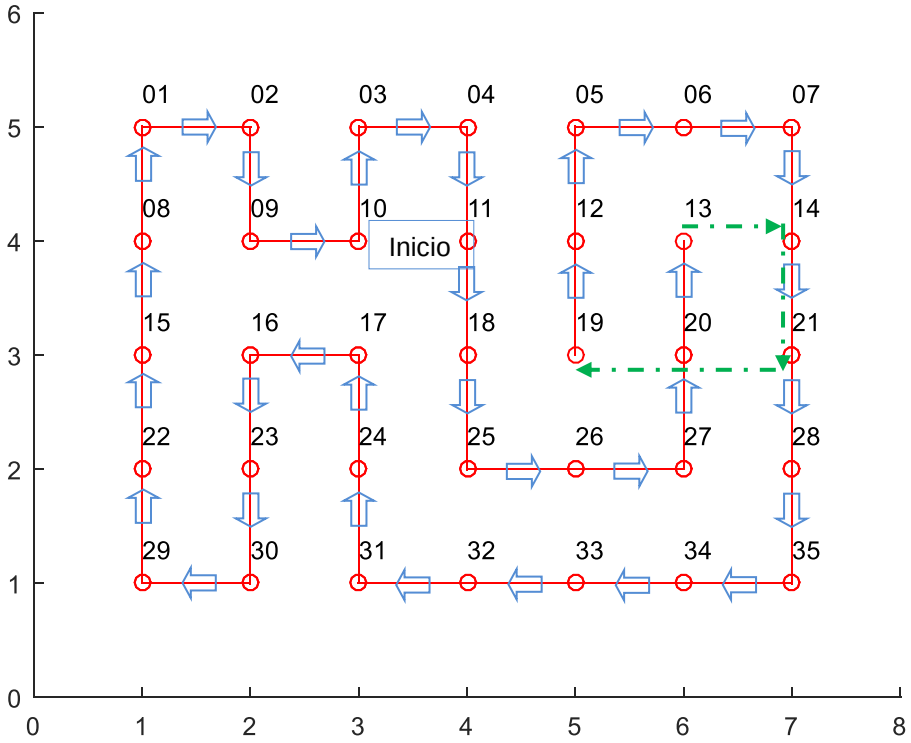


Fig. 22. Grafo de la solución TSP ruta 16; con arista auxiliar **13 - 19**.

Como se puede apreciar, de acuerdo con la definición del TSP, el recorrido es cerrado. Para que este recorrido pudiera lograrse fue necesario considerar el arista auxiliar 13-19 representada con línea punteada.

4.2.5. Verificación de la ruta obtenida.

Siguiendo el recorrido obtenido y con los datos de la matriz de datos se verifica la distancia total (tabla 59).

Tabla 59. Distancia total del recorrido.

Arista	Del nodo	Al nodo	Distancia
1	11	18	72
2	18	25	68
3	25	26	95
4	26	27	98
5	27	20	71
6	20	13	73
7	13	19	430
8	19	12	66
9	12	5	72
10	5	6	100
11	6	7	119
12	7	14	71
13	14	21	71
14	21	28	72
15	28	35	73
16	35	34	176
17	34	33	98
18	33	32	96
19	32	31	68
20	31	24	70
21	24	17	71
22	17	16	90
23	16	23	69
24	23	30	107
25	30	29	85
26	29	22	157
27	22	15	71
28	15	8	70
29	8	1	72
30	1	2	82
31	2	9	70
32	9	10	95
33	10	3	71
34	3	4	72
35	4	11	68
		suma	3309

Al calcularse y agregarse el dato correspondiente a la arista auxiliar 13-19 se obtuvo el resultado del recorrido igual a $3{,}309\ m$. el cual se considera correcto, y el mejor de entre varios intentos que se llevaron a cabo.

4.2.6. Otros resultados del recorrido.

En esta sección se presenta otro resultado de la aplicación del programa AG a la ruta 16, 35 nodos.

Se corrió el programa con los mismos parámetros descritos en la sección anterior. En la ejecución inicial se produjo el siguiente recorrido, el cual corresponde a la 20ª generación (tabla 60).

Tabla 60. Resultado de ejecución inicial del programa AG. 20ª generación.

Generación	20																																				
Individuo	Recorrido (nodos) (continuación)																																				Distancia
1	33	26	19	18	25	32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	27	20	13	33	12,824
2	33	26	19	18	25	32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	27	20	13	33	12,824
3	33	26	19	18	25	32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	27	20	13	33	12,824
4	33	26	19	18	25	32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	27	20	13	33	12,824
5	33	26	19	18	25	32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	27	20	13	33	12,824
6	33	26	19	18	25	32	31	24	17	16	23	30	29	22	15	8	1	2	9	10	3	4	11	12	5	6	7	14	21	28	35	34	27	20	13	33	12,824

En este resultado se observa convergencia, ya que se trata del mismo recorrido para los seis individuos. La arista auxiliar que generó el programa para cerrar el ciclo fue la 13–33. Esta auxiliar se cubre con la secuencia de nodos 13, 14, 21, 28, 35, 34, 33, cuya distancia total asciende a $616\ m$. Esta distancia se capturó en la matriz de adyacencias (ver tabla 61).

Tabla 61. Matriz de adyacencias ponderada con arista auxiliar **13 -33**. Ruta 16, 35 nodos. *

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2	3	4	5
999	**82**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	**95**	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	**72**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	**93**	999	999	999	999	999	**68**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	**100**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	**119**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
72	999	999	999	999	999	999	999	**74**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	**95**	999	999	999	999	999	**71**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	**71**	999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	**95**	999	999	999	999	999	**72**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	**72**	999	999	999	999	999	999	999	**98**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	**68**	999	999	999	999	999	999	999	**126**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**616**	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	**69**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	**69**	999	999	999	999	999	**90**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	**68**	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	**66**	999	999	999	999	999	**95**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	**73**	999	999	999	999	999	**98**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**135**	999	999	999	999	999	999	999	**72**	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**88**	999	999	999	999	999	**107**	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	**69**	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**95**	999	999	999	999	**74**	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**73**	999	999	999	999	999	999	999	**98**	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	**157**	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**73**	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**157**	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**85**	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**70**	999	999	999	999	**91**	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**68**	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**66**	999	999	999	**96**	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**68**	999	999	999	**98**	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**176**	999	

* Para mejorar la visibilidad en esta tabla, se sustituyeron los valores *9999* por *999.*

Posteriormente se ejecutó nuevamente el programa, produciendo el resultado mostrado en la tabla 62.

Tabla 62. Resultado de ejecución final del programa AG. 20ª generación.

Generación	20																																				
Individuo	Recorrido (nodos)																		(continuación) Recorrido																		Distancia
1	13	33	32	31	30	29	22	15	8	1	2	9	16	23	24	17	10	3	4	11	18	25	26	19	12	5	6	7	14	21	28	35	34	27	20	13	3,397
2	13	33	32	31	30	29	22	15	8	1	2	9	16	23	24	17	10	3	4	11	18	25	26	19	12	5	6	7	14	21	28	35	34	27	20	13	3,397
3	13	33	32	31	30	29	22	15	8	1	2	9	16	23	24	17	10	3	4	11	18	25	26	19	12	5	6	7	14	21	28	35	34	27	20	13	3,397
4	13	33	32	31	30	29	22	15	8	1	2	9	16	23	24	17	10	3	4	11	18	25	26	19	12	5	6	7	14	21	28	35	34	27	20	13	3,397
5	13	33	32	31	30	29	22	15	8	1	2	9	16	23	24	17	10	3	4	11	18	25	26	19	12	5	6	7	14	21	28	35	34	27	20	13	3,397
6	13	33	32	31	30	29	22	15	8	1	2	9	16	23	24	17	10	3	4	11	18	25	26	19	12	5	6	7	14	21	28	35	34	27	20	13	3,397

El resultado de la ejecución final fue de *3,397 m*. Aunque este valor es un buen resultado, es mayor que el obtenido en la sección anterior *3,309 m.*, por lo que se puede tomar como mejor resultado el anterior. Sin embargo, bajo criterios operativos dados, se podría elegir el obtenido en esta sección.

Se muestra a continuación el grafo que corresponde a esta solución (Fig. 23).

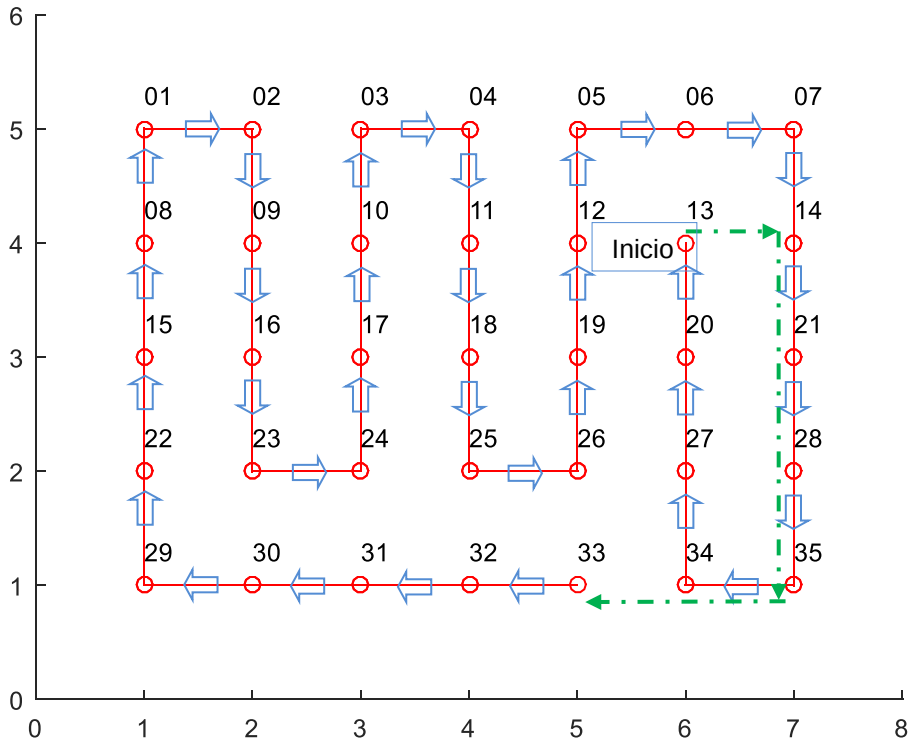


Fig. 23. Grafo de la solución TSP ruta 16; con arista auxiliar **13 – 33**.

4.3. Discusión de los resultados obtenidos.

En esta sección se abordan aspectos observados durante la ejecución del programa con instancias de redes de mayores proporciones a las estudiadas en las secciones anteriores.

Este tipo de redes, que corresponde al esquema normal de cruce de calles en el que el número de conexiones hacia otros nodos es muy bajo (entre uno y tres, generalmente dos), producen matrices dispersas, por lo cual podemos llamarles redes dispersas.

Es importante notar que la numeración de nodos en la red determina la ubicación de los elementos correspondientes dentro de la matriz de adyacencias, lo que influye a su vez en el orden genético (o secuencia de nodos) que se crea durante la generación de los cromosomas o individuos. Esta observación es general**,** es decir, aplica para cualquier manera de agregar los nodos.

Cabe resaltar una característica elemental de las matrices de adyacencia. Es necesario evitar que queden filas (representan llegadas) o columnas (representan salidas) en blanco. En otras palabras, se puede decir que la matriz de adyacencias debe ser consistente. Entendiéndose por consistente que todos los nodos deben tener adyacencia, ya que lo contrario imposibilita el armado de recorridos.

4.3.1. Problemas detectados.

- El programa muestra dificultad en generar la población inicial.
- La ejecución del programa tarda excesivamente con redes de mayores proporciones.

En realidad, los anteriores son dos aspectos del mismo problema. Es útil diferenciarlos, porque más adelante se verá que el problema real es la creación de la población inicial, el cual impacta en el segundo aspecto.

Para dar respuesta, primeramente se describe el procedimiento para la generación de la población inicial y posteriormente se realiza el análisis/discusión del mismo.

En la sección 3.3.3 se describieron los elementos generales que se tomaron en cuenta para la codificación del programa AG en MATLAB. La manera detallada en la que el programa desarrolla el paso 2 en el que se generan los cromosomas (individuos) de la población inicial es:

1. Inicio en un nodo tomado al azar (origen).
2. Selección al azar del siguiente nodo de entre los disponibles adyacentes al anterior, los cuales se encuentran en la misma fila de la matriz de datos.
    2.1. Este nodo debe cumplir la condición de no haber sido utilizado previamente en el presente recorrido (o sea no debe ser repetido).
    2.2. Si el nodo seleccionado al azar ya fue utilizado, entonces el proceso se reinicia desde el paso uno.
3. Repetición del paso dos hasta terminar un recorrido válido, es decir hasta agotar todos los nodos de la red.
4. Conclusión del recorrido agregando el nodo inicial. Este nodo será el único que se repita ya que es necesario para cerrar el ciclo.
    4.1. Es posible que el último nodo no tenga adyacencia con el inicial por lo cual se cree una arista auxiliar que sume un valor *9999* en la función objetivo (distancia total de recorrido).

Derivado de la descripción anterior se puede observar que la construcción del recorrido durante la creación de la población inicial comienza con un nodo cualquiera tomado al azar (origen). Posteriormente se agrega en la posición más a la derecha un nodo no redundante, también tomado al azar, de entre todos los nodos disponibles adyacentes. Si este nodo es redundante (repetido) el proceso vuelve a iniciarse. Si no es redundante, el proceso continúa, buscando otro adyacente al nuevo nodo agregado. Sin embargo, frecuentemente se presenta el

problema de encasillamiento en el que al llegar a un punto dado no existe otro nodo adyacente disponible (no redundante).

Sobre este tema hay que considerar que en este tipo de redes dispersas, después de haber armado parte del recorrido, es muy fácil caer en encasillamientos que impiden continuar su construcción al agotarse las salidas disponibles en un punto dado, por lo que el proceso vuelve a iniciarse antes de terminar, impactando en el tiempo de proceso.

Es decir, conforme se va avanzando en la construcción del recorrido es más factible que se caiga en un encasillamiento ya que se empiezan a tratar de incorporar nodos que ya fueron tomados como parte del recorrido.

De esta manera se puede inferir que mientras más nodos tenga una red, es más probable que al avanzar en la integración del recorrido válido, se caiga en un encasillamiento.

Por otra parte, el número de caminos diferentes que puede darse para el grafo de la red de 35 nodos de la ruta 16 presentada en la sección 4.2 (fig. 24), es función del número de salidas de cada nodo, mismo que está dado por la ecuación 4:

$$N° \, caminos = 1 \times 2 \times 1 \times 2 \times 1 \times 1 \times 1 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 1 \times 1 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 1 \times 1 \times 1 \times 2 \times 1 \times 2 \times 2 \times 1 = 2^{23} = 8'388,608 \quad (4)$$

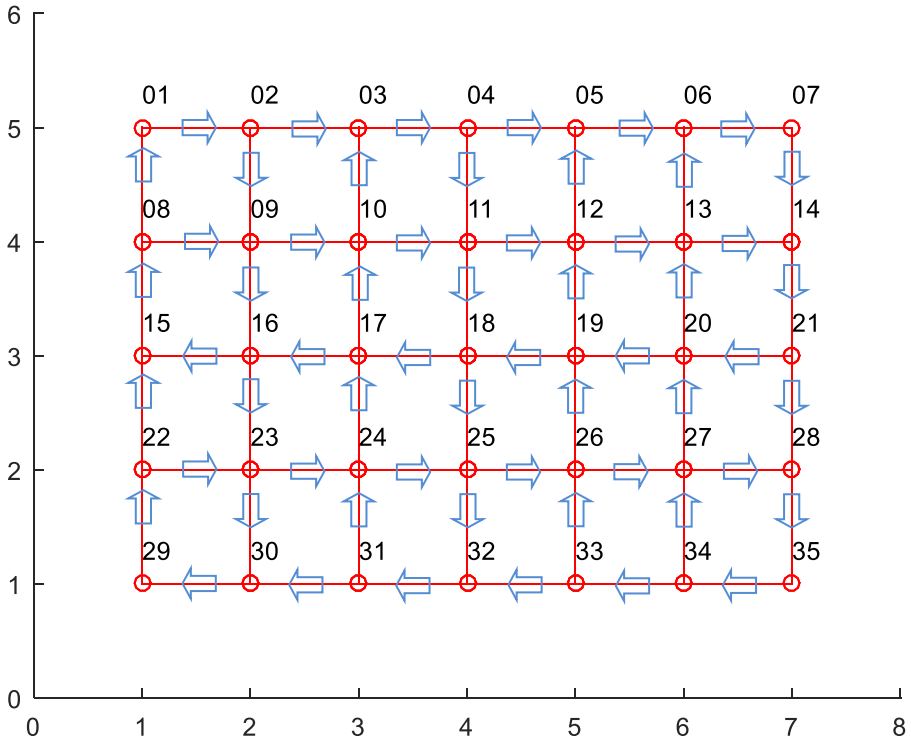


Fig. 24. Grafo de la ruta 16, 35 nodos.

Para una red de 49 nodos de esta misma ruta 16, el número de posibles caminos diferentes es $2^{35} = 34{,}359'738{,}368$ (ya que existen 35 nodos con bifurcación y catorce con una sola salida).

Por lo explicado anteriormente, es paradójico que a pesar de existir un número muy grande de posibles caminos, al programa se le dificulta el armar un recorrido válido inicial, invirtiendo el tiempo tratando de agregar nodos que estén disponibles.

De acuerdo con la revisión del profiler (reporte de análisis del rendimiento) del programa AG en MATLAB, típicamente para la red de 35 nodos, el programa dedica alrededor del 99% del tiempo total de ejecución para la generación de la población inicial y sólo 1% para la optimización.

Por todo lo anterior, se puede observar que la manera de armar los recorridos con esta rutina, agregando nodos elegidos al azar de entre los disponibles, no es tan adecuada para nuestro tipo de redes.

Estas consideraciones son coincidentes con las reiteradas menciones en la literatura sobre el tema de la generación de la **población inicial**, reconociéndose que es un tópico bastante importante.

4.3.2. Alternativas.

Las alternativas lógicas fueron modificar la manera de intentar agregar los nodos en la construcción del recorrido inicial.

Inicialmente se probó con un cambio en la rutina original de generación de la población inicial, introduciendo un procedimiento de reiteración de la selección aleatoria (ver paso 2.2), de tal manera que se maximizaran las posibilidades de integración del recorrido completo, antes de reiniciar en el paso 1. La subrutina así modificada queda de la siguiente forma:

1. Inicio en un nodo tomado al azar (origen).
2. Selección al azar del siguiente nodo de entre los disponibles adyacentes al anterior, los cuales se encuentran en la misma fila de la matriz de datos.
    2.1. Este nodo debe cumplir la condición de no haber sido utilizado previamente en el presente recorrido (o sea no debe ser repetido).
    2.2. Si el nodo elegido ya fue utilizado, entonces se repite el mismo paso de selección al azar de entre los disponibles adyacentes, hasta localizar uno que no haya sido ya empleado o agotar los existentes en la misma fila de la matriz de adyacencias. Si ya no hay nodos elegibles, se reinicia en el paso uno.
3. Repetición del paso dos hasta terminar el recorrido, es decir hasta agotar todos los nodos de la red.

4. Conclusión del recorrido agregando el nodo inicial. Este nodo será el único que se repita ya que es necesario para cerrar el ciclo.
    4.1. Es posible que el último nodo no tenga adyacencia con el inicial por lo cual se cree una arista auxiliar que sume un valor *9999* en la función objetivo (distancia total de recorrido).

Esta alternativa tampoco generó exitosamente los individuos de la población inicial, pues por las razones expuestas se presentó el problema de encasillamiento en el que en un punto dado no existen nodos adyacentes disponibles para continuar el recorrido.

Otra opción fue implementar la siguiente subrutina nombrada GenIndi2, que sólo guarda diferencias en el paso 2 con respecto a la original:

1. Inicio en un nodo tomado al azar (origen).
2. Selección del primer nodo disponible adyacente al anterior, con un valor diferente a 9999, el cual se encuentra en la posición más cercana, dentro de la misma fila de la matriz de datos.
    2.1. Este nodo debe cumplir la condición de no haber sido utilizado previamente en el presente recorrido (o sea no debe ser repetido).
    2.2. Si el nodo elegido ya fue utilizado, entonces se toma el siguiente nodo disponible que sea adyacente al nodo anterior, es decir el que sigue en la misma fila de la matriz de adyacencias. Si ya no hay nodos elegibles, se reinicia en el paso uno.
3. Repetición del paso dos hasta terminar el recorrido, es decir hasta agotar todos los nodos de la red.
4. Conclusión del recorrido agregando el nodo inicial. Este nodo será el único que se repita ya que es necesario para cerrar el ciclo.

4.1. Es posible que el último nodo no tenga adyacencia con el inicial por lo cual se cree una arista auxiliar que sume un valor 9999 en la función objetivo (distancia total de recorrido).

Dado que generalmente los nodos de una red se numeran en forma consecutiva, la subrutina GenIndi2, al buscar agregar el nodo más cercano en la misma fila de la matriz de datos, tiende a generar trayectorias en tramos de líneas rectas de varias aristas.

En realidad, la subrutina GenIndi2, al agregar los nodos de una manera determinística, crea recorridos parcialmente iguales. Esto se puede apreciar fácilmente con unos ejemplos a continuación, recordando que el algoritmo de la subrutina va integrando el recorrido con los nodos subsecuentes disponibles más cercanos en la misma fila de la matriz de adyacencias.

Utilicemos para el efecto, nuevamente el grafo de 35 nodos presentado en la sección 4.2 (figura 25) y la matriz de adyacencias respectiva (tabla 63).

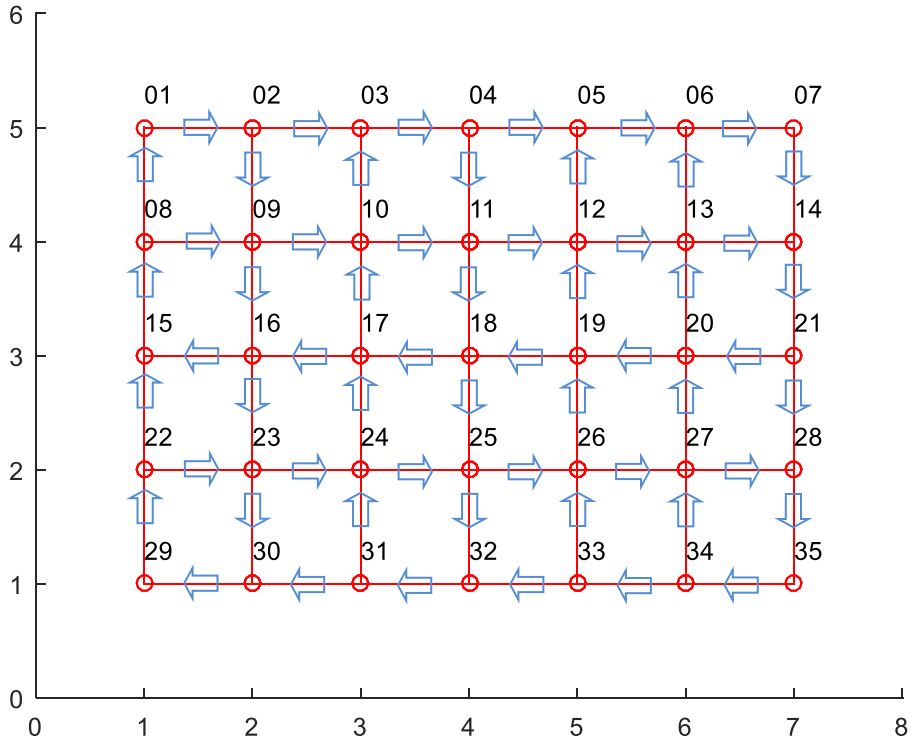


Fig. 25. Grafo de la ruta 16, 35 nodos.

Veamos, por ejemplo, el recorrido iniciado en el nodo 1. La secuencia correspondiente producida con esta subrutina sería:

1, 2, 3, 4, 5, 6, 7, 14, 21, 20, 13.

Después de integrar el nodo 13, el algoritmo de la subrutina ya no puede continuar, dado que las dos únicas salidas posibles 6 y 14, ya están cubiertas, es decir en este caso esos nodos ya fueron empleados, por lo tanto se produce un encasillamiento.

Tabla 63. Matriz de adyacencias ponderada. Ruta 16, 35 nodos. *

1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1	2	3	4	5
999	**82**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	**95**	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	**72**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	**93**	999	999	999	999	999	**68**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	**100**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	**119**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
72	999	999	999	999	999	999	999	**74**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	**95**	999	999	999	999	999	**71**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	**71**	999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	**95**	999	999	999	999	999	**72**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	**72**	999	999	999	999	999	999	999	**98**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	**68**	999	999	999	999	999	999	999	**126**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	**69**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	**69**	999	999	999	999	999	**90**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	**68**	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	**66**	999	999	999	999	**95**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	**73**	999	999	999	999	**98**	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**135**	999	999	999	999	999	999	999	**72**	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	**70**	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**88**	999	999	999	999	999	**107**	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	**69**	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**95**	999	999	999	999	**74**	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**73**	999	999	999	999	999	999	999	**98**	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**71**	999	999	999	999	999	999	999	**157**	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**73**	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**157**	999	999	999	999	999	999	999	999	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**85**	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**70**	999	999	999	999	**91**	999	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**68**	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**66**	999	999	999	999	**96**	999	999	999	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**68**	999	999	999	999	**98**	999	999
999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	999	**176**	999

* Para mejorar la visibilidad en esta tabla, se sustituyeron los valores *9999* por *999.*

Si probamos otro recorrido, iniciando aproximadamente a la mitad, digamos en el nodo 18, la secuencia sería:

18, 17, 10, 3, 4, 5, 6, 7, 14, 21, 20, 13.

Como se puede apreciar, a partir del cuarto nodo la secuencia es la misma que la anterior, terminando en el encasillamiento del nodo 13 por la razón mencionada.

Cabe mencionar que, en una ejecución normal, la subrutina prueba armar recorridos iniciando prácticamente en todos los nodos, siendo la única diferencia que unos recorridos son de mayor longitud que otros, de tal manera que para esta red, la secuencia máxima que se produce es:

30, 29, 22, 15, 8, 1, 2, 3, 4, 5, 6, 7, 14, 21, 20, 13.

Como se puede ver, nuevamente se llega al sub-recorrido que va a culminar en el encasillamiento del nodo 13.

Se puede comprobar fácilmente que esta secuencia de 16 nodos es la máxima que se puede generar con esta subrutina, dada la topología de esta red.

4.3.3. Consideraciones sobre el caso TSP clásico.

Para asegurar el correcto funcionamiento de la metaheurística genética dentro de su implementación en el programa que se desarrolló en MATLAB, se efectuó una serie de corridas con diferentes instancias. Estas instancias corresponden al TSP clásico. En el TSP clásico todos los nodos están conectados entre sí, lo que implica que la matriz de adyacencias sea densa. Existe la variante de que la conectividad entre dos nodos sea la misma en ambos sentidos, es decir que la arista incidente tenga la misma longitud, lo que resulta en que la matriz sea simétrica.

Primeramente se muestra la aplicación del programa AG a instancias correspondientes a sendos ejemplos del TSP de un par de textos sobre Investigación de Operaciones. Estos textos se han revisado durante el desarrollo de esta investigación. Ellos son los debidos a Taha (2009) y Maroto et al. (2002). Dado que la finalidad de estos ejemplos es mostrar métodos presentados por los autores, los ejercicios son de pequeñas proporciones.

A continuación se presenta la aplicación del programa a una matriz de mayor tamaño, correspondiente a un ejercicio del TSP, creada específicamente como instancia de prueba para este propósito.

4.3.3.1. Aplicación del programa AG al ejercicio TSP del libro de Taha (2012).

Se aplicó el programa AG al ejemplo que el autor de los elementos que se siguieron para codificar el programa MATLAB, presenta en su libro para ejemplificar su algoritmo genético para resolver el TSP (Taha, 2012).

La matriz de adyacencias que representa las distancias entre los cinco nodos del problema es una matriz densa y simétrica, es decir existe conexión entre todos los nodos y además su valor es el mismo en ambos sentidos para todos los pares de nodos. Esta matriz de datos se muestra en la tabla 64:

Tabla 64. Matriz de adyacencias ponderada. Ejemplo TSP (Taha, 2012).

9999	120	220	150	210
120	9999	100	110	130
220	100	9999	160	185
150	110	160	9999	190
210	130	185	190	9999

La población inicial que creó el programa (correspondiente a la 1ª generación) se presenta en la tabla 65:

Tabla 65. Población inicial (1ª generación).

Generación 1							
Individuo							Distancia
1	5	3	4	1	2	5	745
2	2	3	4	1	5	2	820
3	3	2	1	4	5	3	725
4	4	5	2	3	1	4	790
5	2	3	4	5	1	2	780
6	3	2	1	5	4	3	790

En este ejercicio el programa no contempló inicialmente ninguna arista auxiliar para cerrar ciclos ya que al tratarse de un TSP clásico implica una red con conexiones entre todos los nodos.

Por cuestiones de espacio se omiten los detalles de las diferentes generaciones, pero se hace hincapié en que el programa realizó el trabajo de optimización en cuatro iteraciones convergiendo desde la 5ª generación. En seguida se indica el resultado final (tabla 66):

Tabla 66. Resultado de ejecución del programa AG (10ª generación).

Generacion 10							
Individuo							Distancia
1	3	2	1	4	5	3	725
2	3	2	1	4	5	3	725
3	3	2	1	4	5	3	725
4	3	2	1	4	5	3	725
5	3	2	1	4	5	3	725
6	3	2	1	4	5	3	725

Cabe aclarar que para obtener este resultado el programa se ejecutó varias ocasiones (cuatro para ser precisos). De esta manera se replicó el recorrido obtenido en el libro, con una longitud de 725 el cual es óptimo, verificándose el adecuado funcionamiento del programa.

4.3.3.2. Aplicación del programa AG al ejercicio TSP del libro de Maroto et al. (2002).

De manera análoga se aplicó el programa AG al ejemplo TSP de cinco ciudades de Maroto et al. (2002), visto con anterioridad en esta misma tesis y que fue resuelto con PLE y heurísticas del paquete QSB.

La matriz de adyacencias que representa las distancias entre los nodos es una matriz densa y se muestra en la tabla 67. En este caso se trata de una matriz no simétrica.

Tabla 67. Matriz de adyacencias ponderada. Ejemplo TSP (Maroto et al., 2012).

9999	552.0	661.4	424.6	178.3
594.4	9999	629.0	628.7	363.2
662.2	628.3	9999	247.2	598.0
424.1	628.2	246.7	9999	360.0
178.1	364.7	595.5	358.7	9999

La población inicial que creó el programa (correspondiente a la 1ª generación) se presenta en la tabla 68:

Tabla 68. Población inicial (1ª generación).

Generacion 1							
Individuo							Distancia
1	5	3	4	2	1	5	2,244.6
2	4	1	5	3	2	4	2,458.5
3	1	4	3	2	5	1	2,673.7
4	1	5	4	3	2	1	2,008.4
5	3	4	5	2	1	3	2,227.7
6	3	4	1	2	5	3	2,182.0

En este ejercicio el programa no contempló inicialmente ninguna arista auxiliar para cerrar ciclos ya que se trata de un TSP clásico con conexiones entre todos los nodos.

Se omiten los detalles de las diferentes generaciones, pero se aclara que el programa realizó el trabajo de optimización en cinco iteraciones convergiendo desde la 6ª generación. En seguida se indica el resultado final (tabla 69):

Tabla 69. Resultado de ejecución del programa AG (10ª generación).

Generacion 10							
Individuo							Distancia
1	1	4	3	2	5	1	1,842.9
2	1	4	3	2	5	1	1,842.9
3	1	4	3	2	5	1	1,842.9
4	1	4	3	2	5	1	1,842.9
5	1	4	3	2	5	1	1,842.9
6	1	4	3	2	5	1	1,842.9

En este caso, para obtener este resultado el programa se ejecutó cinco ocasiones. De esta manera se replicó el recorrido obtenido en el libro, con una longitud de $1,842.9\ km$, por lo que se puede comprobar que el programa funcionó correctamente.

4.3.3.3. Aplicación del programa AG a una matriz de 21 nodos generada aleatoriamente.

Siguiendo esta línea de aplicaciones a casos TSP clásicos se creó una matriz densa de mayor tamaño para aplicar el programa AG.

La matriz se generó exprofeso en una hoja de cálculo con valores entre 0 y 100, utilizando la función random.

Se realizaron varias pruebas del programa correspondientes a otros tantos tamaños de matrices. Se presenta en esta sección la corrida correspondiente a una red de 21 puntos y a continuación se comentan aspectos relacionados con los tiempos de ejecución. Se muestra la matriz respectiva en la tabla 70:

Tabla 70. Matriz de adyacencias ponderada creada aleatoriamente.*

999	29	65	43	17	51	86	27	74	98	43	58	25	46	85	35	52	19	23	84	82
14	999	49	25	51	77	25	50	86	46	52	91	96	73	67	24	28	48	20	18	45
24	85	999	16	81	25	86	91	56	36	18	59	36	50	39	51	100	41	94	69	33
51	71	66	999	45	87	99	19	34	50	83	90	71	21	65	75	71	78	84	76	25
27	53	83	18	999	66	82	25	29	20	86	56	87	17	54	37	59	66	54	91	46
55	96	100	42	20	999	60	39	55	20	76	39	87	63	87	93	62	20	27	65	55
70	29	18	64	37	52	999	42	73	18	66	24	96	25	64	41	29	75	25	43	73
21	77	49	27	45	85	92	999	93	26	45	72	17	38	73	87	47	38	26	97	75
73	55	69	15	93	17	89	75	999	98	87	62	88	96	63	71	50	42	71	21	82
59	37	46	13	16	58	88	48	48	999	46	89	68	100	60	28	86	51	45	39	79
27	82	87	58	26	93	12	81	56	97	999	95	56	19	100	13	67	54	41	47	45
26	69	22	93	90	45	63	15	38	70	70	999	86	45	11	88	54	95	60	45	16
25	90	90	99	21	38	35	82	45	50	35	94	999	38	51	34	87	85	43	97	80
11	23	87	82	61	54	15	82	39	65	76	68	86	999	41	44	26	13	23	50	73
28	46	27	95	28	33	76	41	18	75	81	14	26	44	999	58	69	10	55	91	62
46	74	49	23	65	14	28	61	62	86	95	75	49	40	69	999	45	56	93	82	20
58	76	41	15	46	16	95	39	99	58	18	77	53	84	82	83	999	91	90	65	69
55	50	59	22	24	29	73	42	15	61	72	58	49	30	52	90	31	999	60	95	61
74	22	83	45	31	93	40	97	32	22	71	25	84	87	19	84	45	64	999	70	61
36	57	53	63	94	34	65	86	80	29	12	70	88	46	52	48	27	37	94	999	98
94	70	23	19	78	34	72	11	33	47	63	57	67	58	85	32	61	47	47	84	999

* Por cuestiones de espacio se muestran en esta matriz valores 999 en lugar de 9999.

En esta ejecución el parámetro correspondiente al número de generaciones se fijó en 20.

En la tabla 71, se presentan los resultados de la 1ª generación de la ejecución del programa, los cuales corresponden a la población inicial. Como se puede observar, los valores correspondientes a la función de adaptación de los seis individuos generados inicialmente oscilan entre $1{,}056$ y $1{,}335$. Sabemos que teóricamente el valor promedio de la distancia de las aristas generadas con la función random vale 50 por lo que la distancia total del recorrido promedio ascendería a $1{,}050$.

Tabla 71. Población inicial (1ª generación).

Generación 1																						
Individuo																						distancia
1	9	8	20	21	1	17	4	10	11	18	7	19	3	15	12	14	13	2	16	5	6	9	1,246
2	15	8	2	16	7	3	17	21	13	9	14	1	11	6	5	20	18	19	10	4	12	15	1,056
3	21	15	12	7	5	19	20	17	18	3	4	11	16	8	13	10	9	1	14	2	6	21	1,062
4	2	14	7	3	18	20	8	11	9	6	13	4	12	10	1	15	21	5	19	16	17	2	1,335
5	16	1	7	18	19	17	4	3	20	2	6	13	10	8	15	12	5	14	21	9	11	16	1,181
6	17	14	21	9	20	3	18	4	11	13	10	15	5	1	16	7	19	8	12	6	2	17	1,057

Para evitar el uso excesivo de espacio se omiten las iteraciones intermedias, pero se muestra la correspondiente a la 10ª generación en la tabla 72.

Tabla 72. Resultado de ejecución del programa AG. 10ª generación.

Generación **1 0**																						
Individuo																						distancia
1	9	20	2	16	7	3	18	14	1	11	6	5	21	19	10	4	12	15	8	17	13	9	801
2	9	20	2	16	7	3	18	4	14	1	11	6	5	19	10	12	15	8	21	17	13	9	850
3	20	2	16	7	3	18	5	13	14	1	11	6	21	19	10	4	12	15	8	17	9	20	910
4	9	20	17	7	3	18	13	14	1	11	6	5	19	10	4	12	15	8	2	16	21	9	841
5	9	20	2	16	7	3	18	4	14	1	11	6	5	19	10	12	15	8	21	17	13	9	850
6	21	9	20	17	7	3	18	13	14	1	11	6	5	19	10	4	12	15	8	2	16	21	841

Como se puede esperar el algoritmo genético mejora (minimiza) los individuos en las iteraciones sucesivas. El intervalo en el que se encuentran los valores de la función objetivo (distancia) de los diferentes individuos ha pasado de *[1,056, 1,035]* en la población inicial, a *[801, 910]* en esta 10ª generación por lo que se ha reducido significativamente.

La continuación de la optimización realizada por el AG arroja el siguiente resultado en la 20ª generación (tabla 73):

Tabla 73. Resultado de ejecución del programa AG. 20ª generación.

Generación	**2**	**0**																				
Individuo																						distancia
1	9	20	2	16	7	3	18	14	1	11	6	5	21	19	10	4	12	15	8	17	13	9	801
2	20	2	16	7	21	3	18	14	1	11	6	5	19	10	4	12	15	8	17	13	9	20	840
3	20	2	16	7	21	3	18	14	1	11	6	5	19	10	4	12	15	8	17	13	9	20	840
4	21	9	20	17	7	3	18	14	1	11	6	5	19	10	4	12	15	8	13	2	16	21	814
5	21	9	20	17	7	3	18	14	1	11	6	5	19	10	4	12	15	8	13	2	16	21	814
6	9	20	2	16	7	3	18	14	1	11	6	5	21	19	10	4	12	15	8	17	13	9	801

En esta ejecución el programa AG no alcanza a converger a un mismo individuo en 20 generaciones, sin embargo se observa un intervalo de valores todavía menor al pasar de $[801, 910]$ correspondiente a la 10ª iteración, a $[801, 840]$ en esta 20ª generación. Se infiere que el valor al que convergerá el algoritmo será el menor de estos, 801.

Nuevamente con este ejercicio se puede afirmar que el programa funcionó correctamente.

En estos ejercicios basados en la matriz generada aleatoriamente para el TSP se observaron los tiempos de ejecución. En la tabla 74 se presentan los tiempos de ejecución obtenidos para algunos tamaños de esta red.

Tabla 74. Tiempos de ejecución del programa AG para la matriz de datos generada aleatoriamente.

N° nodos	Segundos	Minutos
15	8.88	0.15
19	308.36	5.14
20	1,579.41	26.32
21	6,444.72	107.41

El tiempo de ejecución crece exponencialmente en función del número de nodos, de tal manera que se puede considerar que el aumento de un nodo, a 22, implicaría un tiempo de ejecución de varias horas y un nodo más, 23, requeriría del orden de un día. Al respecto vale la pena reiterar que el TSP se considera un problema NP-hard.

Para tener una idea más clara del número de recorridos diferentes que se podría dar con las redes que implican el TSP se presenta una tabla de $n\text{-}1!$ hasta $n = 25$ (tabla 75).

Tabla 75. Número de recorridos para redes TSP**.**

N° nodos **(*n*)**	N° de recorridos **(*n-1!*)**
2	1
3	2
4	6
5	24
6	120
7	720
8	5,040
9	40,320
10	362,880
11	3,628,800
12	39,916,800
13	479,001,600
14	6,227,020,800
15	87,178,291,200
16	1,307,674,368,000
17	20,922,789,888,000
18	355,687,428,096,000
19	6,402,373,705,728,000
20	121,645,100,408,832,000
21	**2,432,902,008,176,640,000**
22	51,090,942,171,709,400,000
23	1,124,000,727,777,610,000,000
24	25,852,016,738,885,000,000,000
25	620,448,401,733,239,000,000,000

Se analizó el profiler de MATLAB de estas ejecuciones del programa con matrices densas, se observó que, como en el caso de las redes dispersas, casi todo el tiempo se consume en la integración de los cromosomas de la población inicial.

Con estos resultados se puede afirmar que el algoritmo genético y el programa funcionan correctamente cuando se aplica a casos TSP clásicos, tanto en la integración de los recorridos correspondientes a los individuos de la población inicial como en la etapa pura de optimización debida al Algoritmo Genético.

4.3.4. Diseño y análisis de experimento para reducir el tiempo de ejecución del programa AG.

Para analizar el efecto de los factores de entrada: 1) Tamaño de la población, 2) Número de generaciones y 3) Tasa de probabilidad de mutación, del programa AG, con el fin de determinar los valores que permitan reducir su tiempo de ejecución, se llevó a cabo un experimento, mismo que se detalla a continuación.

Diseño de experimento.

Se definió un diseño factorial 2^3 conformado por los siguientes factores con sus respectivos niveles.

Primer corrida del experimento.

Factores y niveles de diseño:

1). Tamaño de población.

Nivel bajo: 6 individuos (se usó para el resultado presentado en esta tesis).

Nivel alto: 20 individuos.

2). Número de generaciones.

Nivel bajo: 10 (se usó para el resultado presentado en esta tesis).

Nivel alto: 30.

3). Tasa de mutación.

Nivel bajo: 0.10 (se usó para el resultado presentado en esta tesis).

Nivel alto: 0.20 (se usó para el resultado presentado en esta tesis).

Se ejecutó el programa AG con la red correspondiente a 35 nodos de la ruta 16, con lo que se obtuvieron los siguientes datos para el análisis del experimento (tabla 76):

Tabla. 76. Datos para el experimento.

	Población (individuos)	Generaciones (número)	Tasa de mutación (porcentaje)	Corrida Inicial (segundos)	Réplica (segundos)	NotaciónYates
1	6	10	0.1	93.37	173.57	(1)
2	20	10	0.1	309.06	289.81	a
3	6	30	0.1	220.19	189.62	b
4	20	30	0.1	482.9	383.8	ab
5	6	10	0.2	347.66	161.73	c
6	20	10	0.2	363.66	491.85	ac
7	6	30	0.2	155.15	229.25	bc
8	20	30	0.2	481.89	450.27	abc

Se empleó el paquete Statgraphics para ejecutar el análisis estadístico con los datos de las corridas del programa AG obteniendo los siguientes resultados:

Atributos del diseño de experimento.

Multilevel Factorial Design Attributes
Design class: Multilevel Factorial
File name: DesignFile4DOE.sfx

Base Design
Number of experimental factors: 3
Number of blocks: 2
Number of responses: 1
Number of runs: 16
Error degrees of freedom: 8
Randomized: No

Factors	*Low*	*High*	*Levels*	*Units*
A: Población	6.0	20.0	2	Individuos
B: Generaciones	10.0	30.0	2	Número
C: Tasa Mutación	0.1	0.2	2	

Responses	*Units*
Var_1: Tiempo CPU	Segundos

The StatAdvisor

You have created a multilevel factorial design consisting of 16 runs. The design is to be run in 2 blocks. The order of the experiments has not been randomized. If lurking variables are present, they may distort the results.

Se presenta a continuación el bloque de datos generado en el paquete, producto de la ejecución del programa AG, el cual correspondió a ocho corridas originales más una réplica:

Tabla. 77. Bloque de datos para el experimento.

BLOCK	A: Población (Individuos	B: Generaciones (Número)	C: Tasa Mutación	Var_1: TiempoCPU (Segundos)
1	6	10	0.1	93.37
1	20	10	0.1	309.06
1	6	30	0.1	220.19
1	20	30	0.1	482.9
1	6	10	0.2	347.66
1	20	10	0.2	363.66
1	6	30	0.2	155.15
1	20	30	0.2	481.89
2	6	10	0.1	173.57
2	20	10	0.1	289.81
2	6	30	0.1	189.62
2	20	30	0.1	383.8
2	6	10	0.2	161.73
2	20	10	0.2	491.85
2	6	30	0.2	229.25
2	20	30	0.2	450.27

Análisis de variancia.

De acuerdo con el Anova proporcionado por Statgraphics sólo el factor **Población** tiene significancia estadística, al ser el Valor-P menor a 0.05 (tabla 78).

Tabla 78. Análisis de variancia del experimento.

Analysis of Variance for Var_1: Tiempo CPU

Source	*Sum of Squares*	*Df*	*Mean Square*	*F-Ratio*	*P-Value*
A:A: Población	176967.	1	176967.	37.80	0.0003
B:B: Generaciones	8206.55	1	8206.55	1.75	0.2221
C:C: Tasa Mutación	18167.0	1	18167.0	3.88	0.0844
AB	6666.72	1	6666.72	1.42	0.2669
AC	689.85	1	689.85	0.15	0.7111
BC	13169.9	1	13169.9	2.81	0.1320
blocks	440.79	1	440.79	0.09	0.7668
Total error	37452.3	8	4681.54		
Total (corr.)	261761.	15			

R-squared = 85.6921 percent
R-squared (adjusted for d.f.) = 76.1536 percent
Standard Error of Est. = 68.4218
Mean absolute error = 39.2998
Durbin-Watson statistic = 1.84416 (P=0.2740)
Lag 1 residual autocorrelation = 0.0439127

The StatAdvisor
The ANOVA table partitions the variability in Var_1: Tiempo CPU into separate pieces for each of the effects. It then tests the statistical significance of each effect by comparing the mean square against an estimate of the experimental error. In this case, 1 effects have P-values less than 0.05, indicating that they are significantly different from zero at the 95.0% confidence level.

The R-Squared statistic indicates that the model as fitted explains 85.6921% of the variability in Var_1: Tiempo CPU. The adjusted R-squared statistic, which is more suitable for comparing models with different numbers of independent variables, is 76.1536%. The standard error of the estimate shows the standard deviation of the residuals to be 68.4218. The mean absolute error (MAE) of 39.2998 is the average value of the residuals. The Durbin-Watson (DW) statistic tests the residuals to determine if there is any significant correlation based on the order in which they occur in your data file. Since the P-value is greater than 5.0%, there is no indication of serial autocorrelation in the residuals at the 5.0% significance level.
Pob

En concordancia con el Anova, la gráfica de Pareto muestra que el único factor que está causando efecto en la respuesta estudiada (Tiempo CPU) es el tamaño de la Población (ver figura 26).

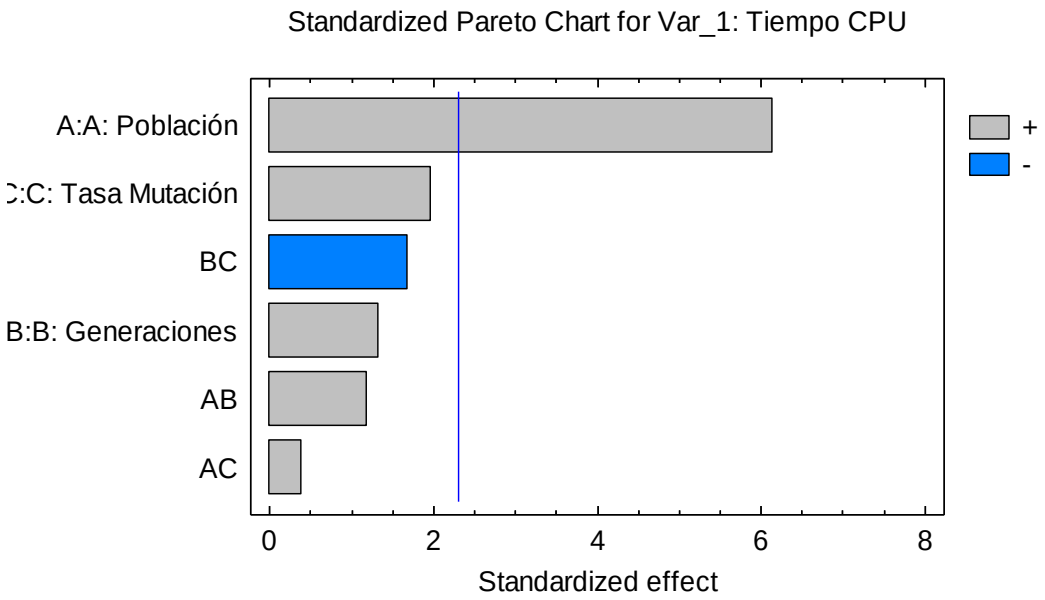


Fig. 26. Gráfica de Pareto del experimento con tres factores.

Asimismo la gráfica de efectos principales reitera al factor Población con efecto en el Tiempo de CPU.

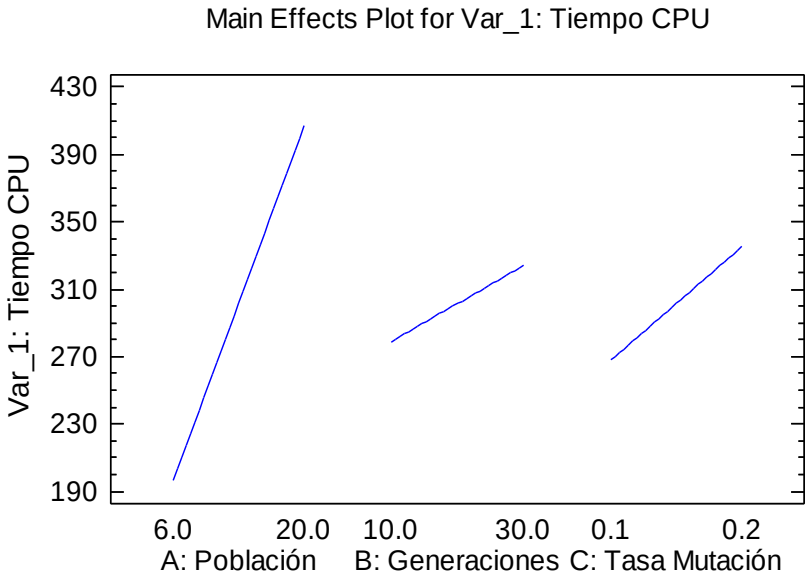


Fig. 27. Gráfica de efectos principales de los factores.

Interacción entre factores.

Por lo que toca al efecto de interacción entre factores, de acuerdo con la gráfica de interacciones (ver figura 28), no se presenta interacción entre factores al presentarse un cierto paralelismo de las líneas en dos de las interacciones, y más aún, no existir un cruce de líneas en ninguna de las tres.

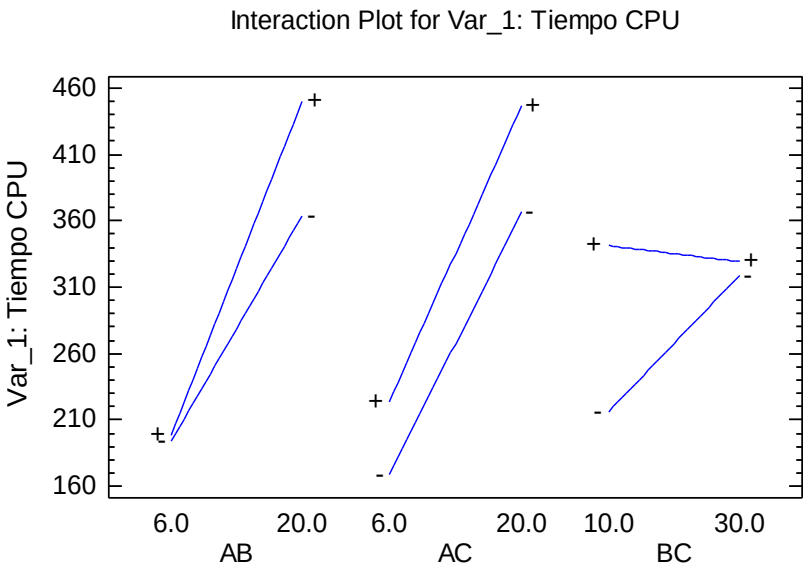


Fig. 28. Gráfica de interacciones de los factores.

Optimización de la respuesta.

Los valores de los factores de diseño que optimizan (minimizan) la respuesta (Tiempo de CPU) son: **Población *6* individuos, número de Generaciones *10* y Tasa de mutación *0.1***, los cuales corresponden a los valores bajos de los mismos.

Tabla 79. Optimización de la respuesta.

Optimize Response
Goal: minimize Var_1: Tiempo CPU

Optimum value = 138.263

Factor	*Low*	*High*	***Optimum***
A: Población	6.0	20.0	**6.0**
B: Generaciones	10.0	30.0	**10.0**
C: Tasa Mutación	0.1	0.2	**0.1**

The StatAdvisor
This table shows the combination of factor levels which minimizes Var_1: Tiempo CPU over the indicated region. Use the Analysis Options dialog box to indicate the region over which the optimization is to be performed. You may set the value of one or more factors to a constant by setting the low and high limits to that value.

Estos valores de los factores Población y Generaciones son confirmados por medio de la gráfica de contornos para la Tasa de mutación = *0.1* (figura 29), en donde podemos apreciar que los valores bajos de la respuesta (Tiempo de CPU) se dan para valores bajos de la Población, aun cuando se maneje variación en el factor Generaciones hacia valores altos, como puede observarse en la primer línea de contorno de la izquierda que corresponde a los valores bajos de la respuesta.

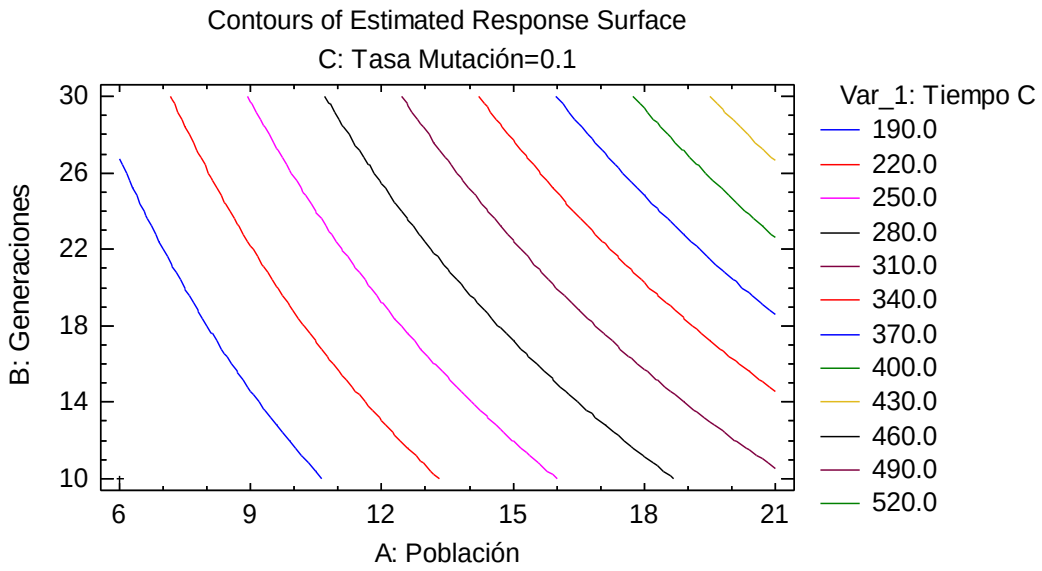


Fig. 29. Gráfica de contornos de la respuesta.

Más claramente la gráfica de la superficie de respuesta muestra que los valores bajos de la Población y de las Generaciones ocasionan el menor valor en el Tiempo de CPU para la Tasa de mutación = *0.1* (figura 30).

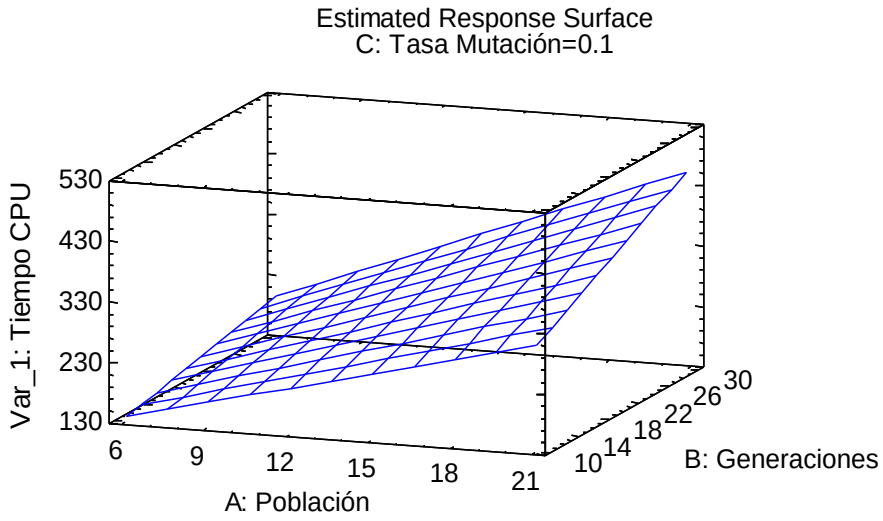


Fig. 30. Gráfica de la superficie de respuesta.

Supuesto de normalidad.

En cuanto al cumplimiento de supuestos del experimento, por lo que toca al de normalidad se puede afirmar que este se está cumpliendo en, virtud del alineamiento de los residuos con respecto a una línea recta (figura 31).

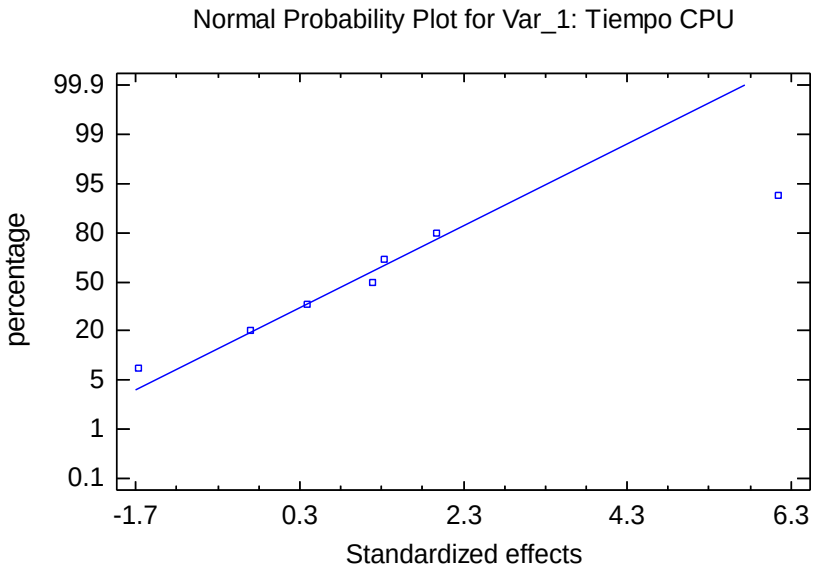


Fig. 31. Gráfica de cumplimiento del supuesto de normalidad.

Supuesto de varianza constante.

Asimismo la figura 32, que corresponde a la representación del cumplimiento de la varianza constante entre tratamientos se está cumpliendo, al no existir un patrón definido en la ubicación de los predichos vs. los residuos.

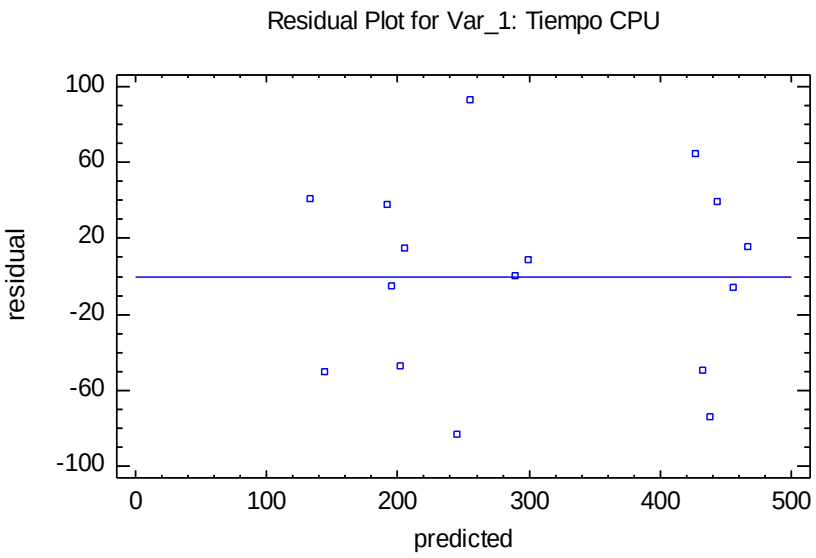


Fig. 32. Gráfica de cumplimiento del supuesto de varianza constante.

Segunda corrida del experimento.

Factores y niveles de diseño:

1). Tamaño de población.

Nivel bajo: $\boldsymbol{10}$ individuos.

Nivel alto: $\boldsymbol{50}$ individuos.

2). Número de generaciones.

Nivel bajo: 10 (se usó para el resultado presentado en esta tesis).

Nivel alto: 30.

3). Tasa de mutación.

Nivel bajo: 0.10 (se usó para el resultado presentado en esta tesis).

Nivel alto: 0.20 (se usó para el resultado presentado en esta tesis).

Se ejecutó el programa AG con la red correspondiente a 35 nodos de la ruta 16.

Se obtuvieron los siguientes datos para el análisis del experimento (tabla 80):

Tabla. 80. Datos para el experimento.

	Población (individuos)	Generaciones (número)	Tasa de mutación (porcentaje)	Corrida Inicial (segundos)	Réplica (segundos)	NotaciónYates
1	10	10	0.1	237.89	202.00	(1)
2	50	10	0.1	1126.40	1033.53	a
3	10	30	0.1	191.55	362.13	b
4	50	30	0.1	1130.65	1154.92	ab
5	10	10	0.2	193.44	140.84	c
6	50	10	0.2	1229.58	1308.62	ac
7	10	30	0.2	248.08	195.92	bc
8	50	30	0.2	944.59	1130.38	abc

Se empleó el paquete Statgraphics para ejecutar el análisis estadístico de los resultados de las corridas del programa AG obteniendo los siguientes resultados:

Atributos del diseño de experimento.

Se creó el siguiente diseño en el paquete Statgraphics.

Multilevel Factorial Design Attributes

Design class: Multilevel Factorial
File name: C:\Users\JOSE LUIS\Documents\A_PROYECTO\Ruta 16\DOE\Población niveles 10 50 individuos\SG PLUS EXPERIMENT 10 50.sfx

Base Design
Number of experimental factors: 3
Number of blocks: 2
Number of responses: 1
Number of runs: 16
Error degrees of freedom: 8
Randomized: No

Factors	*Low*	*High*	*Levels*	*Units*
A **Población**	**10.0**	**50.0**	2	individuos
B Generaciones	10.0	30.0	2	
C Tasa Mutación	0.1	0.2	2	

Responses	*Units*
Var_1 Tiempo CPU	

The StatAdvisor
You have created a multilevel factorial design consisting of 16 runs. The design is to be run in 2 blocks. The order of the experiments has not been randomized. If lurking variables are present, they may distort the results.

Se presenta a continuación el bloque de datos generado en el paquete, producto de la ejecución del programa AG, el cual correspondió a ocho corridas originales más una réplica:

Tabla. 81. Datos para el experimento.

BLOCK	A: Población (Individuos	B: Generaciones (Número)	C: Tasa Mutación	Var_1: TiempoCPU (Segundos)
1	10	10	0.1	237.89
1	50	10	0.1	1126.40
1	10	30	0.1	191.55
1	50	30	0.1	1130.65
1	10	10	0.2	193.44
1	50	10	0.2	1229.58
1	10	30	0.2	248.08
1	50	30	0.2	944.59
2	10	10	0.1	202.00
2	50	10	0.1	1033.53
2	10	30	0.1	362.13
2	50	30	0.1	1154.92
2	10	10	0.2	140.84
2	50	10	0.2	1308.62
2	10	30	0.2	195.92
2	50	30	0.2	1130.38

Análisis de variancia.

De acuerdo con el Anova proporcionado por Statgraphics sólo el factor **Población** tiene significancia estadística, al ser el Valor-P menor a *0.05* (tabla 82).

Tabla 82. Análisis de variancia del experimento.

Analysis of Variance for Var_1 Tiempo CPU

Source	*Sum of Squares*	*Df*	*Mean Square*	*F-Ratio*	*P-Value*
A:A Población	3.31861E6	1	3.31861E6	434.49	**0.0000**
B:B Generaciones	813.39	1	813.39	0.11	0.7525
C:C Tasa Mutación	141.729	1	141.729	0.02	0.8950
AB	19677.1	1	19677.1	2.58	0.1471
AC	9166.15	1	9166.15	1.20	0.3052
BC	21973.6	1	21973.6	2.88	0.1283
blocks	3196.77	1	3196.77	0.42	0.5358
Total error	61103.8	8	7637.97		
Total (corr.)	3.43468E6	15			

R-squared = 98.221 percent
R-squared (adjusted for d.f.) = 97.035 percent
Standard Error of Est. = 87.3955
Mean absolute error = 50.0981
Durbin-Watson statistic = 2.6733 (P=0.7982)
Lag 1 residual autocorrelation = -0.389772

The StatAdvisor
The ANOVA table partitions the variability in Var_1 Tiempo CPU into separate pieces for each of the effects. It then tests the statistical significance of each effect by comparing the mean square against an estimate of the experimental error. In this case, 1 effects have P-values less than 0.05, indicating that they are significantly different from zero at the 95.0% confidence level.

The R-Squared statistic indicates that the model as fitted explains 98.221% of the variability in Var_1 Tiempo CPU. The adjusted R-squared statistic, which is more suitable for comparing models with different numbers of independent variables, is 97.035%. The standard error of the estimate shows the standard deviation of the residuals to be 87.3955. The mean absolute error (MAE) of 50.0981 is the average value of the residuals. The Durbin-Watson (DW) statistic tests the residuals to determine if there is any significant correlation based on the order in which they occur in your data file. Since the P-value is greater than 5.0%, there is no indication of serial autocorrelation in the residuals at the 5.0% significance level.

En concordancia con el Anova, la gráfica de Pareto muestra que el único factor que está causando efecto en la respuesta estudiada (Tiempo CPU) es el tamaño de la **Población** (ver figura 33).

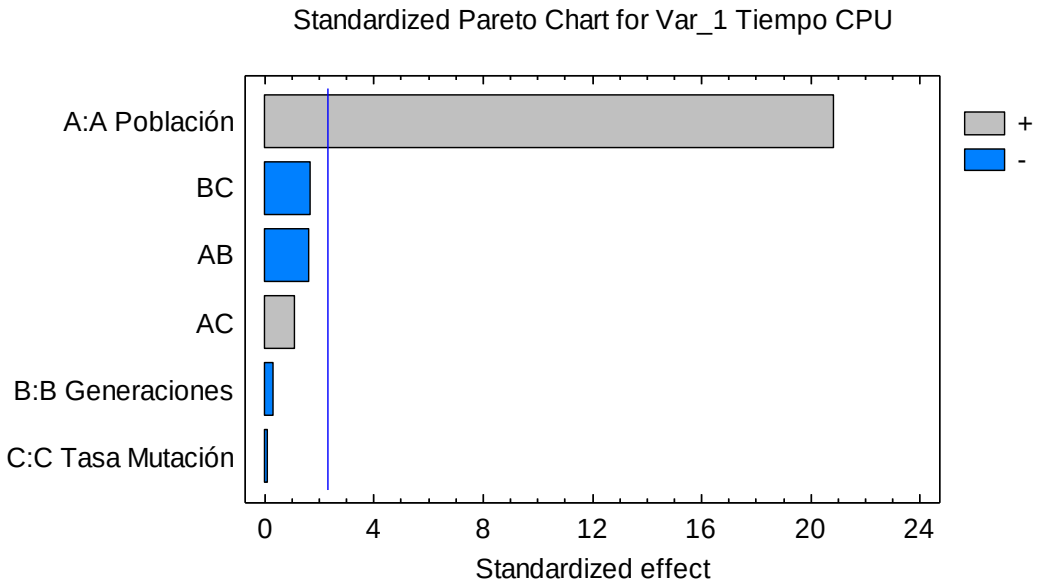


Fig. 33. Gráfica de Pareto del experimento con tres factores y dos niveles cada uno.

Asimismo la gráfica de efectos principales reitera al factor Población con efecto en el Tiempo de CPU.

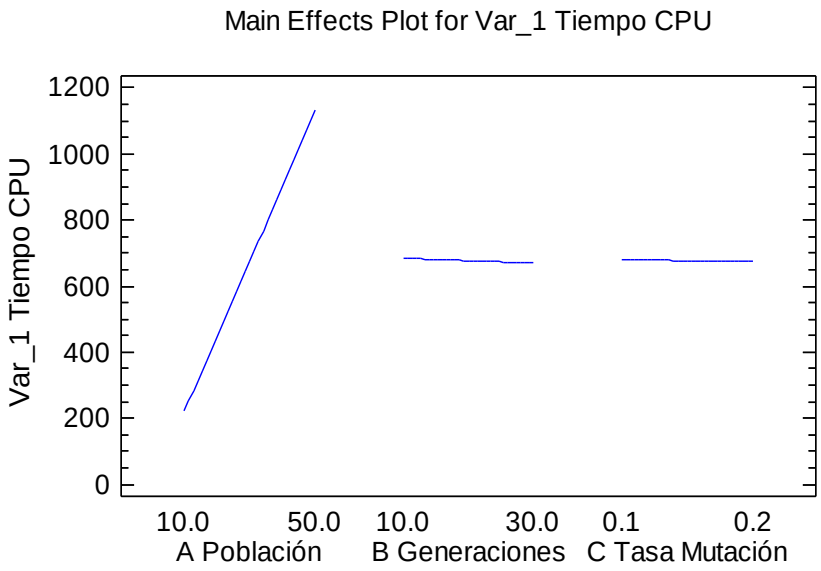


Fig. 34. Gráfica de efectos principales de los factores con dos niveles cada uno.

Interacción entre factores.

De acuerdo con la gráfica de la figura 35, se presenta interacción entre los factores, sin embargo la variación de los valores alto y bajo del número de generaciones (factor B) y de la tasa de mutación (factor C), prácticamente no representan cambio para el valor bajo de la respuesta (Tiempo de CPU). En cuanto a la interacción entre los factores B y C, esta se da a valores muy altos de la respuesta.

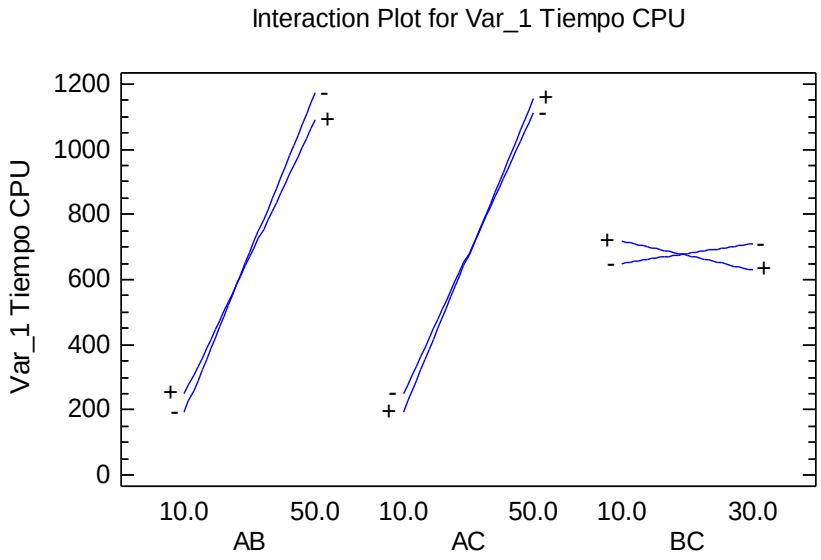


Fig. 35. Gráfica de interacciones de los factores.

Optimización de la respuesta.

Los valores de los factores de diseño que optimizan (minimizan) la respuesta (Tiempo de CPU) son: **Población 10 individuos, número de Generaciones *10* y Tasa de mutación *0.1***, los cuales corresponden a los valores bajos de los mismos.

Tabla 83. Optimización de la respuesta.

Optimize Response
Goal: minimize Var_1 Tiempo CPU

Optimum value = 183.395

Factor	*Low*	*High*	***Optimum***
A Población	10.0	50.0	**10.0**
B Generaciones	10.0	30.0	**10.0**
C Tasa Mutación	0.1	0.2	**0.1**

The StatAdvisor
This table shows the combination of factor levels which minimizes Var_1 Tiempo CPU over the indicated region. Use the Analysis Options dialog box to indicate the region over which the optimization is to be performed. You may set the value of one or more factors to a constant by setting the low and high limits to that value.

Estos valores de los factores Población y Generaciones son confirmados por medio de la gráfica de contornos para la Tasa de mutación = 0.1 (figura 36), en donde podemos apreciar que los valores bajos de la respuesta (Tiempo de CPU) se dan para valores bajos de la Población, aun cuando se maneje variación en el factor Generaciones hacia valores altos, como puede observarse en las líneas de contorno de la izquierda que corresponden a los valores bajos de la respuesta.

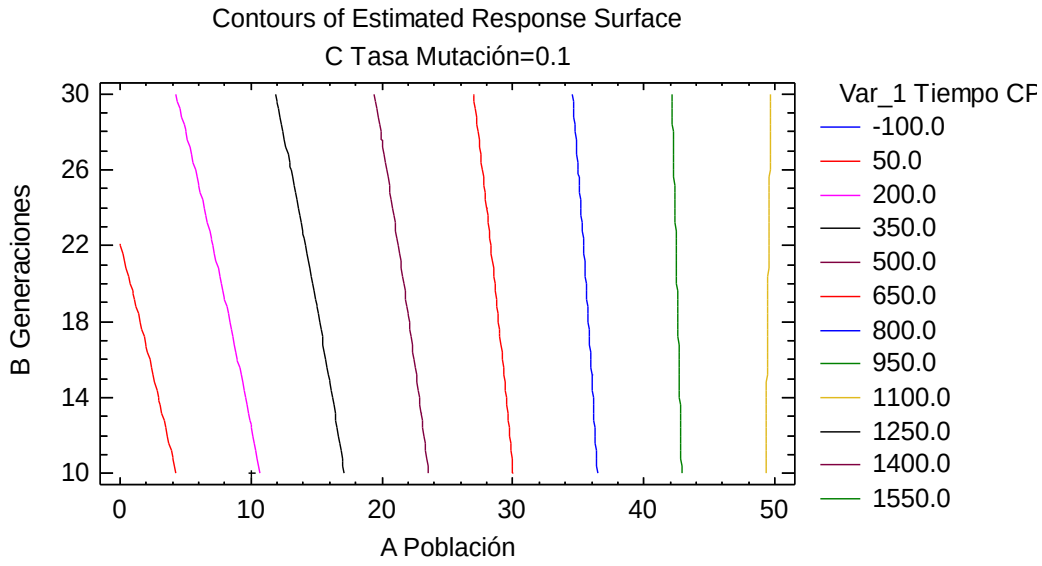


Fig. 36. Gráfica de contornos de la respuesta.

La gráfica de la superficie de respuesta muestra que los valores bajos de la Población y de las Generaciones ocasionan el menor valor en el Tiempo de CPU para la Tasa de mutación = *0.1* (figura 37).

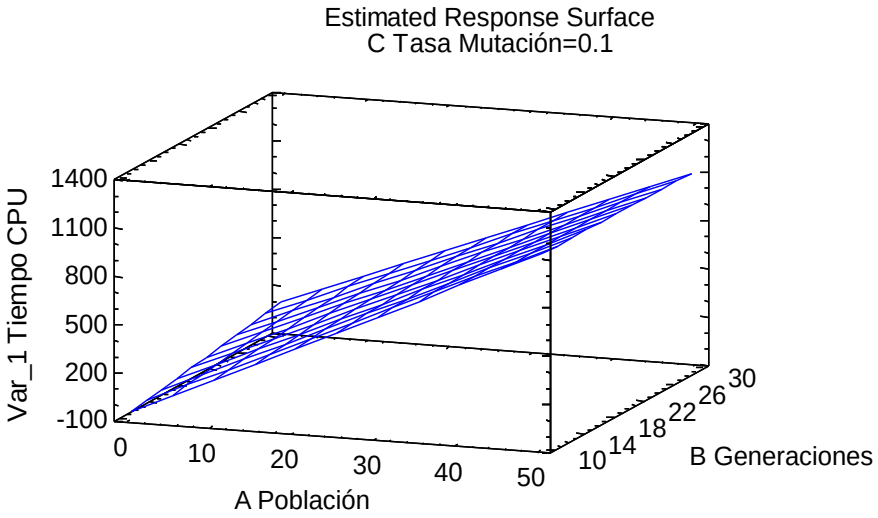


Fig. 37. Gráfica de la superficie de respuesta.

Supuesto de normalidad.

En cuanto al cumplimiento de supuestos del experimento, por lo que toca al de normalidad se puede afirmar que en general se está cumpliendo en virtud del alineamiento de los residuos con respecto a una línea recta (figura 38).

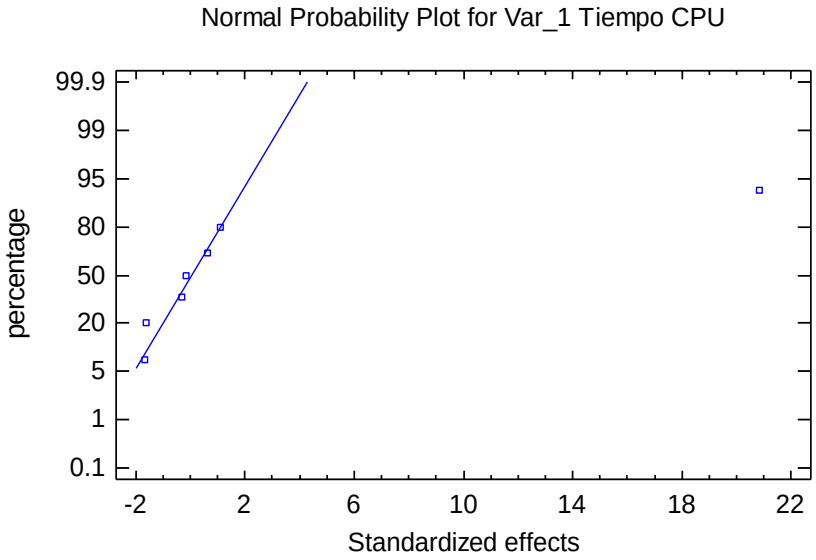


Fig. 38. Gráfica de cumplimiento del supuesto de normalidad.

Supuesto de varianza constante.

Asimismo la figura 39, que corresponde a la representación del cumplimiento de la varianza constante entre tratamientos se está cumpliendo, al no existir un patrón claro en la ubicación de los predichos vs. los residuos.

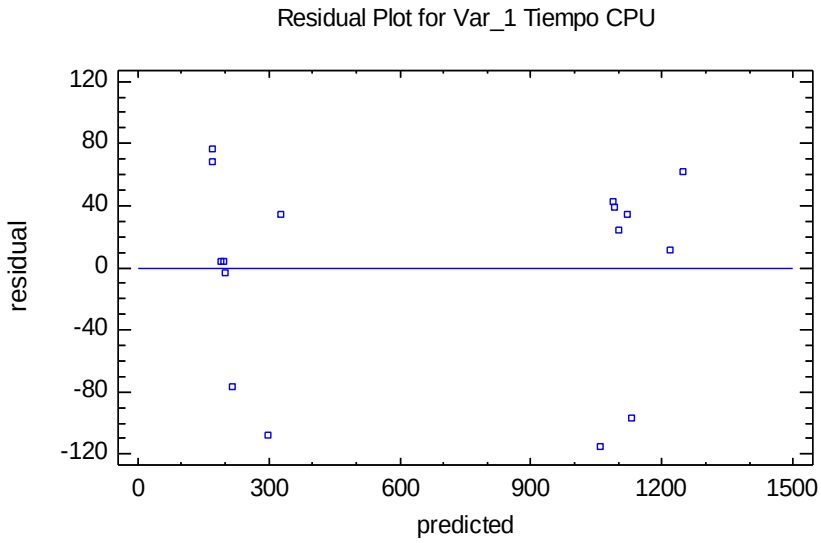


Fig. 39. Gráfica de cumplimiento del supuesto de varianza constante.

Conclusiones sobre el experimento.

Los elementos anteriores nos permiten concluir que los valores menores en la respuesta (Tiempo CPU) se obtienen con los valores menores de los factores, en particular el tamaño de la Población que se detectó que es el único que causa efecto en la respuesta. El valor menor (6 individuos) se empleó en los resultados presentados en la tesis.

La conclusión anterior era de esperarse ya que sabíamos que el tiempo de ejecución depende más bien de la conformación de la población inicial y no de la etapa de optimización en sí, que es donde juegan su rol los factores que se estudiaron en el experimento.

4.3.5. Trabajo a futuro.

Retomando lo planteado en la sección 4.3.2. Alternativas, referente a que la sub-rutina GenIndi2 genera un recorrido parcial de longitud máxima, sería conveniente que cuando el programa llegase al fin del mismo debido a un encasillamiento, se programase la continuación en el nodo de numeración más baja que se quedó pendiente, para que reinicie el proceso de construcción a partir de ese punto, evitando recomenzar desde cero. Esta continuación se podría programar utilizando la selección aleatoria de tal manera que se estuviera manejando un híbrido determinístico-aleatorio respecto a la manera de seleccionar los nodos.

Este paso se podría iterar tantas veces como fuese necesario, involucrando un procedimiento que implicara el uso de aristas auxiliares para efectuar las conexiones que requiere la continuación del recorrido en nodos no adyacentes.

Adicionalmente, un cambio de estrategia que se percibe más conveniente por los resultados que se han visualizado en la literatura para el tratamiento de este tipo de problemas, es la aplicación del enfoque del Problema de Ruteo de Arcos (o ARP, por sus siglas en inglés), del cual se han vislumbrado apenas un par de variantes y en las cuales habría que profundizar a continuación. Una de ellas es el Problema del Cartero Rural (RPP por sus siglas en inglés).

4.4. Productos entregables.

A lo largo de esta investigación se generaron los siguientes productos:

4.4.1. Reporte técnico.

Se elaboró un reporte técnico detallado de la metodología para el rediseño de rutas de recolección Recorrido Mínimo AG-TSP-Google (anexo A1).

4.4.2. Publicación de artículo.

Se publicó un artículo en el XII Encuentro Participación de la Mujer en la Ciencia en el CIO, centrado en la descripción de la metodología para el rediseño de rutas de recolección Recorrido Mínimo AG-TSP-Google y su aplicación a una pequeña ruta del Municipio de León, Guanajuato, enfocado a su divulgación. La presentación se llevó a cabo el pasado 13 mayo’15 (anexo A2).

Conclusiones.

Con base en las experiencias de este trabajo, se desarrolló una propuesta de metodología práctica y accesible para el rediseño de rutas de recolección que se espera pueda ser replicada y que consta de cinco etapas que comprenden desde la obtención del mapa de la zona con las facilidades de Google Maps, la medición de distancias con su herramienta Daft logic, la generación de la matriz de adyacencias, la ejecución de un programa AG en MATLAB y la revisión de la ruta obtenida.

Esta metodología se aplicó a 16 nodos de la ruta uno del Municipio de León, Gto. Se obtuvo un primer recorrido que incluyó una arista inexistente entre dos nodos. Se calculó e introdujo el valor correspondiente a esta arista auxiliar en la matriz de datos. Se ejecutó nuevamente el programa AG, el cual convergió a una solución en ocho generaciones. Con esto fue posible obtener el recorrido que se recomienda. Se efectuó un análisis adicional de los posibles recorridos básicos que presenta esta red, lo que permitió confirmar como mínimo el recorrido que corresponde a la secuencia 3,9,8,7,4,5,6,16,15,14,10,13,12,11,1,2,3, cuya distancia total asciende a $2{,}620\ m$. y que se presentó en detalle en la sección 4.1.

Adicionalmente se aplicó la metodología a 35 nodos de la ruta 16 del Municipio de León. Se encontró que el programa funciona bien para redes pequeñas y medianas, sin embargo al manejar redes de mayores dimensiones se detectó que el programa tarda excesivamente en la generación de la población inicial. Se concluyó que esto es debido a que redes de este tipo generan matrices dispersas, por lo cual es muy fácil caer en encasillamientos antes de lograr conformar los individuos completos correspondientes a la población inicial, impactando en el tiempo de proceso.

A la vez este tipo de redes involucran un gran número de caminos diferentes.

Con base en la experiencia anterior, se contempló como alternativa para salvar esta problemática, la sustitución de la forma en la que se eligen los nodos a agregar al recorrido que se va formando, de una manera aleatoria, a otra en la que se agregan de manera determinística. Sin embargo, los resultados fueron limitados en el sentido de que no se logró la conformación de los recorridos completos, ya que también se cayó en encasillamientos. A pesar de ello, lo anterior permitió visualizar la ventaja de generar un recorrido parcial inicial, y con ello, la posibilidad de convertir esta modificación en un procedimiento iterativo híbrido determinístico-aleatorio, que se re-ejecute tantas veces como sea necesario hasta lograr integrar en una sola ocasión el recorrido completo correspondiente a los cromosomas de la población inicial.

Para verificar la fiabilidad del algoritmo genético y del programa en que fue implementado, se efectuaron una serie de ejecuciones con diferentes instancias del TSP clásico que involucra matrices densas.

Con los resultados que se tuvieron se encontró que el programa AG trabaja bien, tanto con redes dispersas (como las de las rutas de recolección de residuos), como con redes que involucran matrices densas (como las del TSP clásico), ya que normalmente realiza la optimización en forma correcta y ágil, convergiendo además a un solo individuo.

De esta manera, se puede afirmar que tanto la metaheurística genética como el programa funcionan correctamente, ya sea en la integración de los recorridos de los individuos de la población inicial como en la etapa pura de optimización debida al Algoritmo Genético, siendo una limitante el tamaño de la red e implícitamente la matriz de datos respectiva.

Al ser un programa basado en el enfoque TSP las soluciones obtenidas representan un ciclo hamiltoniano, es decir, en él se tocan todos los nodos o cruces pero no se recorren todas las calles de la ruta. Esto es consistente con el hecho de que en la literatura frecuentemente se menciona como enfoque de solución para este tipo de problemas al llamado Problema de Ruteo de Arcos

(ARP), el cual está orientado a cubrir todas las aristas. Este enfoque, además, parece conveniente para evitar el problema del encasillamiento que se ha experimentado, al generar de una manera diferente la población inicial. Lo anterior implica la sustitución de un algoritmo ad-hoc dentro del programa AG que puede ser considerado como trabajo a futuro.

Así mismo, se contempla para etapas futuras, la inclusión de una rutina que calcule en forma automática la distancia entre nodos de una arista artificial, cuando sea requerido, mediante un algoritmo del camino más corto.

Se llevó a cabo un experimento que permitió concluir que la obtención de los valores menores en la respuesta (Tiempo CPU) se obtienen con los valores menores de los factores, en particular el tamaño de la Población que se detectó que es el único que causa efecto en la respuesta.

En este planteamiento se están obviando algunos detalles que pueden ser de importancia, como son: el volumen y peso estimado de recolección, tipo y capacidad de los vehículos, ubicación de los sitios de resguardo de los vehículos y de los vertederos de desechos.

Referencias.

Anbuudayasankar, S.P.; Ganesh, K.; Mohapatra. S. (2014). Models for Practical Routing Problems in Logistics. Design and Practices. ISBN 978-3-319-05035-5 (eBook). DOI 10.1007/978-3-319-05035-5. Springer Cham Heidelberg New York Dordrecht London.

Araujo, L. & Cervigón, C. (2009). Algoritmos Evolutivos. Un enfoque práctico. México. Alfaomega Grupo Editor, S. A. de C. V.

Ball, M. O. (2010). Review. Heuristics based on mathematical programming. University of Maryland.

Bonomo, Flavio; Duran, Guillermo; Larumbe, Frederico; et al. (2012). A method for optimizing waste collection using mathematical programming: a Buenos Aires case study. WASTE MANAGEMENT & RESEARCH, 30(3), 311-324.

Chiong, R. & Dhakal, S. (Eds.). (2009). Natural intelligence for scheduling, planning & packing problems. Springer.

Díaz, A. & Tseng, F. (1996). 1. Introducción a las técnicas heurísticas. En A. Díaz, (Coord.) Optimización Heurística y Redes Neuronales (pp. 19-36). Madrid. Editorial Paraninfo.

Daft Logic. (2015). Recuperado de http://www.daftlogic.com/projects-google-maps-distance-calculator.htm

Google Maps. (2015). Recuperado de https://www.google.com.mx/maps/@21.1193491,-101.683987,15z.

Lacomme, P; Prins, C; Ramdane-Cherif, W. (2001). APPLICATIONS OF EVOLUTIONARY COMPUTING, PROCEEDINGS Book Series: LECTURE NOTES IN COMPUTER SCIENCE, 2037, 473-483.

Lacomme, P; Prins, C; Sevaux, M. (2003). Multiobjective Capacitated Arc Routing Problem. EVOLUTIONARY MULTI-CRITERION OPTIMIZATION, PROCEEDINGS Book Series: LECTURE NOTES IN COMPUTER SCIENCE, 2632, 550-564.

Laguna, M. & Moscato, P. (1996). 3. Algoritmos Genéticos. En A. Díaz, (Coord.) Optimización Heurística y Redes Neuronales (pp. 67-104). Madrid. Editorial Paraninfo.

Maroto A., C., Alcaraz S., J. & Ruiz G., R. (2002). Investigación Operativa. Modelos y técnicas de optimización. Editorial Universidad Politécnica de Valencia.

Martí, Rafael. (2001). Procedimientos Metaheurísticos en Optimización Combinatoria. Universidad de Valencia. Curso de doctorado “Procedimientos Heurísticos”.

Russell, Stuart; Norvig, Peter. (Third Edition 2010). Artificial Intelligence. A Modern Approach. Pearson Education, Inc. New Jersey.

Salazar G., J. J. (2001). Programación Matemática. Editorial Díaz de Santos. Madrid, España.

SIAP, León. (2013). Sistema Integral de Aseo Público de León. Municipio de León, Gto., México. Legajo de información de la solicitud SSI-2013-1201 obtenida a través de la UMAIP.

SIAP, León. (2014). Sistema Integral de Aseo Público de León. Municipio de León, Gto. México. http://www.aseopublicoleon.gob.mx/#!servicios/c1iwz.

Srinivasan, G. (2010). Lecture series on Advanced Operations Research by Prof. G. Srinivasan, Department of Management Studies, IIT Madras. https://www.youtube.com/watch?v=cLsEHP0qt0&index=24&list=PL004010FEA702502F.

Taha, H. (2012 9a ed.). Investigación de operaciones. Pearson Educación México.

University of Waterloo. (2015). The Traveling Salesman Problem. Waterloo, Ontario, Canadá. Recuperado de http://www.math.uwaterloo.ca/tsp/.

Viotti, P; Polettini, A; Pomi, R; et al. (2003). Genetic, algorithms as a promising tool for optimisation of MSW collection routes. WASTE MANAGEMENT&RESEARCH 21(4) 292-98.

Zhu, Zhengyu; Xia, Mengshuang; Yang, Yong; et al. (2008). An Improved Genetic Algorithm for the Extended Capacitated Arc Routing Problem. 7TH WORLD CONGRESS ON INTELLIGENT CONTROL AND AUTOMATION, 1-23, 2017-2022.

Anexos.

A1. Reporte técnico de la metodología para el rediseño de rutas de recolección Recorrido Mínimo AG-TSP-Google.

CIATEC **CONACYT**

POSGRADO INTERINSTITUCIONAL EN CIENCIA Y TECNOLOGÍA

MAESTRÍA EN INGENIERIA INDUSTRIAL Y DE MANUFACTURA

PROYECTO:

Rediseño de las rutas de recolección de residuos sólidos urbanos en el Municipio de León, Guanajuato, México.

REPORTE TÉCNICO

Metodología para el rediseño de rutas de recolección.

Recorrido mínimo con AG-TSP-Google

ESTUDIANTE

José Luis Ramos González

TUTOR ACADÉMICO

Dr. Javier Yáñez Mendiola

CO-TUTOR

Dr. Luis Ernesto Mancilla Espinoza

León, Guanajuato, Junio 2015.

Índice

1. Introducción.

Diariamente se generan un promedio de 991 toneladas de residuos sólidos urbanos en la Ciudad de León, Gto. (SIAP León, 2014). La recolección de residuos sólidos urbanos es una función gubernamental muy importante desde los puntos de vista económico y ambiental.

Sin embargo, las rutas de recolección de residuos sólidos urbanos tradicionalmente han sido diseñadas de manera intuitiva sin considerar aspectos importantes como el modelado del sistema y la optimización del mismo mediante técnicas científicas e ingenieriles que permitan tener rutas de recolección más eficientes.

Se estableció como objetivo del proyecto proponer y aplicar una fusión de herramientas para la solución del problema del diseño de las rutas de recolección, con el fin de rediseñarlas. Derivado de lo anterior se presenta en este documento una propuesta de metodología para mejorar las rutas de recolección actuales.

Se revisó la literatura y herramientas que se han empleado para caracterizar y resolver el problema de la recolección de residuos sólidos municipales. El problema se abordó con el enfoque general para el tratamiento de los problemas de rutas de transporte conocido como el Problema del Agente Viajero (TSP por sus siglas en inglés).

En la literatura se encontraron abundantes ejemplos de aplicaciones de los Algoritmos Genéticos (AG) con resultados positivos. Se eligió esta técnica metaheurística para ser aplicada en este proyecto, por lo que se desarrolló un programa empleando el software matemático MATLAB.

Adicionalmente se utilizaron conceptos de Programación Matemática como es el caso de los grafos, aprovechando su capacidad de representar la topología de redes correspondientes a problemas de optimización de rutas. Estos elementos se complementaron con fuentes de información y herramientas informáticas disponibles a cualquier usuario tales como Google Maps y su herramienta Daft Logic para medir distancias.

Con base en los elementos anteriores se desarrolló la presente metodología nombrada Recorrido mínimo con AG-TSP-Google y que se propone para el rediseño de rutas. Cabe resaltar que para su aplicación es necesario contar con licencia del paquete MATLAB para ejecutar el programa desarrollado.

En la siguiente sección se presenta una breve descripción de las bases teóricas de los métodos metaheurísticos de los cuales forman parte los Algoritmos Genéticos, así como del Problema del Agente Viajero (TSP). Posteriormente, en la sección tres, se desarrolla

propiamente la metodología que se presenta en este documento. En la sección cuatro se muestra la aplicación de la metodología al caso de la ruta uno del municipio.

2. Marco teórico.

2.1. Métodos metaheurísticos.

El término metaheurísticos fue introducido por Fred Glover en 1986 (Martí, 2001, p.5). Osman y Kelly (1989) (citados por Martí, 2001) los definen de la siguiente manera:

> Los procedimientos **metaheurísticos** son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria en los que los heurísticos clásicos no son efectivos. Proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de la inteligencia artificial, la evolución biológica y los mecanismos estadísticos (entre otros). (Martí, 2001, p.5)

En particular, para atacar los problemas de recorridos en los que se visita un número elevado de puntos se recomiendan las técnicas metaheurísticas dado que este tipo de problemas encuentran grandes dificultades para ser resueltos con los métodos exactos.

La extensa literatura sobre el tema muestra que ha habido un gran crecimiento y desarrollo de los métodos metaheurísticos. Entre los más conocidos se enlistan aquellos relativamente consolidados y que han probado su eficacia sobre una colección amplia de problemas. Cada uno de ellos tiene sus propias características específicas.

- Algoritmos Genéticos.
- Búsqueda Tabú.
- Colonias de Hormigas.
- Enjambres de Partículas.
- Recocido Simulado.

En la siguiente sección se describen los Algoritmos Genéticos, relacionándolos con los conceptos del TSP.

2.2. Algoritmos Genéticos (AG) y el Problema del Agente Viajero (TSP).

Para hacer una descripción general de los AG, Díaz & Tseng (1996) comienzan refiriéndose a los procesos evolutivos en la naturaleza:

> …los algoritmos genéticos son "técnicas de búsqueda basadas en la mecánica de la selección natural y la genética" [Goldberg, 1989]. Su estructura se ha diseñado basándose en un intento de abstracción artificial del "algoritmo" de selección propio de la naturaleza, con la esperanza de que así se consigan éxitos similares en relación a la capacidad de adaptación a un amplio número de ambientes diferentes. Es decir, los algoritmos genéticos son flexibles y de ámbito general, y pueden ser aplicados en un amplio número de problemas.

> En los organismos biológicos, la información hereditaria es pasada a través de los cromosomas que contienen la información de todos esos factores, es decir, los genes, los cuales a su vez están compuestos por un determinado número de valores (alelos). Varios organismos se agrupan formando una población, y aquellos que mejor se adaptan son los que más probabilidades tienen de sobrevivir y reproducirse. Algunos de los supervivientes son seleccionados por la naturaleza para ser cruzados y así producir una nueva generación de organismos. Esporádicamente, los genes de un cromosoma pueden sufrir ligeros cambios (las denominadas mutaciones).
>
> Con esto, los conceptos básicos de los algoritmos genéticos ya pueden ser presentados: por lo general, los alelos son binarios (cero o uno), representando valores de las variables de decisión, que se corresponderán con los genes. Los cromosomas representan pues las soluciones. En realidad, los cromosomas no son más que tiras de bits, y en su forma más simple, un organismo puede estar formado por un simple cromosoma.
>
> Las tres operaciones antes mencionadas (reproducción, cruzamiento y mutación) han sido frecuentemente utilizadas en este tipo de algoritmos. Un algoritmo genético sencillo podría aplicar operaciones de reproducción, cruzamiento y mutación a una población y generar nuevos organismos de modo iterativo (p. 31).

Dentro de los enfoques generales para el tratamiento de los problemas de rutas de transporte se encuentra el Problema del Agente Viajero (TSP por sus siglas en inglés). Es el caso más sencillo de plantear aunque de compleja solución.

En el TSP se dispone de un sólo vehículo y una única estación. En un sólo viaje el viajero debe visitar n ciudades de forma que la distancia recorrida o sus costos c sean mínimos. Se dispone de una matriz de datos que corresponde a las distancias entre puntos o nodos (Maroto, Alcaraz & Ruiz, 2002).

Retomando la descripción que Díaz & Tseng (1996) hacen de los Algoritmos Genéticos con respecto al proceso de reproducción, mencionan que:

> …se guía a través de una función que mide el grado de adaptación del cromosoma, siendo el proceso de selección del cromosoma por reproducir dependiente de los valores que esa función le asigne: a mayor valor (para maximización, o menor valor para minimización), mayor probabilidad de selección y supervivencia. Los miembros de la nueva generación de cromosomas son de nuevo emparejados aleatoriamente (p. 32).

Este proceso de reproducción que a su vez depende del proceso de selección, enfocado al TSP, se puede implementar mediante la interpretación de un modelo matemático. Para el efecto consideremos el empleado por Maroto et al. (2002, pp.198-199):

-Sea:

> *Un grafo $G = (V, E)$ con etiquetas en los arcos o aristas (distancias) c_{ij}, y lo que se desea encontrar es un único viaje o "tour" de mínima longitud, o lo que es lo mismo, un ciclo dirigido hamiltoniano (es decir, que visite todos los puntos sólo una vez) y que contenga los n nodos del grafo.*

-Variables.

Para cada arista posible del grafo se define una variable,

$x_{ij} = 1$, *si la arista (i, j) está en el ciclo,*
$x_{ij} = 0$, *en caso contrario,*
dónde $i, j = 1,\ldots, n$

-Función objetivo.

Minimizar los costos del viaje:

$$Minimizar \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} \left(c_{i,j}\, x_{i,j}\right) \quad (1)$$

-Restricciones.

Llegadas: a cada ciudad llega el visitante (cada nodo es visitado):

$$\sum_{i=1, i\neq j}^{n} \left(x_{i,j}\right) = 1, \;\; j = 1, \ldots, n \quad (2)$$

Salidas: de cada ciudad de donde parte el visitante (cada nodo es abandonado):

$$\sum_{j=1, j\neq i}^{n} \left(x_{i,j}\right) = 1, \;\; i = 1, \ldots, n \quad (3)$$

Complementando la explicación sobre el proceso de los AG, Díaz & Tseng (1996) aclaran en qué consisten las operaciones de cruce o cruzamiento y de mutación.

> Mediante la operación de cruzamiento se intercambian genes entre la pareja. Un modo sencillo de hacerlo consiste en dividir cada una de las dos tiras de bits en dos sub-tiras y luego cambiar los valores de las últimas sub-tiras, obteniendo así dos nuevos cromosomas de una nueva generación. Las mutaciones consisten en alterar un bit de un cromosoma (de 0 a 1 ó viceversa), con una probabilidad relativamente pequeña. Todo este proceso se repite hasta que algún crtiterio de finalización se cumpla (p. 32).

Finalmente, Díaz & Tseng (1996) mencionan algunas características de los AG, de donde destaca el que esta técnica metaheurística genera poblaciones de soluciones en forma aleatoria, y no una solución simple en cada iteración.

> Los algoritmos genéticos, al menos en sus versiones más clásicas, manipulan la información en tiras de bits. Mientras la mayoría de las otras metaheurísticas generan una solución simple en cada iteración, los algoritmos genéticos trabajan sobre una población de soluciones, generando una nueva en cada iteración. Por lo general son algoritmos neutrales al contexto y no consideran ni explotan la información del problema dentro del proceso de búsqueda. La generación de nuevas soluciones es, pues, aleatoria por naturaleza. Los algoritmos genéticos utilizan una elección al azar para guiar la búsqueda, repitiendo durante un número relativamente largo de iteraciones operaciones muy simples que dan lugar a un volumen masivo de soluciones, dentro de la búsqueda de las que se consideran buenas soluciones (p. 32).

El funcionamiento general de un AG, puede ser esquematizado mediante pseudocódigo. Se presenta el propuesto por Araujo & Cervigón (2009, p.22):

```
función Algoritmo_Genético()
{
        TPoblacion pob;          // población
        TParametros parámetros// tamaño población

        obtener_parametros(parametros);
        pob = poblacion_inicial();
        evaluación(pob, tam_pob, pos_mejor, sumadaptacion);

        // bucle de evolución
        mientras no se alcanza condición de terminación hacer
        {
                seleccion(pob, parámetros);
                reproducción(pob, parámetros);
                evaluación(pob, parámetros, pos_mejor, sumadaptacion);
        }
        devolver pob[pos_mejor]
}
```

Normalmente la operación *(obtener_parametros)* incluye el tamaño de la población y la longitud del cromosoma. Las líneas *selección, reproducción y evaluación* representan procesos que se implementan por medio de funciones o subrutinas. El proceso de reproducción considera el cruce y la mutación. Para mayor abundamiento en la explicación es posible consultar el texto de Araujo & Cervigón (2009), Algoritmos Evolutivos. Un enfoque práctico.

Los mismos autores explican el funcionamiento del algoritmo descrito:

> El algoritmo procesa un conjunto de individuos que forman la población *pob.* Al comienzo del algoritmo se obtienen los datos de entrada al problema *(obtener_parametros)* y se genera la población inicial, cuyos individuos se evalúan mediante la función de adaptación del algoritmo. El resto del algoritmo consiste en un bucle, cada una de cuyas iteraciones es una generación en la que se produce un proceso de selección, que da mayores probabilidades de tener copias en la nueva población a los individuos más adaptados, seguido de un proceso de reproducción en el que se generan nuevos individuos a partir de los de la población mediante operaciones de mezcla y pequeñas alteraciones, y finalmente una evaluación de la nueva población. En muchas ocasiones se utilizan pequeñas variantes de este esquema. Así, por ejemplo, a veces se selecciona un subconjunto de la población, que es el único que participa en las operaciones de reproducción. (pp. 22-23)

Este esquema general puede ser aplicable a cualquier algoritmo evolutivo (v.gr. AG's, AG's de codificación real, Programas Evolutivos y otros).

> Siguiendo este esquema general se han desarrollado distintas variantes de algoritmos evolutivos, cuya principal diferencia se encuentra en la representación de los individuos. Lógicamente, los operadores genéticos que se utilizan para la reproducción en cada caso dependen de la representación adoptada. (Araujo & Cervigón, 2009, pp. 23)

Si bien la representación de los individuos o cromosomas de la población tienen un papel muy importante, evidentemente el problema que se estudie va a dar lugar a un algoritmo en específico.

> …dentro de cada una de esas clases, cada algoritmo es un caso particular, no sólo en función del problema concreto al que se aplica, sino también dependiendo de la elección concreta de los métodos que se aplican en los distintos procesos involucrados, como la selección y la reproducción. (Araujo & Cervigón, 2009, pp. 28)

En este sentido resulta necesaria la utilización de un algoritmo orientado al TSP, lo que se trata en la siguiente subsección.

2.3. Algoritmos Genéticos (AG) aplicados al Problema del Agente Viajero (TSP).

La siguiente es una descripción de los elementos del AG propuesto por Taha (2012) para el TSP, los cuales fueron tomados como base para codificar el programa en MATLAB:

1. Codificación de genes (nodos o vértices) del cromosoma (individuos).
   Esto se realiza mediante la representación numérica directa de los nodos del recorrido (por ejemplo *1-2-5-4-3-1*).
2. Población inicial.
   El primer paso es identificar las aristas que salen de cada nodo en la red, lo cual se hace a partir de la matriz de adyacencias.
   El siguiente paso es la construcción del recorrido, lo que se lleva a cabo comenzando en un nodo origen específico, agregando en la posición extrema a la derecha un nodo único no redundante, seleccionado de entre todos los que salen del último nodo agregado.
3. Creación de un hijo.
   Primeramente se seleccionan los dos mejores padres, mediante la evaluación de su función de su aptitud (longitud de recorrido).
   A continuación se realiza un intercambio de genes (cruce) para la creación de dos hijos mediante el procedimiento de cruce ordenado.
4. Mutación en los genes de los hijos.
   Se lleva a cabo mediante el intercambio de nodos de dos posiciones seleccionadas al azar (con excepción del nodo u origen), con pequeña probabilidad de ocurrencia *(0.1)*. (pp.423-424)

Esta descripción se complementa agregando un bucle que incluye la sustitución de los dos peores individuos de la generación anterior por los nuevos hijos creados, así como la evaluación de la nueva población para repetir los pasos descritos en los pasos 4 y 5 hasta que se cumpla una condición de terminación.

Goldberg (citado por Araujo & Cervigón, 2009) propuso el llamado algoritmo genético simple, el cual utilizó para explicar con claridad el funcionamiento de los AG's en general. Se toma como referencia para presentar una comparación contra el algoritmo propuesto por Taha (2012) para el TSP, mismo que se empleó para desarrollar el programa en MATLAB de este trabajo (Tabla 1):

Tabla 1. Comparativo AG Simple (Goldberg) vs. AG para TSP (Taha).

Elemento	AG Simple (Goldberg)	TSP (Taha)
Representación de individuos	Binaria	Numérica directa (ejemplo 1-4-5-2-3)
Generación de la población inicial	Aleatoria	Inicio en un nodo elegido al azar. Continuación del recorrido agregando sucesivamente un nodo de entre los que son adyacentes al ultimo.
Selección	Por ruleta, o proporcional a la adaptación relativa de los individuos	Directa (los dos individuos mejor adaptados de la generación anterior).
Cruce	Monopunto	Cruce ordenado. Incluye armar el recorrido preferentemente con nodos adyacentes.
Mutación	Aleatoria bit a bit	Intercambio de dos genes (nodos) elegidos al azar.
Reemplazo	De los padres	Los dos individuos peor adaptados de la generación anterior.

3. Metodología para el rediseño de rutas de recolección - Recorrido mínimo con AG-TSP-Google.

La metodología desarrollada consta de las siguientes fases:

6. Preparación de datos.
7. Medición de aristas con Daft Logic.
8. Generación de la matriz de adyacencias ponderada.
9. Ejecución del programa AG.
10. Verificación de la ruta obtenida.

La figura 1 esquematiza las fases de la metodología.

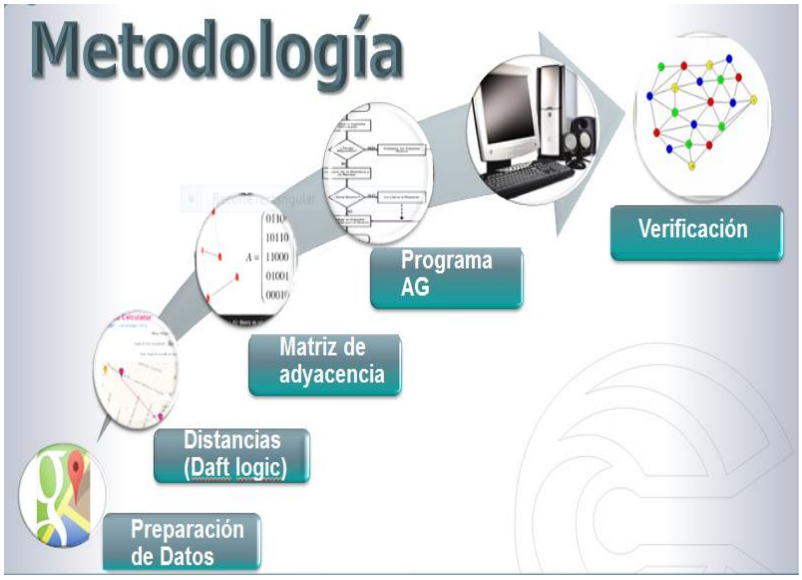


Fig. 1. Metodología para el rediseño de rutas de recolección.

A continuación se explican sus detalles.

3.1. Preparación de datos.

3.1.1 Empleando como referencia la información de la zona correspondiente a la ruta que se desea estudiar, básicamente sus delimitaciones, se descarga de Google Maps el mapa respectivo.

3.1.2 Se puede dibujar o marcar el grafo dirigido de la red en una copia del mapa obtenido de Google Maps.

3.1.3 Se elabora la relación de calles, se determinan sus cruces y se etiquetan los nodos correspondientes.

3.1.4 Tomando en cuenta los sentidos de las calles se elabora la matriz de adyacencias unitaria. Esta matriz de adyacencias se genera en un archivo Excel, capturando *(1's)* en los cruces de las filas y columnas correspondientes a los nodos donde existe conexión entre nodos consecutivos. En la primera hoja de este archivo debe ir la matriz de adyacencias sin columnas, renglones ni ningún dato adicional a los que corresponden exclusivamente a las adyacencias. De tal

manera que la matriz se puede desarrollar previamente en hojas internas del archivo, ya que normalmente para su elaboración son muy útiles, al menos, una columna y un renglón de identificación de los nodos.

3.1.5 Se nombra el archivo identificando la ruta correspondiente, asignándole un nombre, por ejemplo: *nombre de archivo con ruta.xlsx,* el cual va a ser empleado posteriormente en la etapa de ejecución del programa AG.

3.2. Medición de aristas con Daft Logic.

3.2.1 Se accede a la herramienta Daft Logic de Google Maps y se descarga el mapa correspondiente a la zona.

3.2.2 Se miden una a una las aristas del grafo para obtener las distancias correspondientes.

3.3. Generación de la matriz de adyacencias ponderada.

3.3.1 Con estos datos se convierte la matriz de adyacencias unitaria en la matriz de adyacencias ponderada, sustituyendo los *(1's)* por los valores de las distancias previamente obtenidos.

3.3.2 Para el manejo de los casos en los que no hay adyacencia entre nodos, se agrega un valor muy alto, *9999*, en los elementos respectivos de la matriz.

3.4. Ejecución del programa AG.

3.4.1 Se alimenta la matriz de adyacencias ponderada registrando el nombre del archivo Excel al inicio del programa. La instrucción de lectura del archivo es:

d = xlsread('nombre de archivo con ruta.xlsx')

3.4.2 Se confirman o redefinen los parámetros del programa. Esto se realiza en las líneas correspondientes al inicio del programa. Para fines ilustrativos se muestran valores pequeños en el caso del número de generaciones y el tamaño de la población. Generalmente se recomiendan valores bajos para la tasa de mutación:

Número de generaciones = 10
Tamaño de la población = 6 individuos (recorridos)
Tasa de probabilidad de mutación = 0.1

Estos parámetros se pueden modificar en función de los resultados que arroje el programa tomando en cuenta que sus variaciones influirán en la convergencia del programa pero también en sus tiempos de ejecución. Esto dependerá naturalmente del tamaño de la red alimentada. Se probó que al menos para redes pequeñas (menos de 30 nodos) estos parámetros funcionan eficientemente para este programa.

3.4.3 El programa AG produce el número de generaciones especificado. Presenta los cromosomas o individuos formados en cada una de sus iteraciones mostrando el recorrido propuesto a través de la secuencia de genes o nodos para cada uno de ellos.

La 1ª generación presenta la población inicial creada aleatoriamente por el AG con la característica de que los nodos en cada individuo no se repiten salvo el nodo inicial, condicionando el armado del recorrido a que exista adyacencia entre nodos, con excepción del último eslabón, si es que es necesario para poder cerrar el ciclo.

En general el programa AG busca formar los recorridos usando sólo eslabonamientos dados por la adyacencia entre nodos. En particular la operación de cruce se lleva a cabo tratando de integrar nodos que cumplan con esta condición. La operación de mutación puede generar aristas no existentes al hacer su intercambio de nodos. El programa AG en su desempeño recompone la conformación correcta de los individuos.

Dada la topología de la red es posible que para cerrar un ciclo, el programa considere eslabonamientos correspondientes a aristas no existentes. Esta es la razón por la cual el programa AG puede requerir su ejecución en dos fases. En la primera se genera una propuesta de recorrido que incluye una (o más) aristas no existentes, es decir con valores muy altos, *9999*.

3.4.4 Lo anterior da lugar a la creación de aristas auxiliares que se definen y se calculan por inspección aplicando el criterio del camino más corto, y cuyas distancias totales se introducen en la matriz de datos para efectuar una nueva corrida del programa en una 2ª fase.

El programa AG entrega el resultado final cuando se cumple la condición de terminación, generalmente el número de iteraciones definido. El programa puede converger o no, dependiendo de si presenta el mismo recorrido para todos sus individuos o cromosomas en la última generación, aun cuando estos individuos marquen inicios en nodos diferentes, lo cual no tiene relevancia si el recorrido es el mismo.

Lo que se busca es que no existan tramos con valores muy altos, *9999*.

3.5. Verificación de la ruta obtenida.

3.5.1 Se puede marcar el grafo dirigido de la red correspondiente a la solución en una copia del mapa de Google Maps verificando los sentidos.

3.5.2 Con base en el recorrido obtenido y los datos de la matriz de adyacencias ponderada se confirma la distancia total. Esto puede realizarse en una hoja Excel para facilidad de cálculo.

4. Aplicación de la metodología para el rediseño de rutas de recolección - Recorrido mínimo con AG-TSP-Google.

Se muestra la aplicación de la metodología a 16 nodos (cruces de calles) de la ruta uno del municipio.

4.1. Preparación de datos.

A partir de la información de las delimitaciones de la zona se descargó de Google Maps (2015) el mapa correspondiente a los 16 nodos de la ruta uno (Fig. 2).

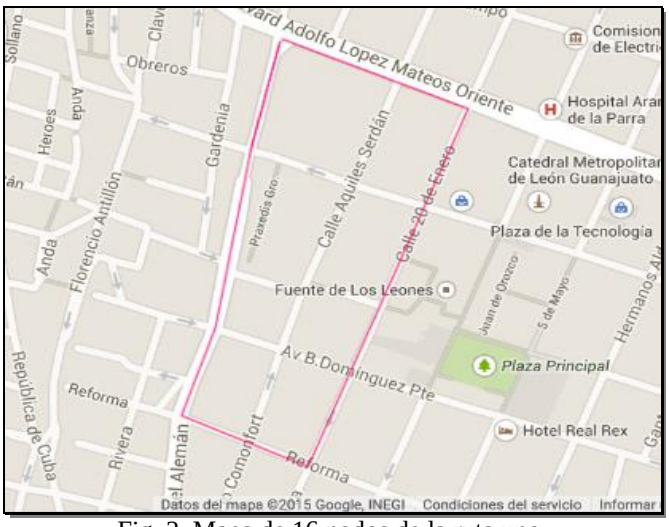


Fig. 2. Mapa de 16 nodos de la ruta uno.

Se dibujó el grafo de la red (Fig. 3).

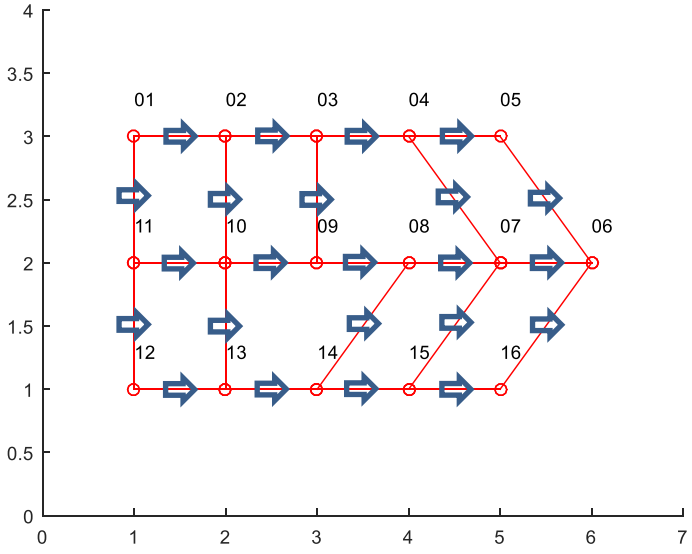


Fig. 3. Grafo de la ruta uno; 16 nodos.

Se elaboró la relación de calles y se determinaron los sentidos, restricciones y otras características de las vialidades (ver tabla 2).

Tabla 2. Relación de calles

	Calle	Sentido
1	Blvd. Miguel Alemán	Sur – Norte
2	Ignacio Comonfort/Aquiles Serdán	Sur - Norte
3	Justo Sierra/20 de Enero	Norte - Sur
4	Reforma	Oriente - Poniente
5	Belisario Domínguez	Poniente - Oriente
6	Josefa Ortiz de Domínguez	Poniente - Oriente
7	Alvaro Obregón	Oriente - Poniente
8	Blvd. Adolfo López Mateos	Poniente – Oriente

Se definieron los cruces de calles y se etiquetaron los nodos (tabla 3).

Tabla 3. Etiquetado de nodos.

Nodo	Calle	Esquina con
1	Blvd. M. Alemán	Reforma
2	Blvd. M. Alemán	Belisario Domínguez
3	Blvd. M. Alemán	Josefa Ortiz de Domínguez
4	Blvd. M. Alemán	Alvaro Obregón
5	Blvd. M. Alemán	Bvd. Adolfo López Mateos
6	I. Comonfort/Aquiles.Serdán	Bvd. Adolfo López Mateos
7	I. Comonfort/Aquiles Serdán	Alvaro Obregón
8	I. Comonfort/Aquiles Serdán	Josefa Ortiz de Domínguez
9	I. Comonfort/Aquiles Serdán	Josefa Ortiz de Domínguez
10	I. Comonfort/Aquiles Serdán	Belisario Domínguez
11	I. Comonfort/Aquiles Serdán	Reforma
12	Justo Sierra/20 Enero	Reforma
13	Justo Sierra/20 Enero	Belisario Domínguez
14	Justo Sierra/20 Enero	Josefa Ortiz de Domínguez
15	Justo Sierra/20 Enero	Alvaro Obregón
16	Justo Sierra/20 Enero	Bvd. Adolfo López Mateos

Se generó la matriz de adyacencias unitaria (tabla 4) a partir de los cruces de calles, tomando en cuenta los sentidos. Se utilizó un archivo Excel, el cual se nombró como: *MatrizAdyRuta1_16nodos.xlsx.*

Tabla 4. Matriz de adyacencias unitaria.

		N			O			D			O			S			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
N	1		1														
	2			1							1						
	3				1					1							
O	4					1											
	5						1										
	6																1
D	7				1		1										
	8							1							1		
	9								1								
O	10									1				1			
	11	1									1						
	12											1					
S	13												1				
	14													1			
	15							1							1		
	16															1	

4.2. Medición de aristas con Daft Logic.

Con la herramienta Daft Logic (2014) se midieron las aristas del grafo para obtener las distancias. En la figura 4 se muestra la medición de la arista incidente en los nodos *1* y *2*. El valor obtenido *149.823* se redondeó a *150 m*. el cual se capturó en el elemento de la matriz que corresponde a la adyacencia entre los dos nodos (fila *1*, columna *2*), ver tabla 5.

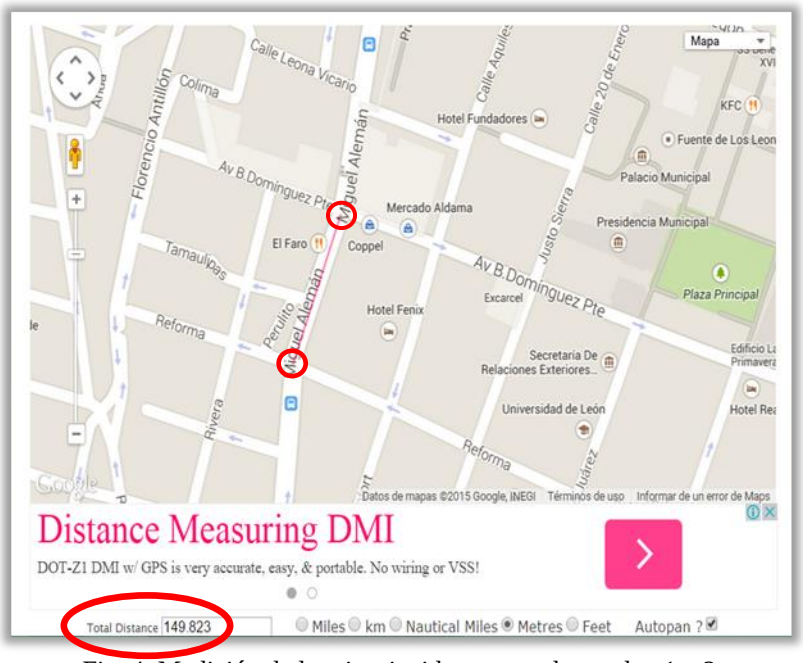


Fig. 4. Medición de la arista incidente entre los nodos *1* y *2*.

4.3. Generación de la matriz de adyacencias ponderada.

Con los datos anteriores se convirtió la matriz de adyacencias unitaria en matriz de adyacencias ponderada. Para el efecto se copió la matriz unitaria en otra hoja del archivo *MatrizAdyRuta1_16nodos.xlsx* dónde se substituyeron los *(1's)* por los valores de las distancias de las aristas en metros. Se incluyó el valor *9999* en los elementos de la matriz en los que no hay adyacencia entre nodos. Finalmente se copió esta hoja al principio del archivo, eliminando la primera fila y la primera columna, las cuales sirvieron para identificación de los nodos (tabla 5).

Tabla 5. Matriz de adyacencias ponderada.

9999	0150	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	0100	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999	9999
9999	9999	9999	0210	9999	9999	9999	9999	0110	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	0200	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120
9999	9999	9999	0160	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	0180	9999	9999	9999	9999	9999	9999	0120	9999	9999
9999	9999	9999	9999	9999	9999	9999	0020	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	0110	9999	9999	9999
0100	9999	9999	9999	9999	9999	9999	9999	9999	0160	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0170	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120	9999	9999	9999
9999	9999	9999	9999	9999	9999	0120	9999	9999	9999	9999	9999	9999	0180	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0180	9999

4.4. Ejecución del programa AG.

Se alimentó la matriz de adyacencias ponderada al programa mediante el registro del nombre del archivo *MatrizAdyRuta1_16nodos.xlsx*, en la instrucción:

d = xlsread(' MatrizAdyRuta1_16nodos.xlsx ')

Se definieron los siguientes parámetros del programa. Dado el tamaño de la red se establecieron valores pequeños en el número de generaciones y en el tamaño de la población. El correspondiente a la tasa de mutación se dejó también con un valor bajo como generalmente se maneja:

Número de generaciones = 10
Tamaño de la población = 6 individuos (recorridos)
Tasa de probabilidad de mutación = 0.1

La corrida inicial del programa AG presentó los individuos formados en cada una de sus iteraciones. A continuación se analiza en detalle cada una de las diez generaciones.

En la tabla 6 se presenta el resultado correspondiente a la 1ª generación. En ella se aprecian los seis individuos generados por el programa los cuales corresponden a la población inicial creada aleatoriamente por el AG con la característica de que los nodos no se repiten salvo el nodo inicial. Podemos apreciar que hay dos pares de individuos idénticos el *1* y el *3*, y el *5* y *6*. Los individuos *2* y *4* son diferentes a los demás.

Para fines de análisis del desempeño del AG nos enfocaremos en el individuo *6*.

En esta primera generación, el AG consideró al final del recorrido, para cerrar el ciclo, el arista *7-1* que no existe, ya que no hay adyacencia entre esos nodos. Por lo anterior, incluyó el valor *9999* en el acumulado, dando una longitud de recorrido de *12,179 m*. Este valor es uno de los dos más altos de entre los seis individuos, por lo que se convierte en un candidato para ser substituido en la siguiente generación.

Tabla 6. Resultado de ejecución inicial del programa AG. 1ª generación.

Generación 1																		
individuo					R	e	c	o	r	r	i	d	o					distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	1	2	3	4	5	6	16	15	14	13	12	11	10	9	8	7	1	12,179
6	**1**	**2**	**3**	**4**	**5**	**6**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**1**	**12,179**

En la 2ª generación (tabla 7), al aplicarse los procesos de selección y reproducción (cruce y mutación) se produjo un nuevo individuo 6 con características más deficientes que el inicial, al pasar de *12,179* a *21,898 m*. en su longitud de recorrido, ya que incluyó dos aristas no existentes, la *15-10* y la *10-14*. Nuevamente éste es candidato a ser sustituido dentro del proceso del AG, ya que presenta el valor más alto de entre los seis individuos.

Tabla 7. Resultado de ejecución inicial del programa AG. 2ª generación.

generacion 2																		
individuo					R	e	c	o	r	r	i	d	o					distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**14**	**13**	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**15**	**10**	**14**	**21,898**

La 3ª generación (tabla 8), después de llevar a cabo el proceso del AG, presenta un individuo 6 con un peor resultado, *41,566 m.*, ya que en esta ocasión se incluyeron cuatro aristas inexistentes, la *13-2*, la *16-12*, la *1-15* y la *14-10*. De nueva cuenta este individuo va a ser substituido en la siguiente iteración del AG, dado que presenta uno de los dos valores más grandes dentro del conjunto de individuos.

Tabla 8. Resultado de ejecución inicial del programa AG. 3ª generación.

generacion 3																		
Individuo					R	e	c	o	r	r	i	d	o					Distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	16	15	14	13	12	11	1	5	6	4	2	10	3	51,445
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**10**	**13**	**2**	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**12**	**11**	**1**	**15**	**14**	**10**	**41,566**

El valor que presenta la 4ª generación, correspondiente a la función de adaptación del individuo *6, 21,828 m*., es mejor que la anterior (ver tabla 9). Considera sin embargo las aristas *14-7* y *15-10* que son inexistentes. El mismo individuo continua formando parte del par de individuos que van a ser sustituidos en la siguiente generación del proceso.

Tabla 9. Resultado de ejecución inicial del programa AG. 4ª generación.

generacion 4																		
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	1	2	3	9	8	7	4	5	6	16	15	14	13	12	11	10	1	12,139
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**10**	**13**	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**14**	**7**	**4**	**5**	**6**	**16**	**15**	**10**	**21,828**

En la nueva iteración del proceso (5ª generación, tabla 10) se produjo un individuo con un nuevo deterioro en su función de adaptación (*31,627 m*.) ya que incluye tres aristas inexistentes, la *8-5*, la *4-14* y la *14-10,* por lo que el mismo individuo continúa como candidato a mejorar en el siguiente proceso.

Tabla 10. Resultado de ejecución inicial del programa AG. 5ª generación.

generacion 5																		
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	4	5	6	16	15	2	3	9	8	14	10	13	12	11	1	7	4	31,677
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**1**	**2**	**3**	**9**	**8**	**5**	**6**	**16**	**15**	**7**	**4**	**14**	**10**	**13**	**12**	**11**	**1**	**31,627**

Según la tabla 11 (6ª generación) el proceso del AG produjo un nuevo individuo 6 con peores características que el anterior ya que incluye cuatro aristas inexistentes (*9-5, 14-10, 10-8 y 4-13*) totalizando una longitud de recorrido de *41,736 m*. Por consiguiente, este individuo va a ser sustituido en la siguiente iteración.

Tabla 11. Resultado de ejecución inicial del programa AG. 6ª generación.

generación	6																	
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	11	1	2	3	9	8	7	4	15	14	10	5	6	16	13	12	11	41,576
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**3**	**9**	**5**	**6**	**16**	**15**	**14**	**10**	**8**	**7**	**4**	**13**	**12**	**11**	**1**	**2**	**3**	**41,736**

El nuevo sustituto del individuo *6* permaneció, en la 7ª generación, en un nivel similar de longitud de recorrido (*41,576* m.) ya que consideró las aristas no existentes *4-14, 14-10, 10-5* y *15-13* (tabla 12). Junto con el individuo *1*, el individuo *6* va ser substituido por otro en la siguiente ocasión.

Tabla 12. Resultado de ejecución inicial del programa AG. 7ª generación.

generacion	7																	
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	3	9	8	5	6	16	15	14	10	7	4	13	12	11	1	2	3	41,576
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**7**	**4**	**14**	**10**	**5**	**6**	**16**	**15**	**13**	**12**	**41,576**

La nueva función de adaptación del individuo *6* desciende a *31,737 m.* ya que incluye las aristas no existentes *14-12, 7-10* y *13-4* (tabla 13). Vuelve a quedar como candidato a ser removido y substituido en el siguiente proceso.

Tabla 13. Resultado de ejecución inicial del programa AG. 8ª generación.

generacion	8																	
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
6	**15**	**14**	**12**	**11**	**1**	**2**	**3**	**9**	**8**	**7**	**10**	**13**	**4**	**5**	**6**	**16**	**15**	**31,737**

En esta ocasión (9ª generación) el programa produjo un individuo con mejores características para sustituir al seis, ya que sólo incluyó la arista inexistente (*14-10),* arrojando una longitud de recorrido de sólo *12,069 m.* De hecho este mismo valor se repitió en otros cuatro individuos. El individuo *6* ya no es candidato a priori para ser substituido en el siguiente ciclo, ya que al igual que los otros cuatro individuos tiene el mismo valor en su función de adaptación o de aptitud (tabla 14).

Tabla 14. Resultado de ejecución inicial del programa AG. 9ª generación.

generacion 9																		
individuo					R	e	c	o	r	r	i	d	o					distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	13	12	11	1	2	10	3	12,079
5	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
6	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**15**	**14**	**10**	**13**	**12**	**11**	**1**	**2**	**3**	**12,069**

Finalmente el programa converge en la 10ª generación (tabla 15) entregando los mismos cromosomas (individuos). Aun cuando el individuo *2* inició en un nodo diferente no tiene relevancia ya que el recorrido es el mismo. El valor de la función de adaptación quedó en *12,069 m*. incluyendo el arista inexistente *14-10*.

Tabla 15. Resultado de ejecución inicial del programa AG. 10ª generación.

generacion 10																		
individuo					R	e	c	o	r	r	i	d	o					distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
2	10	13	12	11	1	2	3	9	8	7	4	5	6	16	15	14	10	12,069
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
4	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
5	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	12,069
6	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**15**	**14**	**10**	**13**	**12**	**11**	**1**	**2**	**3**	**12,069**

Dada la topología de la red, fue necesario alimentar en el programa la distancia de la arista entre los nodos *14* y *10* que no siendo adyacentes, fue requerida para cerrar el recorrido y terminar la ejecución del algoritmo. Se observa por inspección que el recorrido que une estos dos nodos es: *14, 13, 12, 11* y *10*, el cual suma una distancia total de *550 m.*, misma que se agregó a la matriz de datos (tabla 16).

Tabla 16. Matriz de adyacencias ponderada con arista auxiliar 14-10.

9999	0150	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	0100	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	0210	9999	9999	9999	9999	0110	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	0200	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0120
9999	9999	9999	0160	9999	0190	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	0180	9999	9999	9999	9999	9999	9999	0120	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	0020	9999	9999	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	0110	9999	9999	9999	9999
0100	9999	9999	9999	9999	9999	9999	9999	9999	0160	9999	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0100	9999	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0170	9999	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	0550	9999	9999	0120	9999	9999	9999	9999
9999	9999	9999	9999	9999	9999	0120	9999	9999	9999	9999	9999	9999	0180	9999	9999	9999
9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	0180	9999	9999

Con la matriz de adyacencias actualizada se ejecutó nuevamente el programa AG. El programa convergió en ocho generaciones ya que entregó los mismos cromosomas (individuos) desde esa iteración. El resultado de la ejecución final (10ª generación) se observa en la tabla 17.

Tabla 17. Resultado final de ejecución del programa AG.

generacion 10																		
Individuo	R	e	c	o	r	r	i	d	o									distancia
1	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
2	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
3	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
4	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
5	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620
6	3	9	8	7	4	5	6	16	15	14	10	13	12	11	1	2	3	2,620

4.5. Verificación de la ruta obtenida.

Se representó la solución obtenida mediante el siguiente grafo (Fig. 6).

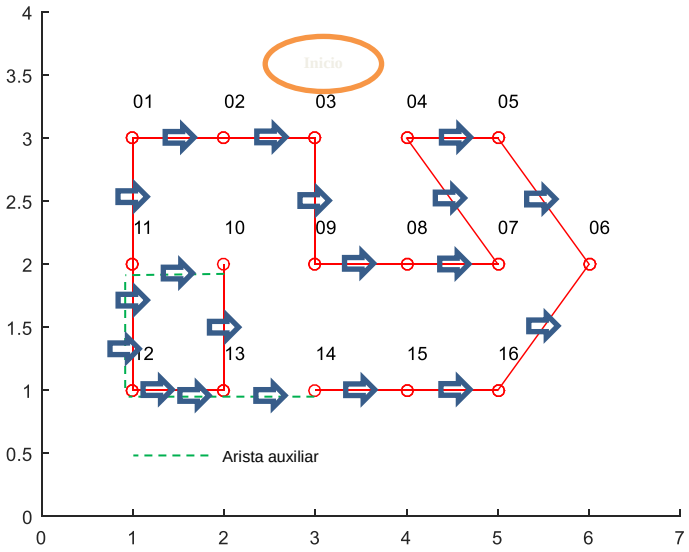


Fig. 6. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 14-10.

Como se puede apreciar, de acuerdo con la definición del TSP, el recorrido es cerrado. Sin embargo, para que esto pudiera lograrse fue necesario considerar el arista auxiliar 14-10 representada con línea punteada.

Siguiendo el recorrido obtenido y con los datos de la matriz de adyacencias ponderada se verifica la distancia total (tabla 18).

Tabla 18. Distancia total del recorrido.

Arista	De nodo	Al nodo	Distancia
1	3	9	110
2	9	8	20
3	8	7	180
4	7	4	160
5	4	5	200
6	5	6	190
7	6	16	120
8	16	15	180
9	15	14	180
10	14	10	550
11	10	13	110
12	13	12	170
13	12	11	100
14	11	1	100
15	1	2	150
16	2	**3**	100
		suma	2620

Al calcularse y agregarse el dato correspondiente a la arista auxiliar *14-10* se obtuvo el resultado del recorrido igual a *2620 m*. el cual se considera correcto, mismo que fue validado mediante varias ejecuciones del programa, lo cual se discute a continuación.

Se considera válido iniciar el recorrido en un nodo diferente del que originalmente propuso el programa AG en su solución, lo cual dependerá de aspectos operativos que así lo recomienden. Por ejemplo, el mismo recorrido puede ser iniciado y terminado en el nodo 1 que se encuentra ubicado en una esquina. Esto nos ejemplifica los casos en los que iniciar en un nodo diferente podría conducir a un recorrido más práctico y menos costoso

4.6. Otros resultados del recorrido – Casos básicos.

La ejecución reiterada del programa arroja diferentes resultados en el recorrido propuesto. El análisis de estos recorridos permite concluir que existen sólo algunos recorridos básicos para esta red, sin embargo se presentan variantes relativas al nodo de inicio.

Además del que se analizó en la subsección anterior, al cual le llamaremos caso 1, a continuación se presentan los que se identificaron como recorridos básicos.

La tabla 19 muestra un recorrido inicial generado por el programa, nombrado aquí como caso básico 2. En este caso la propuesta incluye el arista inexistente *7-10*. Si esta arista se calcula por fuera y se introduce el valor correspondiente (que asciende a *1,220* m.) en la matriz de datos del programa, el resultado final del recorrido resulta ser de *3,220 m*., el cual es mayor que el obtenido como resultado final (*2,620 m*.) del apartado anterior.

Tabla 19. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 2.

generacion 10																		
Individuo	R	e	c	o	r	r	i	d	o									distancia
1	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
2	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
3	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
4	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
5	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999
6	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999

Se presenta el grafo correspondiente al caso básico 2 en la figura 7.

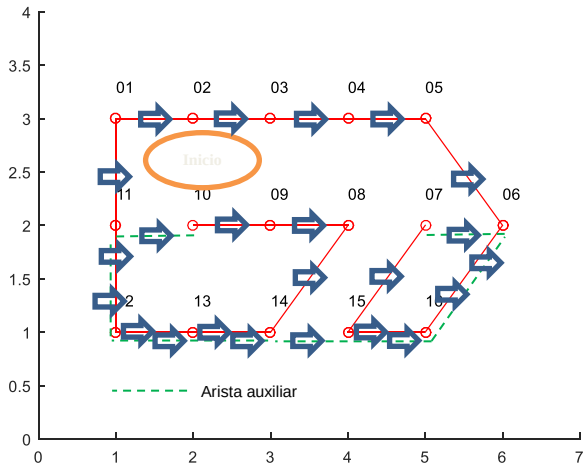


Fig. 7. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 7-10.

Para entregar un circuito cerrado, el programa AG consideró el arista auxiliar 7-10 al final (línea punteada).

La tabla 20 muestra otro recorrido inicial generado por el programa (caso básico 3). Incluye el arista inexistente *3-10*. Cabe resaltar al individuo 4 que muestra un recorrimiento en la posición de sus genes pero que sin embargo representa al mismo individuo de los otros renglones. Si se calcula el arista auxiliar *3-10* que corresponde a *800 m.* y se introduce en la matriz de datos del programa, el resultado final es de *2,870 m*. también mayor que el obtenido como mejor resultado final (*2,620 m*).Se observa el grafo del caso básico 3(fig.8).

Tabla 20. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 3.

generacion 10																		
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
2	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
3	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
4	10	9	8	7	4	5	6	16	15	14	13	12	11	1	2	3	10	12,069
5	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069
6	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069

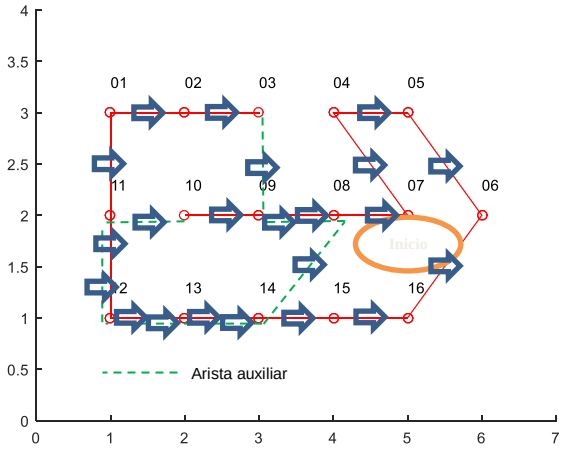


Fig. 8. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 3-10.

En la tabla 21 se presenta otro resultado inicial diferente (caso básico 4). En este caso el arista inexistente viene siendo la *7-3*, la cual vale *1,410 m.*, de tal manera que el nuevo resultado final sería *3,530 m.*, más elevado que el mejor (*2,620 m.*).

Tabla 21. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 4.

generacion 10																		
individuo					R	e	c	o	r	r	i	d	o					distancia
1	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
2	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
3	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
4	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
5	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119
6	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119

La figura 9 muestra el grafo correspondiente al caso básico 4. En este caso el cierre del recorrido nuevamente se da al final precisamente con el arista auxiliar 7-3.

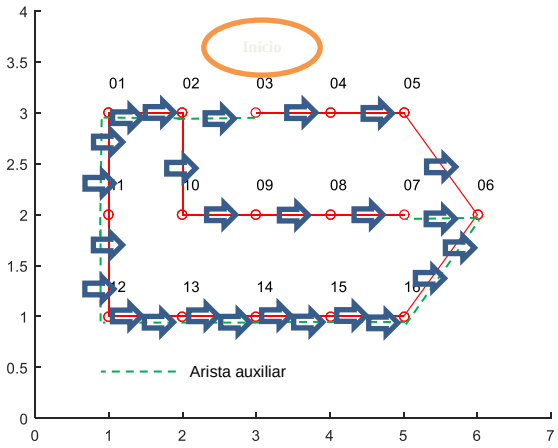


Fig. 9. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 7-3.

El caso básico 5 se muestra en la tabla 22. Contempla el arista inexistente *10-3*, con valor de *730 m*. El nuevo resultado final viene siendo *3,170 m*. Se puede observar el grafo correspondiente en la figura 10.

Tabla 22. Resultado inicial de ejecución del programa Algoritmo Genético. Caso básico 5.

generacion 10																		
Individuo					R	e	c	o	r	r	i	d	o					distancia
1	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
2	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
3	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
4	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
5	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079
6	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079

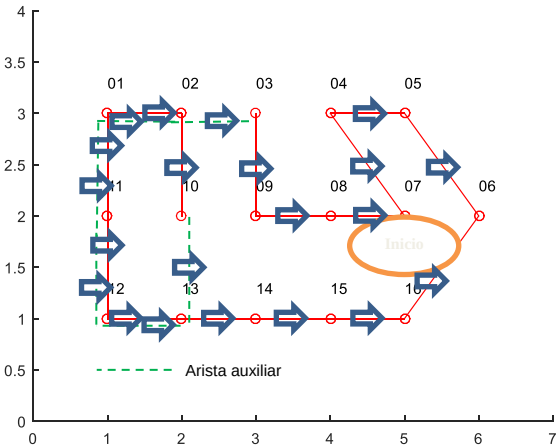


Fig. 10. Grafo de la solución TSP ruta uno; 16 nodos con arista auxiliar 10-3.

Resumiendo lo anterior en la tabla 23, se puede discernir el recorrido inicial que produce el recorrido mínimo que corresponde al caso 1 presentado en la subsección 4.4.

Tabla 23. Diferentes resultados del recorrido. Casos básicos.

Recorrido																		Distancia inicial m.	Arista auxiliar m.	Recorr. final m.
1	**3**	**9**	**8**	**7**	**4**	**5**	**6**	**16**	**15**	**14**	**10**	**13**	**12**	**11**	**1**	**2**	**3**	12,069	550	2,620
2	10	9	8	14	13	12	11	1	2	3	4	5	6	16	15	7	10	11,999	1,220	3,220
3	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9	8	7	12,069	800	2,870
4	3	4	5	6	16	15	14	13	12	11	1	2	10	9	8	7	3	12,119	1,410	3,530
5	7	4	5	6	16	15	14	13	12	11	1	2	10	3	9	8	7	12,079	730	2,810

Sin embargo, hay que considerar la posibilidad de elegir alguno de los otros recorridos de acuerdo con criterios operativos que sea necesario tomar en cuenta. Por ejemplo, que la primera recolección deba efectuarse en un punto (nodo) específico de la ruta. O bien algunas de las recolecciones deban respetar un cierto orden por diversos motivos (v.gr. horarios pactados, aspectos viales, etc.), que hagan recomendable elegir una ruta diferente a aquella con el recorrido mínimo.

5. Conclusiones.

Se diseñó una metodología para el rediseño de rutas de recolección que consta de cinco fases que comprenden desde la obtención del mapa de la zona, con las facilidades de Google Maps, la medición de distancias con su herramienta Daft logic, la generación de la matriz de adyacencias, la ejecución de un programa de AG en MATLAB y la revisión de la ruta obtenida.

Esta metodología se aplicó a 16 nodos de la ruta uno del Municipio de León, Guanajuato. Se obtuvo un primer recorrido que incluyó una arista inexistente entre dos nodos. Se calculó e introdujo el valor correspondiente a esta arista auxiliar en la matriz de datos. Se ejecutó nuevamente el programa AG, el cual convergió a una solución en ocho generaciones. Con esto fue posible obtener el recorrido que se recomienda.

Se efectuó un análisis adicional de los posibles recorridos básicos que presenta esta red, lo que permitió confirmar como mínimo el recorrido que corresponde a la secuencia 3,9,8,7,4,5,6,16,15,14,10,13,12,11,1,2,3, cuya distancia total asciende a 2,620 m. y que se presenta en detalle en la subsección 4.4.

Al ser un programa basado en el enfoque TSP la solución obtenida representa un ciclo hamiltoniano, es decir, en él se tocan todos los nodos o cruces pero no se recorren todas las calles de la ruta.

Se contempla, para etapas futuras, la inclusión de una rutina que calcule en forma automática la distancia entre nodos de una arista artificial, cuando sea requerido, mediante un algoritmo del camino más corto.

En este planteamiento se están obviando algunos detalles que pueden ser de importancia, como son: el volumen y peso estimado de recolección, tipo y capacidad de los vehículos, ubicación de los sitios de resguardo de los vehículos y de los vertederos de desechos.

6. Bibliografía.

Araujo, L. & Cervigón, C. (2009). Algoritmos Evolutivos. Un enfoque práctico. México. Alfaomega Grupo Editor, S. A. de C. V.

Díaz, A. & Tseng, F. (1996). 1. Introducción a las técnicas heurísticas. En A. Díaz, (Coord.) Optimización Heurística y Redes Neuronales (pp. 19-36). Madrid. Editorial Paraninfo.

Daft Logic. (2014). Recuperado de http://www.daftlogic.com/projects-google-maps-distance-calculator.htm.

Google Maps. (2015). Recuperado de https://www.google.com.mx/maps/@21.1193491,-101.683987,15z.

Maroto A., C., Alcaraz S., J. & Ruiz G., R. (2002). Investigación Operativa. Modelos y técnicas de optimización. Editorial Universidad Politécnica de Valencia.

Martí, Rafael. (2001). Procedimientos Metaheurísticos en Optimización Combinatoria. Universidad de Valencia. Curso de doctorado "Procedimientos Heurísticos".

SIAP, León. (2014). Sistema Integral de Aseo Público de León, Gto. Municipio de León, Gto. México. http://www.aseopublicoleon.gob.mx/#!servicios/c1iwz.

Taha, H. 2012. (9a ed.). Investigación de operaciones. Pearson Educación México.

A2. Artículo publicado en el XII Encuentro Participación de la Mujer en la Ciencia. CIO (2015).

Rediseño de las rutas de recolección de residuos sólidos urbanos en el Municipio de León Guanajuato, México.

José Luis Ramos[a], Javier Yáñez[a], Luis Ernesto Mancilla[b],

[a]Centro de Innovación Aplicada en Tecnologías Competitivas, León, Gto., jramos.picyt@ciatec.mx
[b]Instituto Tecnologico de Leon (SEP)

RESUMEN:

A pesar de que la recolección de residuos sólidos urbanos es una función gubernamental muy importante desde los puntos de vista económico y ambiental, tradicionalmente las rutas de recolección han sido diseñadas de manera intuitiva por parte de quienes tienen a cargo esa responsabilidad.

En este trabajo se tiene como objetivo proponer y aplicar una fusión de herramientas con el fin de rediseñar las rutas de recolección. El problema se abordó con el enfoque general para el tratamiento de los problemas de rutas de transporte conocido como el Problema del Agente Viajero (TSP por sus siglas en inglés). Se utilizaron conceptos de Programación Matemática como es el caso de los grafos, relacionándolo con la técnica metaheurística Algoritmos Genéticos. Estos elementos se complementaron con fuentes de información y herramientas informáticas disponibles a cualquier usuario como es el caso de Google Maps y su herramienta Daft Logic para medir distancias. Se desarrolló un programa de Algoritmos Genéticos en MatLab aplicándolo a una ruta del municipio. Con base en esta experiencia se desarrolló una metodología que se propone para el rediseño de rutas.

1. INTRODUCCIÓN

Actualmente se generan un promedio de 991 toneladas de residuos sólidos urbanos diariamente en la Ciudad de León, Gto. (Sistema Integral de Aseo Público, SIAP León, 2014).

Las rutas de recolección de residuos sólidos urbanos tradicionalmente han sido diseñadas de manera intuitiva sin considerar aspectos importantes como el modelado del sistema y la optimización del mismo mediante técnicas científicas e ingenieriles que permitan tener rutas de recolección mas eficientes.

Se tiene como objetivo aplicar diferentes herramientas de solución al problema del diseño de las rutas de recolección, con el fin de rediseñarlas. Se aplicó el enfoque general para el tratamiento de estos problemas conocido como el Problema del Agente Viajero (TSP por sus siglas en inglés).

Dado que en la literatura sobre el tema de ruteo de vehículos son abundantes los ejemplos, con resultados positivos, de aplicaciones de los Algoritmos Genéticos (AG), se eligió esta técnica metaheurística para ser aplicada en este proyecto. Más adelante se presentan algunos ejemplos.

En términos sencillos, el AG aplicado al TSP, consiste en la selección de los dos mejores padres, de entre una población, para crear dos hijos. Los hijos creados a partir de un cruce de los padres

son mejorados por medio de mutaciones, las cuales ocurren con una baja probabilidad. Posteriormente estos hijos reemplazan a los dos padres menos aptos. La aptitud se mide en función de la longitud de recorrido, a menor longitud mayor aptitud y viceversa. El proceso de crear hijos y retirar padres se repite hasta que se presenta una condición de terminación (Taha, 2012).

Lacomme, Prins & Ramdane-Cherif (2001), mencionan que los algoritmos exactos están aún limitados para problemas pequeños y que los metaheurísticos son requeridos para instancias de gran escala. Los autores publican el primer AG para el Problema de Ruteo de Arcos Capacitado (CARP, por sus siglas en inglés), planteando un híbrido que ataca extensiones realistas como vueltas prohibidas y grafos mixtos.

Posteriormente, continuando con el planteamiento del CARP, Lacomme, Prins & Sevaux, (2003), buscan minimizar la longitud del viaje más largo además de la longitud de la ruta total (el único criterio minimizado en el problema académico). Para el efecto, estos investigadores presentan un AG bi-objetivo para este CARP más realista nunca estudiado en la literatura hasta ese momento. Sobre esta base, incluyen dos características clave: uso de heurísticas constructivas para sembrar la población inicial y la hibridación con un procedimiento de búsqueda local.

Viotti, Polettini, Pomi, et al. (2003), argumentan que hasta la fecha, las rutas óptimas se han desarrollado de acuerdo con metodologías intuitivas y experiencia de campo, y que para analizar estas complejidades se usan aplicaciones que normalmente se basan en procedimientos heurísticos que permiten soluciones de alta calidad pero que en el tema computacional tienen una complejidad que es una limitación por la calidad de las soluciones calculadas y para la representación exacta de las zonas urbanas. Ellos emplean una metodología alternativa que se basa en un AG.

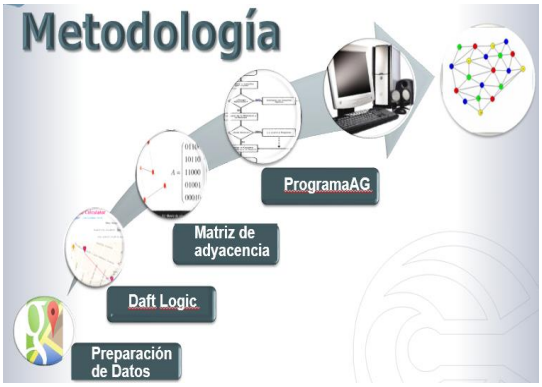
Zhu, Xia, Yang, et al. (2008), presentan un AG mejorado para resolver una versión extendida del CARP (ECARP) como vueltas prohibidas o problemas de encharcamientos. El nuevo algoritmo mejora su estructura, evitando el fenómeno de la convergencia prematura. La mejora propuesta puede resolver el ECARP eficazmente, y la comparación entre la mejora de AG y un algoritmo memético clásico, muestra que este algoritmo es más eficaz y puede conseguir mejores resultados en la resolución del CARP básico de gran tamaño.

Por su parte, Bonomo, Duran, Larumbe, et al. (2012), proponen un método que utiliza técnicas de investigación de operaciones para optimizar las rutas de los vehículos de recolección de residuos de contenedores. El problema de recolección de residuos se reduce al problema del viajante de comercio clásico (TSP). El enfoque de solución emplea la teoría de grafos y herramientas de programación matemática. También se discute el proceso de corrección de datos.

2. DESARROLLO

Después de revisar la literatura y las herramientas que se han empleado para caracterizar y resolver el problema de la recolección de residuos sólidos municipales, se plantea la siguiente metodología.

Metodología para el rediseño de las rutas de recolección.
La figura muestra las fases de la metodología. A continuación se explican sus detalles.



Metodología para el rediseño de rutas de recolección

a) Preparación de datos básicos

Se obtuvo información del SIAP a través de la Unidad Municipal de Acceso a la Información, consistente en las zonas correspondientes a las rutas que se tienen implementadas en el Municipio.

Tomando como referencia esta información es posible descargar los mapas respectivos de Google Maps. Se elabora la relación de calles, se determinan sus cruces y se etiquetan los nodos correspondientes. Tomando en cuenta los sentidos de las calles se genera la matriz de adyacencias. Esto permite dibujar el grafo dirigido de la red.

b) Generación de la matriz de adyacencia ponderada

Se miden las aristas del grafo con la herramienta Daft Logic de Google Maps para obtener las distancias correspondientes. Con estos datos se convierte la matriz de adyacencias en la matriz de adyacencias ponderada. Para dirigir el programa en el manejo de los casos en los que no hay adyacencia entre nodos, se agrega un valor muy alto, 9999, en los elementos respectivos de la matriz.

c) Ejecución del programa AG.

Se desarrolló en MatLab el programa del AG propuesto por Taha (2012) para resolver el problema del TSP.

Se alimenta la matriz de adyacencias ponderada; se definen los parámetros correspondientes y se ejecuta el programa del AG. Si es necesario, dada la topología de la red, se calculan aristas auxiliares, las cuales se definen como aquellas en las que no hay adyacencia entre dos nodos, pero que el programa las requiere para cerrar un ciclo y terminar la ejecución del algoritmo. Esto se determina cuando el programa, en el resultado de una primera corrida incluye un valor muy alto (9999) en su recorrido. El valor de las aristas auxiliares se puede calcular por inspección, sumando los tramos correspondientes al recorrido más corto que une los dos nodos. Se dibuja el grafo de la solución.

d) Verificación de distancias de la ruta obtenida

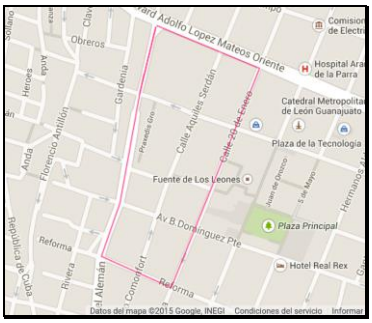
Con base en el recorrido obtenido y los datos de la matriz de adyacencias ponderada se verifica la distancia total.

3. RESULTADOS

Se trabajó con la ruta 1 del municipio, de la cual se tomaron 16 nodos (cruces de calles).

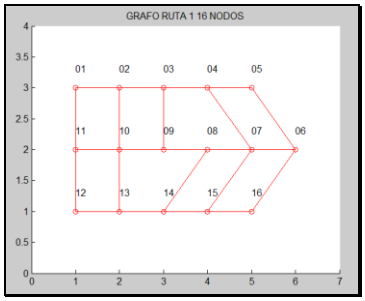
a) Preparación de datos básicos

A partir de la información del SIAP se descargó de Google Maps (2015) el mapa de la zona correspondiente a la ruta 1.



Mapa de 16 nodos de la ruta 1

Se elaboró la relación de calles y se determinaron los sentidos, restricciones y otras características de las vialidades. Se definieron los cruces de calles y se etiquetaron los nodos. Se generó la matriz de adyacencias a partir de los cruces de calles, tomando en cuenta los sentidos. Se dibujó el grafo de la red.



Grafo de la ruta 1 16 nodos

b) Generación de la matriz de adyacencia ponderada

Con la herramienta Daft Logic (2014) se midieron las aristas del grafo para obtener las distancias. Con estos datos se convirtió la matriz de adyacencias a matriz de adyacencias ponderada. Se incluyó el valor 9999 en los elementos de la matriz en los que no hay adyacencia entre nodos.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	9999	**0150**	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
2	9999	9999	**0100**	9999	9999	9999	9999	9999	9999	**0100**	9999	9999	9999	9999	9999	9999
3	9999	9999	9999	**0210**	9999	9999	9999	9999	**0110**	**0800**	9999	9999	9999	9999	9999	9999
4	9999	9999	9999	9999	**0200**	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
5	9999	9999	9999	9999	9999	**0190**	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
6	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	**0120**
7	9999	9999	9999	**0160**	9999	**0190**	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999
8	9999	9999	9999	9999	9999	9999	**0180**	9999	9999	9999	9999	9999	9999	9999	**0120**	9999
9	9999	9999	9999	9999	9999	9999	9999	**0020**	9999	9999	9999	9999	9999	9999	9999	9999
10	9999	9999	9999	9999	9999	9999	9999	9999	**0100**	9999	9999	9999	**0110**	9999	9999	9999
11	**0100**	9999	9999	9999	9999	9999	9999	9999	9999	**0160**	9999	9999	9999	9999	9999	9999
12	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	**0100**	9999	9999	9999	9999	9999
13	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	**0170**	9999	9999	9999	9999
14	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	**0120**	9999	9999	9999
15	9999	9999	9999	9999	9999	9999	**0120**	9999	9999	9999	9999	9999	9999	**0180**	9999	9999
16	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	9999	**0180**	9999

Matriz de adyacencias ponderada

c) Ejecución del programa AG.

Se alimentó la matriz de adyacencias ponderada y se definieron parámetros del programa:

Número de nodos = 16
Tamaño de la población = 6 individuos (recorridos)
Número de generaciones = 10
Tasa de probabilidad de mutación = 0.1

Dada la topología de la red, fue necesario alimentar en el programa la distancia entre los nodos 3 y 10, que no son adyacentes, pero que fue requerida para cerrar un ciclo y terminar la ejecución del algoritmo. El recorrido que une estos dos nodos es: 3, 9, 8, 14, 13, 12, 11, 10, el cual suma una distancia total de 800, misma que fue agregada a la matriz de datos.

El resultado de la ejecución fue el siguiente:

generacion =

10

individ =

16	15	14	13	12	11	1	2	3	10	9	8	7	4	5	6	16
16	15	14	13	12	11	1	2	3	10	9	8	7	4	5	6	16
16	15	14	13	12	11	1	2	3	10	9	8	7	4	5	6	16
9	8	7	4	5	6	16	15	14	13	12	11	1	2	3	10	9
16	15	14	13	12	11	1	2	3	10	9	8	7	4	5	6	16
16	15	14	13	12	11	1	2	3	10	9	8	7	4	5	6	16

distancia =

2870 2870 2870 2870 2870 2870

>>

Resultado de ejecución del programa Algoritmo Genético

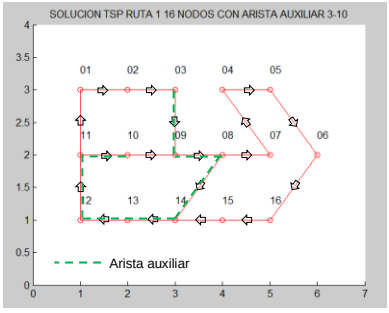
d) Verificación de distancias de la ruta obtenida

Siguiendo el recorrido obtenido y con los datos de la matriz de adyacencias ponderada se verifica la distancia total.

	Del nodo	Al nodo	Distancia
1	16	15	180
2	15	14	180
3	14	13	120
4	13	12	170
5	12	11	100
6	11	1	100
7	1	2	150
8	2	3	100
9	3	10	**800**
10	10	9	100
11	9	8	20
12	8	7	180
13	7	4	160
14	4	5	200
15	5	6	190
16	6	16	120
		suma	**2870**

Distancia total del recorrido

Se representó la solución obtenida mediante el dibujo del grafo correspondiente.



Grafo de la solución

4. CONCLUSIONES.

Al calcularse y agregarse el dato correspondiente a la arista artificial 3-10 se obtuvo el resultado del recorrido igual a 2870 el cual se considera correcto, mismo que fue validado mediante varias ejecuciones del programa.

El programa converge a una solución en 10 generaciones presentando un ciclo hamiltoniano, es decir, se visitan todos los cruces pero no todas las calles de la ruta. Esto es consistente con el hecho de que en la literatura frecuentemente se menciona como enfoque de solución para este tipo de problemas al llamado Problema de Ruteo de Arcos (ARP, por sus siglas en inglés). Lo anterior implica la sustitución de un algoritmo ad-hoc dentro del programa AG para etapas futuras. Así mismo se contempla la inclusión de una rutina que calcule en forma automática la distancia entre nodos de una arista artificial cuando sea requerido, mediante un algoritmo del camino más corto.

En este planteamiento se están obviando algunos detalles que pueden ser de importancia, como son: el volumen y peso estimado de recolección, tipo y capacidad de los vehículos, ubicación de los sitios de resguardo de los vehículos y de los vertederos de desechos.

Con base en las experiencias de este trabajo, se definió una propuesta de metodología práctica y accesible para el rediseño de rutas que se espera pueda ser replicada en otras.

BIBLIOGRAFÍA

1. SIAP, León. 2014. Sistema Integral de Aseo Público de León, Gto. Municipio de León, Gto. México. http://www.aseopublicoleon.gob.mx/#!servicios/c1iwz.
2. Taha, H. 2012. 9a ed. Investigación de operaciones. Pearson Educación México.
3. Lacomme, P; Prins, C; Ramdane-Cherif, W. (2001). APPLICATIONS OF EVOLUTIONARY COMPUTING, PROCEEDINGS Book Series: LECTURE NOTES IN COMPUTER SCIENCE, 2037, 473-483.
4. Lacomme, P; Prins, C; Sevaux, M. (2003). Multiobjective Capacitated Arc Routing Problem. EVOLUTIONARY MULTI-CRITERION OPTIMIZATION, PROCEEDINGS Book Series: LECTURE NOTES IN COMPUTER SCIENCE, 2632, 550-564.
5. Viotti, P; Polettini, A; Pomi, R; et al. (2003). Genetic, algorithms as a promising tool for optimisation of MSW collection routes. WASTE MANAGEMENT&RESEARCH 21(4) 292-98
6. Zhu, Zhengyu; Xia, Mengshuang; Yang, Yong; et al. (2008). An Improved Genetic Algorithm for the Extended Capacitated Arc Routing Problem. 7TH WORLD CONGRESS ON INTELLIGENT CONTROL AND AUTOMATION, 1-23, 2017-2022.
7. Bonomo, Flavio; Duran, Guillermo; Larumbe, Frederico; et al. (2012). A method for optimizing waste collection using mathematical programming: a Buenos Aires case study. WASTE MANAGEMENT & RESEARCH, 30(3), 311-324
8. Google Maps. 2015. Recuperado de https://www.google.com.mx/maps/@21.1193491,-101.683987,15z
9. Daft Logic. 2014. Recuperado de http://www.daftlogic.com/projects-google-maps-distance-calculator.htm