ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

VICERRECTORADO DE INVESTIGACIÓN, INNOVACIÓN Y TRANSFERENCIA DE TECNOLOGÍA

CENTRO DE POSGRADOS

MAESTRÍA EN LA ENSEÑANZA DE LA MATEMÁTICA

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE MAGÍSTER EN ENSEÑANZA DE LA MATEMÁTICA

TEMA:

OPTIMIZACIÓN DE RUTAS DE MANTENIMIENTO, APLICANDO UNA RED NEURONAL TIPO HOPFIELD Y EL ALGORITMO KNAPSACK

AUTOR:

ING. ARÉVALO LUZURIAGA, MARCELO JAVIER

DIRECTOR:

MAT. MEDINA VÁSQUEZ, PAÚL LEONARDO PhD

SANGOLQUÍ

2019

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

VICERRECTORADO DE INVESTIGACIÓN, INNOVACIÓN Y TRANSFERENCIA DE TECNOLOGÍA

CENTRO DE POSGRADOS

CERTIFICACIÓN

Certifico que el trabajo de titulación, **"OPTIMIZACIÓN DE RUTAS DE MANTENIMIENTO, APLICANDO UNA RED NEURONAL TIPO HOPFIELD Y EL ALGORITMO DE KNAPSACK"** fue realizado por el señor Arévalo Luzuriaga, Marcelo Javier el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 14 septiembre 2018

Firma:

Mat. Paúl Leonardo Medina Vásquez PhD

C.C.: 1712227295.

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

VICERRECTORADO DE INVESTIGACIÓN, INNOVACIÓN Y TRANSFERENCIA DE TECNOLOGÍA

CENTRO DE POSGRADOS

AUTORÍA DE RESPONSABILIDAD

Yo, **Arévalo Luzuriaga, Marcelo Javier**, con cédula de ciudadanía n°1714290077, declaro que el contenido, ideas y criterios del trabajo de titulación: **"OPTIMIZACIÓN DE RUTAS DE MANTENIMIENTO, APLICANDO UNA RED NEURONAL TIPO HOPFIELD Y EL ALGORITMO DE KNAPSACK"** es de mi autoría y responsabilidad, cumpliendo con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

Sangolquí, 30 de noviembre 2018

Firma:

.....

Ing. Marcelo Javier Arévalo Luzuriaga

C.C.: 1714290077.

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

VICERRECTORADO DE INVESTIGACIÓN, INNOVACIÓN Y TRANSFERENCIA DE TECNOLOGÍA

CENTRO DE POSGRADOS

AUTORIZACIÓN

Yo, **Arévalo Luzuriaga, Marcelo Javier** autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **"OPTIMIZACIÓN DE RUTAS DE MANTENIMIENTO, APLICANDO UNA RED NEURONAL TIPO HOPFIELD Y EL ALGORITMO DE KNAPSACK"** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 30 de noviembre 2018

Firma:

.....

Ing. Marcelo Javier Arévalo Luzuriaga

C.C.: 1714290077.

Dedicatoria

El presente trabajo lo dedico principalmente a Dios por bendecir mi vida, y a San Judas Tadeo por ser el inspirador, darme fuerza y protegerme cada día. A mis hijos Marcelo, Nicolás y Martín la razón de mi existencia y el orgullo de mi vida. A mi Daniela por acompañarme siempre y darme su amor incondicional. A mis padres por su apoyo y confianza en todos los momentos de mi vida. Y a toda mi familia por estar siempre presente a mi lado brindándome su cariño.

Esto es para ustedes ... .

Agradecimientos

Agradezco a Dios por permitirme cumplir este objetivo y culminarlo con éxito.
A la compañera de mi vida Daniela por su ayuda, comprensión y apoyo durante la ejecución de este logro.
A mi hijo Marcelo por acompañarme siempre y darme su fuerza, el mejor amigo que la vida me dio.
A mis padres por estar siempre pendientes.
Un agradecimiento especial al Dr. Paúl Medina por su dirección y guía en el desarrollo de esta enriquecedora experiencia.
Gracias,

Marcelo Arévalo

enero 2019

Índice general

Índice de tablas

Índice de figuras

Resumen

Muchas empresas de servicios logísticos o con procesos que involucren movilizar personal para ejecutar las actividades de servicio mantienen una necesidad común, optimizar las rutas de los recorridos, disminuyendo tiempos de movilización. En este trabajo se analizará una empresa situada en Quito, que brinda este servicio en todo el país. Mediante el estudio de distintas rutas disponibles y efectivas (vías transitables con vehículo), se buscará la optimización de estas. Para ello, utilizando criterios de problemas clásicos como el agente viajero y técnicas convencionales como las redes neuronales se buscará la ruta más óptima. Este trabajo pretende desarrollar una aplicación (software) capaz de realizar la optimización de una ruta de recorrido para un número finito de puntos, que serán visitados y además presentarán varias características que deben considerarse: *Prioridad de puntos a visitar, cada punto tendrá una prioridad que vendrá dada por un valor cuantitativo asignado por el cliente. Disponibilidad de rutas vehiculares, el acceso de cada punto a otro debe darse en función de las rutas de acceso vehicular que existan. Días laborables de disponibilidad para que el personal cumpla la ruta asignada*.Para el desarrollo e implementación del proyecto se utilizó la plataforma MATLAB donde se construyó una Red Neuronal tipo Hopfield, programación lineal y la teoría de grafos, obteniendo un software instalable para PC, que puede ser utilizado en cualquier ordenador con plataforma de Windows 7 o superior, de igual manera se desarrollará un Add-in instalable para trabajar en una hoja de cálculo del software Excel de Office.

PALABRAS CLAVES:

- **RED NEURONAL HOPFIELD**
- **PROBLEMA DEL AGENTE VIAJERO TSP**
- **KNAPSACK**

Abstract

Many logistic service companies or that have processes that involve mobilizing personnel to execute service activities maintain a common need, optimize route routes and decrease mobilization times. In this research we will analyze a company located in the city of Quito, which provides this type of services throughout the country. Through the study of the different available and effective routes (passable roads with vehicle), the optimization of routes will be sought. To do this, using criteria of classic problems such as the traveling agent and conventional techniques such as neural networks will seek the most optimal route. In particular, this work aims to develop an application (software) capable of performing the optimization of a route of travel for a finite number of points, which will be visited and also present several features that must be considered, among which we have: *Priority of the points to visit, each point will have a priority that will be given by a quantitative value assigned by the client. Availability of vehicular routes, access from each point to another must be given depending on the vehicular access routes that exist. Availability working days for staff to complete the assigned route*. For the development and implementation of this project, the MATLAB platform has been used, where a Hopfield-type Neural Network, linear programming and graph theory has been built, obtaining an installable software for PC, which can be used in any computer with platform of Windows 7 or higher, in the same way an installable Add-in will be developed to work directly on an Excel spreadsheet of Office.

KEY WORDS:

- **HOPFIELD NEURAL NETWORK**
- **TSP TRAVELING SALESMAN PROBLEM**
- **KNAPSACK**

El comienzo es la parte más importante del trabajo.

Platón

CAPÍTULO

1

Introducción

El mantenimiento industrial constituye uno de los ejes más importantes del funcionamiento y operación de toda empresa o industria que mantenga como patrimonio y razón el desarrollo de equipos y maquinarias que generen valor agregado sobre una determinada materia prima, servicio o prestación hacia el cliente. Es esta premisa la motivación principal para el desarrollo de este proyecto, que busca implementar una solución técnica a uno de los mayores inconvenientes que existe en la administración de equipos y maquinarias, la planificación de las rutinas de mantenimiento, al tener que cumplir con una frecuencia determinada de atención a cada uno de los equipos y al estar estos distribuidos a lo largo de varias zonas geográficas, surgen las necesidades obvias de planificar intervenciones con grupos de trabajo que se deberán movilizar creando rutas programadas y circuitos de atenciones. A la luz de este requerimiento donde la optimización de recursos y de tiempo es primordial, pues de esta dependerá la eliminación de tiempos muertos, reprocesos, recorridos innecesarios, etc., que se magnifican en función del tamaño de la empresa y ubicación geográfica; surge la necesidad de construir un modelo que permita optimizar el trabajo de los grupos de mantenimiento considerando el recurso humano disponible, tiempos y ubicación geográfica de los puntos de mantenimiento.

Para este proyecto se ha tomado como referencia una entidad financiera con más de 300 puntos de atención a nivel nacional, la cual mantiene equipos y maquinarias vitales para su funcionamiento, entre los cuales podemos mencionar: grupos electrógenos, sistemas de climatización, sistemas de energía estabilizada, sistemas de seguridad y comunicación, entre otros. Dicha maquinaria debe ser mantenida periódicamente de manera óptima con el uso eficiente de recursos. Por lo citado en el siguiente proyecto se desarrollará e implementará un

software que permita optimizar la planificación del mantenimiento en puntos remotos, optimizando rutas calculadas mediante la implementación de una Red Neuronal Hopfield, Algoritmos de Programación Lineal y la Teoría de Grafos.

El trabajo se desarrollará de la siguiente manera: el presente capítulo a más de la introducción hablará sobre el estado del arte y los desarrollos que se han implementado en esta área hasta la actualidad. En el capítulo 2 se analizarán los conceptos fundamentales de las Redes Neuronales, Algoritmo de Knapsack, Teoría de Grafos y Problema del Agente Viajero, los cuales son la base para el desarrollo e implementación del software propuesto. En el capítulo 3 se formulará el problema que se pretende resolver con la implementación del software, las variables que intervendrán, y las restricciones que deberán incluirse para su funcionamiento óptimo. En el capítulo 4 se implementará el software sobre la plataforma de **Matlab**® mediante una Red Neuronal y aplicándola sobre la formulación matemática desarrollada para optimizar la ejecución de rutas de mantenimiento; así mismo se realizarán las pruebas y se analizarán los resultados. Para finalizar, en el capítulo 5 se establecerán las conclusiones y recomendaciones que el proyecto genere; como anexos se plantea el desarrollo del manual de funcionamiento y manejo del software que será el producto final de este proyecto.

1.1 Estado del Arte

La planificación y optimización logística son los puntos determinantes para el éxito de un plan de mantenimiento donde la diversificación de los puntos de atención constituye la característica común y habitual del día a día. Uno de los problemas más famosos de optimización que se ajusta a esta necesidad es el Problema del Agente Viajero (Travelling Salesman Problem TSP), término que tuvo su aparición en el año de 1832 en un libro alemán titulado "El viajante de Comercio: cómo debe ser y que debe hacer para conseguir comisiones y triunfar en su negocio. Por un viajante de comercio veterano" [Pérez De Vargas,2015]; en este libro se habla principalmente aspectos de la profesión y su último capítulo se define de manera explícita el problema del agente viajero. Posteriormente Merrill Flood es el principal responsable de ingresar el término TSP a la comunidad matemática, quien junto con A.W Tucker y Hassler Witney en la Universidad de Princeton en los años de 1931 y 1932 desarrollaron esta teoría e iniciaron investigaciones sobre la misma [Pérez De Vargas,2015]. En las décadas de los 50 y 60 este problema se volvió popular debido a muchas campañas publicitarias llevadas a cabo por "Procter and Gamble (P&G)" en 1962, la cual impulsaba un concurso que daba una recompensa de $10 000 dólares para quien ayudara a Toody y Multon a decidir una ruta más corta entre 33 ciudades de Estados Unidos conduciendo su famoso Car 54 [Pérez De Vargas,2015].

Muchas personas escribieron a la compañía P&G indicando que la solución del problema era imposible, pues no podrían comprobar todas las soluciones posibles. Ocho años antes en 1954 un equipo formado por George Dant Zinc, Ray Fullkerson y Selmer Jhonson encontraron una ruta óptima aplicada a 49 ciudades, gracias a la implementación de sus ideas en un ordenador se pudo trabajar en el desarrollo del problema del TSP [Pérez De Vargas,2015].

En el año de 1971 los investigadores de IBM Michael Held y Richard Karp resolvieron un problema con 64 nodos; en 1977 los matemáticos Martín Grotschel y Manfred Padberg construyeron una ruta optima a través de 120 ciudades alemanas, lo que le permitió a Padberg trabajar con el investigador de IBM Harlan Crowder encontrando una solución óptima para un ejemplo de 318 nodos que se encontraban dentro de una tarjeta de circuito impreso en 1987 [Pérez De Vargas,2015]. Grotscehl y Padberg continuaron su trabajo y resolvieron de forma consecutiva 3 rutas óptimas, una de 532 ciudades de Estados Unidos y otro de 666 localizaciones en el mundo y la última de 2 399 nodos en un problema de perforación. En el año de 1992 los matemáticos Vasek Chvátal y Willian J. Cook, con la colaboración de David Applegate y Robert Bisxby, consiguen resolver un problema de 3 038 nodos gracias a una larga red de ordenadores trabajando en paralelo, tras este record este equipo encuentra una nueva ruta óptima de 13 509 ciudades a través de los Estados Unidos en 1998. En el año 2004 el mismo equipo logra obtener una ruta óptima para 24 978 ciudades de Suecia y en el 2006 una ruta óptima 85 900 ciudades [Pérez De Vargas,2015].

Figura 1. Concurso Car 54 .
Fuente: ([Pérez De Vargas,2015])

Tabla 1
Hitos históricos en la resolución del TSP

Año	Autores	Nodos
1954	G. Dantzig, R. Fulkerson, S. Johnson	49
1971	M. Held, R.M. Karp	64
1975	P.M. Camerini, L. Fratta, F. Maffioli	67
1977	M. Grötschel	120
1980	H. Crowder and M. W. Padberg	318
1987	M. Padberg and G. Rinaldi	532
1987	M. Grötschel and O. Holland	666
1987	M. Padberg and G. Rinaldi	2 392
1994	D. Applegate, R. Bixby, V. Chvátal, W. Cook	7 397
1998	D. Applegate, R. Bixby, V. Chvátal, W. Cook	13 509
2001	D. Applegate, R. Bixby, V. Chvátal, W. Cook	15 112
2004	D. Applegate, R. Bixby, V. Chvátal, W. Cook	18 512
2004	D. Applegate, R. Bixby, V. Chvátal, W. Cook, K. Helsgaun	24 978
2004	W. Cook, Espinoza and Goycoolea	33 810
2006	D. Applegate, R. Bixby, V. Chvátal, W. Cook	85 900

Fuente: ([Espinoza,2006])

Figura 2. Progresos en la historia del TSP.
Fuente: ([Espinoza,2006])

Estos han sido los avances en la resolución del TSP. Matemáticos y otros profesionales trabajan cada día en mejorar los programas de cálculo para la optimización de rutas mediante la resolución del TSP [Espinoza,2006]. La optimización de las rutas de mantenimiento entonces consisten en hallar valores mínimos o máximos de una función definida, la toma de decisiones que implica ir construyendo la ruta más corta conlleva una complejidad, que dependiendo del número de nodos son inviables con métodos de resolución exactos, por lo que estos problemas de optimización frecuentemente se resuelven con métodos aproximados. Para el desarrollo de dichos métodos los ordenadores son la herramienta indispensable sin la cual sería imposible poder ejecutarlos, la búsqueda permanente del hombre de nuevas vías y métodos para mejorar las condiciones de estos cálculos lo han llevado a construir máquinas calculadoras, que lo ayuden a procesar la información y a tomar las decisiones inteligentemente, que no solo ejecuten algoritmos preestablecidos sino que tengan la capacidad de emular la inteligencia del ser humano [Ledesma,2000]. Existe actualmente una tendencia a establecer un nuevo campo en las ciencias de la computación para integrar diferentes métodos de resolución de problemas que no necesariamente utilizan algoritmos tradicionales. Estos métodos tienen su origen en la emulación de la inteligencia y el comportamiento de los sistemas biológicos. Así la inteligencia artificial es un intento por descubrir aspectos de la inteligencia humana que pueden ser emulados mediante máquinas por lo cual este desarrollo ha tenido un gran progreso en los últimos años y en varios campos como el reconocimiento de imágenes, la toma de decisiones, etc. Para este fin se dispone de un conjunto de metodologías como la lógica difusa, el razonamiento aproximado, los algoritmos genéticos, la teoría del caos, la teoría del aprendizaje y las redes neuronales, siendo estas últimas parte del desarrollo de este proyecto [Ohlsson,1992].

En 1936 Alan Turing fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación, más tarde los primeros teóricos que concibieron la computación neuronal fueron Warren McCulloch un neurofisiólogo y Walter Pitts un matemático que en 1943 crean una teoría acerca de la forma de trabajar de las neuronas, modelando una red neuronal mediante circuitos eléctricos (un cálculo lógico de la inminente idea de la actividad nerviosa boletín de Matemáticas Biofísica # 5) [Espinoza,2006]. En 1949 Donald Hebb fue el primero en explicar los procesos de aprendizaje desde un punto de vista sicológico, desarrollando la regla de como el aprendizaje ocurría, siendo esta hasta la actualidad el fundamento de las funciones de aprendizaje de las redes neuronales. Sus trabajos formaron las bases de la Teoría de las Redes Neuronales. En 1950 Karl Lashley encontró que la información en el cerebro no era almacenada de forma centralizada sino que era distribuida encima de él [Basogain,1998].

En 1956 se registra el nacimiento de la inteligencia artificial en el Congreso de Dart Mouth. Frank Rosenblatt inicia el desarrollo del Perceptrón en 1957, siendo esta la red neuronal más antigua que se la utiliza hasta la actualidad para reconocimiento de patrones. En 1959 Frank

lanza el libro denominado “Principios de Neurodinámica” en donde confirma que el aprendizaje del Perceptrón convergía hacia un estado finito bajo ciertas condiciones. En 1960 Bernard Widroff y Marcian Hoff desarrollaron el Modelo Adaline (Adaptative Linear Elements) que fue la primera red neuronal aplicada a un problema real cuyo objetivo fue eliminar los ecos en las líneas telefónicas, la cual ha sido utilizada por varias décadas. En 1961 Karl Steinbeck escribe el libro “Die Lernmatrix” que habla sobre la memoria asociativa de las redes neuronales con aplicaciones técnicas simples. Marvin Minsky y Seymour Papert en el año de 1969 probaron matemáticamente que el Perceptrón no era capaz de resolver problemas sencillos como el aprendizaje de una función no lineal demostrando su debilidad lo que casi produjo la muerte abrupta de las redes neuronales.

En 1974 Paul Werbos desarrolla la idea básica del algoritmo del aprendizaje de programación hacia atrás (Back Propagation) cuyo significado quedo definitivamente aclarado en 1985. Stephen Grossberg lanza la Teoría de Resonancia Adaptada (TRA), que es una arquitectura de red nueva para las ya conocidas hasta esa fecha, la misma que simula habilidades del cerebro como la memoria a largo y corto plazo, esto en el año de 1977. Jhon Hopfield en 1985 provoca el renacimiento de las redes neuronales con su libro “Computación Neuronal de Decisiones en Problemas de Optimización”. Mientras que David Rumelhart y G. Himpton en 1986 redescubrieron el algoritmo de aprendizaje de propagación hacia atrás (Back Propagation) [Basogain,1998]. A partir de este año se han desarrollado numerosos trabajos que se publican permanentemente sobre el adelanto de las redes neuronales, las aplicaciones nuevas que surgen sobre todo en el área de control permite a las empresas generar nuevos productos tanto en hardware como en software [Basogain,1998]. Otro problema de optimización que ha sido ampliamente estudiado a lo largo del tiempo, es el Problema de Knapsack haciéndose referencia a él ya en el año 1897. El nombre del Problema de la Mochila fue dado por el matemático Tobias Dantzig a principios del siglo XX. Desde esa fecha hasta la actualidad se han desarrollado varios algoritmos para la solución de este problema que también es considerado un problema de NP duro.[1]

Para el desarrollo de estos problemas de optimización la teoría de Grafos es muy importante y la base de su estudio y comprensión.

El trabajo de Leonhard Euler, en 1736, sobre el Problema de los Puentes de Konigsberg es considerado el primer resultado de la Teoría de Grafos. También se considera uno de los primeros resultados topológicos en geometría (que no depende de ninguna medida). Este ejemplo ilustra la profunda relación entre la Teoría de Grafos y la Topología [Rodríguez,2011].

En 1845 Gustav Kirchhoff publicó sus leyes de los circuitos para calcular el voltaje y la corriente en los circuitos eléctricos.

[1] Problemas de decisión que no tienen algoritmos polinomiales que los resuelva en forma exacta.

En 1852 Francis Guthrie planteo el problema de los cuatro colores que propone si es posible, utilizando solamente cuatro colores, colorear cualquier mapa de países de tal forma que dos países vecinos nunca tengan el mismo color. Este problema, que no fue resuelto hasta un siglo después por Kenneth Appel y Wolfgang Haken, puede ser considerado como el nacimiento de la teoría de grafos. Al tratar de resolverlo, los matemáticos definieron términos y conceptos teóricos fundamentales de los grafos [Rodríguez,2011].

La aplicación de una red neuronal en la solución del problema del Agente Viajero, es una alternativa que en 1993 Jean-Yves Potvin investigó utilizando para esto la red Hopfield y compara su efectividad con otros métodos heurísticos encontrando un buen desempeño en esta alternativa, entre el año de 1996 - 2000 Jacek Mandziuk continua un estudio similar con métodos estocásticos para la aplicación de la red neuronal Hopfield en la solución de los problemas del Agente Viajero obteniendo buenos resultados.

En el siguiente capítulo se profundiza en la teoría y descripción de las Redes Neuronales, el Problema del Agente Viajero, la Teoría de Grafos y el Algoritmo de Knapsack, para la búsqueda de un método de optimización en el cálculo de rutas eficientes para el mantenimiento de maquinarias y equipos bajo una distribución geográfica dispersa [Potvin,1993].

Hay, en verdad, dos cosas diferentes: saber y creer que se sabe. La ciencia consiste en saber; en creer que se sabe está la ignorancia.

Hipócrates

CAPÍTULO

2

Marco Teórico

En este capítulo se inicia el estudio y el análisis de lo que es una Red Neuronal, sus partes y tipos, se mostrarán aplicaciones y en general su funcionamiento. Como parte de la implementación de este proyecto se utilizará la Red Neuronal Hopfield, por lo que se profundizará en la implementación de este tipo de red, su programación y calibración con el objetivo de disminuir su error. De igual manera se explicará el algoritmo de optimización Knacksap y como se lo utilizará en la implementación de este desarrollo, ésto como paso previo al ingreso de la información a la Red Neuronal. Para finalizar se hablará sobre el problema del Agente Viajero o TSP, y como su formulación ha sido aplicada en la resolución del problema planteado.

2.1 Red Neuronal

El cerebro de un ser humano es el órgano que procesa la información que llega desde el cuerpo y del exterior, para actuar en consecuencia a partir de reacciones químicas e impulsos eléctricos. El cerebro no es eficiente para resolver ciertos problemas, como grandes multiplicaciones; sin embargo, hay otra clase de problemas que las personas pueden resolver de manera más eficiente. Cuantitativamente comparar al cerebro con un microprocesador de un computador establece una diferencia abismal en el tiempo de ejecución de las actividades por lo que siempre el cerebro estaría en desventaja, más al compararlos cualitativamente en la resolución de problemas de reconocimiento facial, por ejemplo, los sistemas neurobiológicos son generalmente más eficientes. De igual manera el cerebro gestiona la información de manera no lineal y en paralelo a diferencia de un computador, esto le permite

realizar operaciones simultáneamente y no secuencialmente como lo hace un microprocesador [De Castro,2002].

Es por esta razón que a lo largo de la historia el hombre siempre ha querido mejorar su calidad de vida y reducir los esfuerzos (físicos y mentales) a la hora de realizar tareas o resolver problemas. Las redes neuronales están basadas en la inteligencia artificial, y esta es un intento por descubrir y describir aspectos de la inteligencia humana que pueden ser simulados mediante máquinas, así pues, las redes neuronales constituyen otra manera de facilitar un trabajo al ser humano, en este caso, es el de emular ciertas características propias de los humanos, como la capacidad de aprender. Hay muchos problemas en la vida que no son capaces de ser resueltos aplicando un algoritmo secuencial, sino más bien se debe utilizar la experiencia; las redes neuronales intentan emular esta experiencia para la resolución de esos problemas, para ello usan la célula fundamental del sistema nervioso humano, la neurona [De Castro,2002].

Una neurona es una célula viva y, como tal, contiene los mismos elementos que forman parte de todas las células biológicas. Además, contiene elementos característicos que las diferencian. En general, una neurona consta de un cuerpo celular de aproximadamente 5 a 10 micras de diámetro esférico, del que salen una rama principal, el axón, y varias ramas más cortas, llamadas dendritas. A su vez, el axón puede producir ramas en torno a su punto de arranque, y con frecuencia se ramifica extensamente cerca de su extremo. Una de las características que diferencia a las neuronas del resto de las células vivas, es su capacidad de comunicarse. En términos generales, las dendritas y el cuerpo celular reciben señales de entrada; el cuerpo celular las combina e integra y emite señales de salida. El axón transporta esas señales a los terminales axónicos, que se encargan de distribuir información a un nuevo conjunto de neuronas. Por lo general, una neurona recibe información de miles de otras neuronas, y a su vez, envía información a miles de neuronas más. Se calcula que en el cerebro humano existen 10^{15} conexiones [Arredondo,2008].

Las redes neuronales no son más que otra forma de emular ciertas características propias de los humanos, como la capacidad de memorizar y de asociar hechos. Si se examinan con atención aquellos problemas que no pueden expresarse a través de un algoritmo, se observará que todos ellos tienen una característica en común: la experiencia. El hombre es capaz de resolver estas situaciones acudiendo a la experiencia acumulada. Así, parece claro que una forma de aproximarse al problema consista en la construcción de sistemas que sean capaces de reproducir esta característica humana. En definitiva, las redes neuronales no son más que un modelo artificial y simplificado del cerebro humano, que es el ejemplo más perfecto del que disponemos para un sistema que es capaz de adquirir conocimiento a través de la experiencia. Una red neuronal es “un nuevo sistema para el tratamiento de la información, cuya

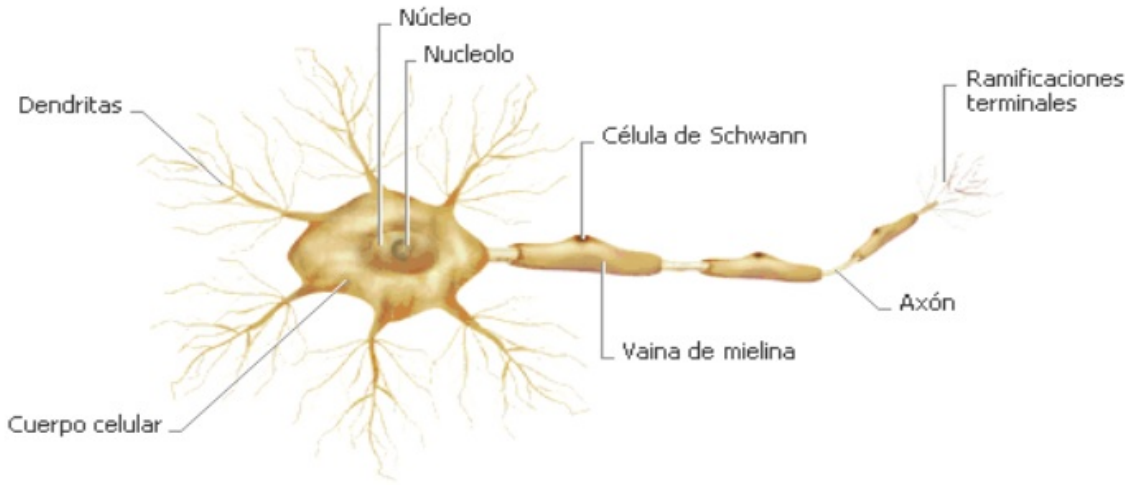


Figura 3. Neurona.
Fuente: ([Matich,2011])

unidad básica de procesamiento está inspirada en la célula fundamental del sistema nervioso humano: la neurona" [Basogain,1998].

Todos los procesos del cuerpo humano se relacionan en alguna u otra forma con la (in)actividad de estas neuronas. Las mismas son un componente relativamente simple del ser humano, pero cuando millares de ellas se conectan en forma conjunta se hacen muy poderosas.

También, es bien conocido que los humanos son capaces de aprender. Aprendizaje significa que aquellos problemas que inicialmente no pueden resolverse, pueden ser resueltos después de obtener más información acerca del problema. Por lo tanto, las Redes Neuronales consisten de unidades de procesamiento que intercambian datos o información. Se utilizan para reconocer patrones, incluyendo imágenes, manuscritos y secuencias de tiempo (por ejemplo: tendencias financieras). Tienen capacidad de aprender y mejorar su funcionamiento. Una primera clasificación de los modelos de redes neuronales podría ser, atendiendo a su similitud con la realidad biológica según lo siguiente: [Lasarte,2017]

1. El modelo de tipo biológico. Este comprende las redes que tratan de simular los sistemas neuronales biológicos, así como las funciones auditivas o algunas funciones básicas de la visión y,

2. El modelo dirigido a aplicación. Este modelo no tiene por qué guardar similitud con los sistemas biológicos. Su arquitectura está fuertemente ligada a las necesidades de las aplicaciones para la que es diseñada.

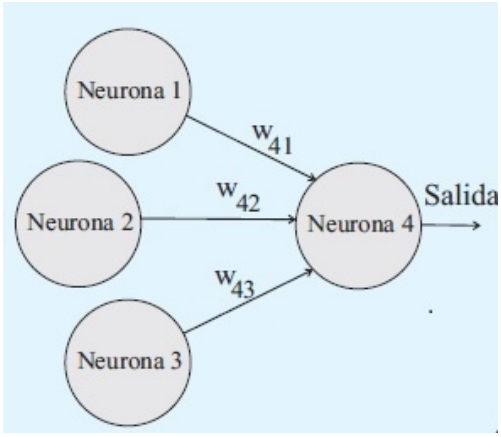


Figura 4. Una Red Neuronal.
Fuente: ([Basogain,1998])

2.1.1 **Red Neuronal Artificial RNA**

Se definen a las redes neuronales artificiales (RNA) como modelos matemáticos ó computacionales inspirados en sistemas biológicos, adaptados y simulados en computadoras convencionales, siendo una colección de procesadores paralelos conectados entre sí en forma de un grafo dirigido, organizados de tal modo que la estructura de la red sea adecuada para el problema que se esté considerando se puede ver cada unidad de procesamiento como un nodo, y las conexiones que hay entre unidades de procesamiento como arcos. Las RNA se conocen con diferentes nombres, como por ejemplo modelos conexionistas (connectionist models), procesamiento distribuido en paralelo (parallel distributed processing ó PDP), sistemas neuronales artificiales (artificial neural systems ó ANS), sistemas neuromórficos, etc. [Basogain,1998].

Las características más importantes de las RNA son:

- Aprendizaje Adaptativo
- Auto-organización
- Tolerancia a fallos
- Operación en tiempo real
- Fácil inserción dentro de la tecnología existente

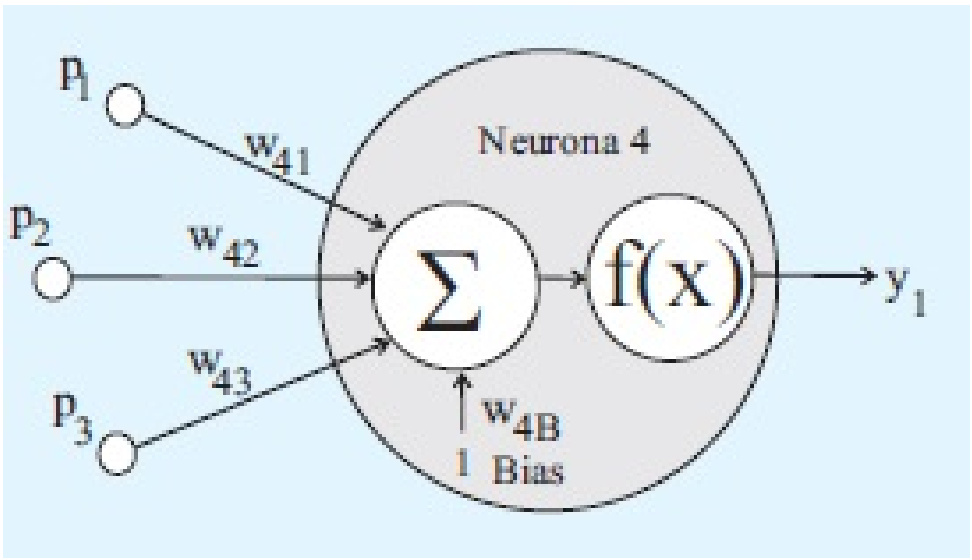


Figura 5. Modelo de una Red Neuronal.
Fuente: ([Basogain,1998])

2.1.1.1 **Aprendizaje Adaptativo**

Las redes neuronales son sistemas dinámicos autoadaptativos. Son adaptables debido a la capacidad de autoajuste de los elementos procesales (neuronas) que componen el sistema. Son dinámicos, pues son capaces de estar constantemente cambiando para adaptarse a las nuevas condiciones. En el proceso de aprendizaje, los enlaces ponderados de las neuronas se ajustan de manera que se obtengan ciertos resultados específicos. Una red neuronal no necesita un algoritmo para resolver un problema, ya que ella puede generar su propia distribución de pesos en los enlaces mediante el aprendizaje. También existen redes que continúan aprendiendo a lo largo de su vida, después de completado su período de entrenamiento. La función del diseñador es únicamente la obtención de la arquitectura apropiada. No es problema del diseñador el cómo la red aprenderá a discriminar. Sin embargo, sí es necesario que desarrolle un buen algoritmo de aprendizaje que le proporcione a la red la capacidad de discriminar, mediante un entrenamiento con patrones [Ohlsson,1992].

2.1.1.2 **Auto-organización**

Las redes neuronales emplean su capacidad de aprendizaje adaptativo para autoorganizar la información que reciben durante el aprendizaje y/o la operación. Mientras que el aprendizaje es la modificación de cada elemento procesal, la autoorganización consiste en la modificación de la red neuronal completa para llevar a cabo un objetivo específico. Cuando las redes neuronales se usan para reconocer ciertas clases de patrones, ellas autoorganizan la

información usada. Por ejemplo, la red llamada backpropagation, creará su propia representación característica, mediante la cual puede reconocer ciertos patrones [Lasarte,2017]. Esta autoorganización provoca la generalización: facultad de las redes neuronales de responder apropiadamente cuando se les presentan datos o situaciones a las que no había sido expuesta anteriormente. El sistema puede generalizar la entrada para obtener una respuesta. Esta característica es muy importante cuando se tiene que solucionar problemas en los cuales la información de entrada no es muy clara; además permite que el sistema dé una solución, incluso cuando la información de entrada está especificada de forma incompleta [Matich,2011].

2.1.1.3 **Tolerancia a fallos**

Las redes neuronales fueron los primeros métodos computacionales con la capacidad inherente de tolerancia a fallos. Comparados con los sistemas computacionales tradicionales, los cuales pierden su funcionalidad cuando sufren un pequeño error de memoria, en las redes neuronales, si se produce un fallo en un número no muy grande de neuronas y aunque el comportamiento del sistema se ve influenciado, no sufre una caída repentina [Lasarte,2017]. Hay dos aspectos distintos respecto a la tolerancia a fallos:

- Las redes pueden aprender a reconocer patrones con ruido, distorsionados o incompletos. Esta es una tolerancia a fallos respecto a los datos.
- Las redes pueden seguir realizando su función (con cierta degradación) aunque se destruya parte de la red.

La razón por la que las redes neuronales son tolerantes a los fallos es que tienen su información distribuida en las conexiones entre neuronas, existiendo cierto grado de redundancia en este tipo de almacenamiento. La mayoría de los ordenadores algorítmicos y sistemas de recuperación de datos almacenan cada pieza de información en un espacio único, localizado y direccionable. En cambio, las redes neuronales almacenan información no localizada. Por lo tanto, la mayoría de las interconexiones entre los nodos de la red tendrán sus valores en función de los estímulos recibidos, y se generará un patrón de salida que represente la información almacenada [Arredondo,2008].

2.1.1.4 **Operación en tiempo real**

Una de las mayores prioridades, casi en la totalidad de las áreas de aplicación, es la necesidad de realizar procesos con datos de forma muy rápida. Las redes neuronales se adaptan bien a esto debido a su implementación paralela. Para que la mayoría de las redes puedan operar en un entorno de tiempo real, la necesidad de cambio en los pesos de las conexiones o entrenamiento es mínimo [Arredondo,2008].

2.1.1.5 **Fácil inserción dentro de la tecnología existente**

Una red individual puede ser entrenada para desarrollar una única y bien definida tarea (tareas complejas, que hagan múltiples selecciones de patrones, requerirán sistemas de redes interconectadas). Con las herramientas computacionales existentes (no del tipo PC), una red puede ser rápidamente entrenada, comprobada, verificada y trasladada a una implementación hardware de bajo coste. Por lo tanto, no se presentan dificultades para la inserción de redes neuronales en aplicaciones específicas, por ejemplo de control, dentro de los sistemas existentes. De esta manera, las redes neuronales se pueden utilizar para mejorar sistemas en forma incremental y cada paso puede ser evaluado antes de acometer un desarrollo más amplio [Arredondo,2008].

Las Redes Neuronales Artificiales se han aplicado con éxito en numerosos campos, siendo los más destacados los siguientes:

Tabla 2
Aplicación de las RNA

Campos	**Ejemplos**
Aeroespacial	Pilotos Automáticos
Vehicular	Sistema de conducción autónoma Sistema de Inyección y Frenado
Defensa	Guiado de Misiles Rastreo de Objetivos
Electrónica	Diseños de Circuitos Integrados Control de Procesos
Entretenimiento	Efectos Especiales
Finanzas	Detección de Fraudes Predicción de Rentabilidad
Medicina	Ayuda al Diagnostico Desarrollo de Medicamentos
Telecomunicaciones	Compresión de datos Traducción de Idiomas
Transporte	Optimización de Rutas Optimización de Distribución
Robótica	Control de Trayectoria Sistema de Visión

Fuente: ([Arredondo,2008])

2.1.2 Elementos de la Red Neuronal Artificial RNA

Una Red Neuronal Artificial RNA está constituida por neuronas interconectadas y arregladas comúnmente en tres capas (esto último puede variar). Los datos ingresan por medio de la "capa de entrada", pasan a través de la "capa oculta" y salen por la "capa de salida". Cabe mencionar que la capa oculta puede estar constituida a su vez por varias capas.

Cada entrada llega a todas las neuronas de la primera capa, las salidas de las neuronas de la primera capa irán hacia todas las neuronas de la segunda capa. Esto se repite hasta llegar a la capa de salida donde se obtienen los valores finales. Por lo tanto, son siempre hacia adelante, no hay conexiones laterales ni hacia atrás [Lasarte,2017].

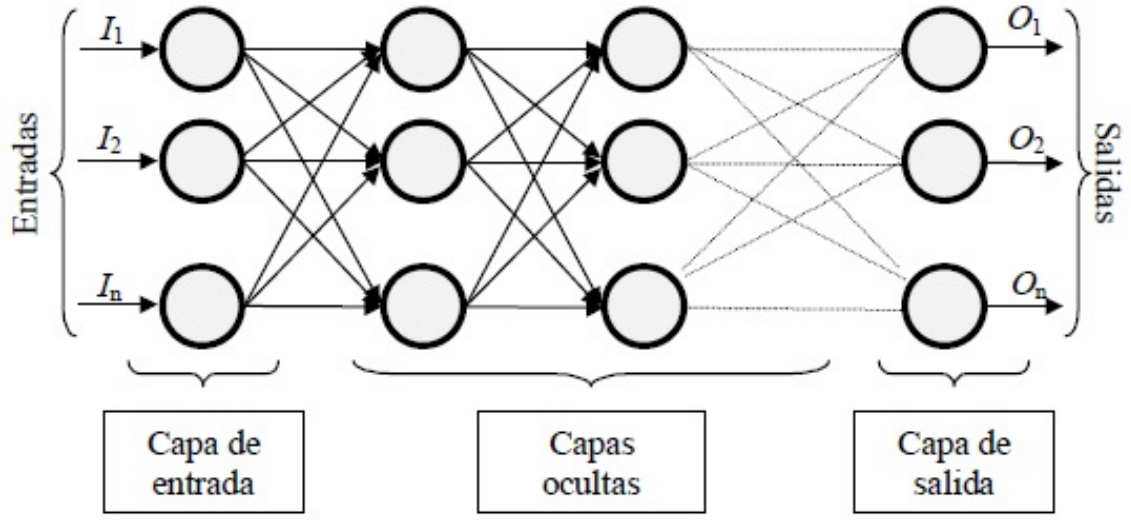


Figura 6. Elementos de una Red Neuronal.
Fuente: ([Lasarte,2017])

2.1.2.1 Función de entrada (input function)

Todos los valores de entrada hacia una neurona son considerados como una única entrada, denominada entrada global. Estas entradas $(In_1, In_2, ...)$ se combinan dentro de la entrada global, (gIn_i). Esto se logra a través de la función de entrada, la cual se calcula a partir del vector entrada. La función de entrada puede describirse como sigue:

$$input_i = (in_1 \bullet w_1) \star (in_2 \bullet w_2) \star ...(in_n \bullet w_n)$$

donde: $\star$ representa al operador apropiado (por ejemplo: máximo, sumatoria, producto, etc.), n_i representa al número de entradas a la neurona N y w_i representa su peso o valor, $\bullet$ representa que los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona. Por consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un

gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños. Algunas de las funciones de entrada más comúnmente utilizadas y conocidas son:

1. *Sumatoria de las entradas:* es la suma de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\sum_{j=1}^{n}(n_{ij}w_{ij}), \qquad conj = 1, 2, ..., n$$

2. *Producto de las entradas:* es el producto de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\prod_{j=1}^{n}(n_{ij}w_{ij}), \qquad conj = 1, 2, ..., n$$

3. *Máximo de las entradas:* solamente toma en consideración el valor de entrada más fuerte, previamente multiplicado por su peso correspondiente.

$$Max_j(n_{ij}w_{ij}), \qquad conj = 1, 2, ..., n$$

2.1.2.2 **Función de activación (activation function)**

Una neurona biológica puede estar activa (excitada) o inactiva (no excitada); es decir, que tiene un "estado de activación". Las neuronas artificiales también tienen diferentes estados de activación; algunas de ellas solamente dos, al igual que las biológicas, pero otras pueden tomar cualquier valor dentro de un conjunto determinado.

McCullogh y Pitts realizaron en 1943 un estudio biológico del cerebro obteniendo un modelo formal de neurona, con lo que introdujeron así el concepto de umbral: una neurona responde a un cierto estímulo siempre que éste sobrepase un cierto umbral de activación. Suele ser habitual añadir al conjunto de pesos de la neurona un parámetro adicional Θi, que se denomina umbral, el cual se acostumbra a restar a la entrada global Si los valores de entrada netos superan un cierto umbral la neurona devolverá una señal de activación positiva, por ejemplo "1", y si no lo superan devolverá una señal negativa, por ejemplo "0" o "-1".

La *función activación* calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral, Θi) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (–1 a 1). Esto es así, porque una neurona puede estar totalmente inactiva (0 o –1) o activa (1). La función activación, es una función de la entrada global *(gIni)* menos el umbral (Θi). Las funciones de activación se pueden ver en el resumen de la Figura 2.5 y las más comúnmente utilizadas se detallan a continuación [Campos,1998]:

1. *Función Escalon:*

$$f(x) = \begin{cases} 1, & \text{si } x \geq 0, \\ 0/-1, & \text{si } x < 0. \end{cases}$$

2. *Función lineal:*

$$f(x) = \begin{cases} -1, & \text{si } x \leq -1/a, \\ a \times x, & \text{si } -1/a < x < 1/a. \\ 1, & \text{si } x \geq 1/a. \end{cases}$$

3. *Función Sigmoidea:*

$$f(x) = \frac{1}{1 + e^{-g*x}}, o$$
$$f(x) = tgh(g * x), con :$$
$$x = (\textit{(gIni)} - \Theta i)$$

4. *Función tangente hiperbólica:*

$$f(x) = \frac{e^{g*x} - e^{-g*x}}{e^{g*x} + e^{-g*x}}, con :$$
$$x = (\textit{(gIni)} - \Theta_i)$$

2.1.2.3 **Función de salida (output function)**

El último componente que una neurona necesita es la *función de salida*. El valor resultante de esta función es la salida de la neurona i *(out_i)*; por ende, la función de salida determina que valor se transfiere a las neuronas vinculadas. Si la función de activación está por debajo de un umbral determinado, ninguna salida se pasa a la neurona subsiguiente. Normalmente, no cualquier valor es permitido como una entrada para una neurona; por lo tanto, los valores de salida están comprendidos en el rango [0, 1] o [-1, 1]. También pueden ser binarios 0, 1 o -1, 1. Dos de las funciones de salida más comunes son:

- *Ninguna:* este es el tipo de función más sencillo, tal que la salida es la misma que la entrada. Es también llamada *función identidad*.
- $Binaria : \begin{cases} 1, & \text{si } act_i \geq \xi_i, \\ 0, & \text{de lo contrario} \end{cases}$, donde ξ_i es el umbral.

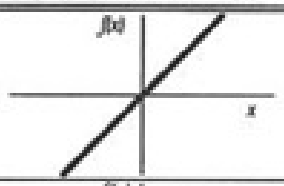
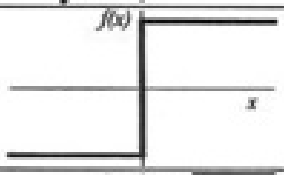
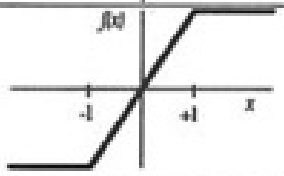
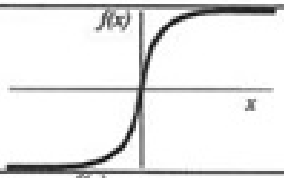
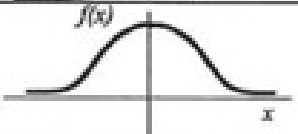
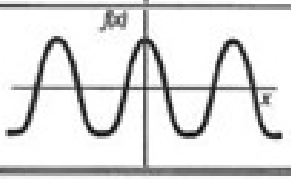
	Función	Rango	Gráfica
Identidad	$y = x$	$[-\infty, +\infty]$	
Escalón	$y = sign(x)$ $y = H(x)$	$\{-1, +1\}$ $\{0, +1\}$	
Lineal a tramos	$y = \begin{cases} -1, & si\ x < -l \\ x, & si\ +l \le x \le -l \\ +1, & si\ x > +l \end{cases}$	$[-1, +1]$	
Sigmoidea	$y = \frac{1}{1+e^{-x}}$ $y = tgh(x)$	$[0, +1]$ $[-1, +1]$	
Gaussiana	$y = Ae^{-Bx^2}$	$[0,+1]$	
Sinusoidal	$y = A\,\mathrm{sen}(\omega x + \varphi)$	$[-1,+1]$	

Figura 7. Funciones de Activación Habituales.
Fuente: ([Campos,1998])

2.1.3 Clasificación de las Redes Neuronales Artificiales RNA

Las RNA pueden clasificarse segun:

- Su arquitectura.
- Su aprendizaje.

2.1.3.1 Según la arquitectura

La arquitectura de una red consiste en la disposición y conexionado de las neuronas. Se puede distinguir en una red, el número de capas, el tipo de las capas, que pueden ser ocultas o visibles, de entrada o de salida y la direccionalidad de las conexiones de las neuronas [Arredondo,2008].

- Redes Monocapa.
- Redes Multicapa.

Redes Monocapa

En las redes monocapa, se establecen conexiones entre las neuronas que pertenecen a la única capa que constituye la red. Las redes monocapas se utilizan generalmente en tareas relacionadas con lo que se conoce como autoasociación (regenerar información de entrada que se presenta a la red de forma incompleta o distorsionada).

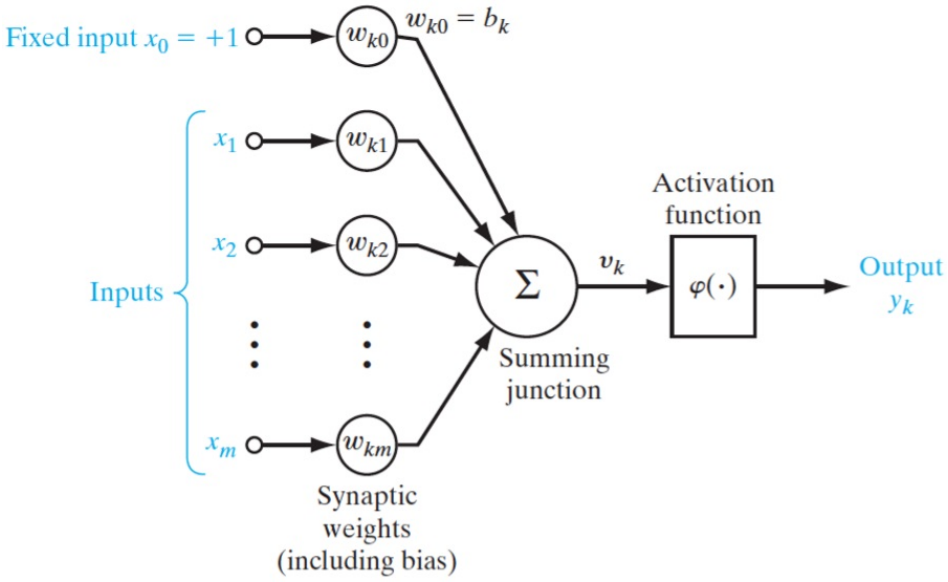


Figura 8. Red Neuronal Monocapa.
Fuente: ([Arredondo,2008])

Redes Multicapa

Las redes multicapas son aquellas que disponen de un conjunto de neuronas agrupadas en varios (2, 3, etc.) niveles o capas. En estos casos, una forma para distinguir la capa a la que pertenece una neurona, consistiría en fijarse en el origen de las señales que recibe a la entrada y el destino de la señal de salida. Normalmente, todas las neuronas de una capa reciben señales de entrada desde otra capa anterior (la cual está más cerca a la entrada de la red), y envían señales de salida a una capa posterior (que está más cerca a la salida de la red). A estas conexiones se las denomina *conexiones hacia adelante o feedforward* [Arredondo,2008].

Sin embargo, en un gran número de estas redes también existe la posibilidad de conectar la salida de las neuronas de capas posteriores a la entrada de capas anteriores; a estas conexiones se las denomina *conexiones hacia atrás o feedback*. Estas dos posibilidades permiten distinguir entre dos tipos de redes con múltiples capas: las redes con conexiones hacia adelante o redes feedforward, y las redes que disponen de conexiones tanto hacia adelante como hacia atrás o redes *feedforward/feedback*.

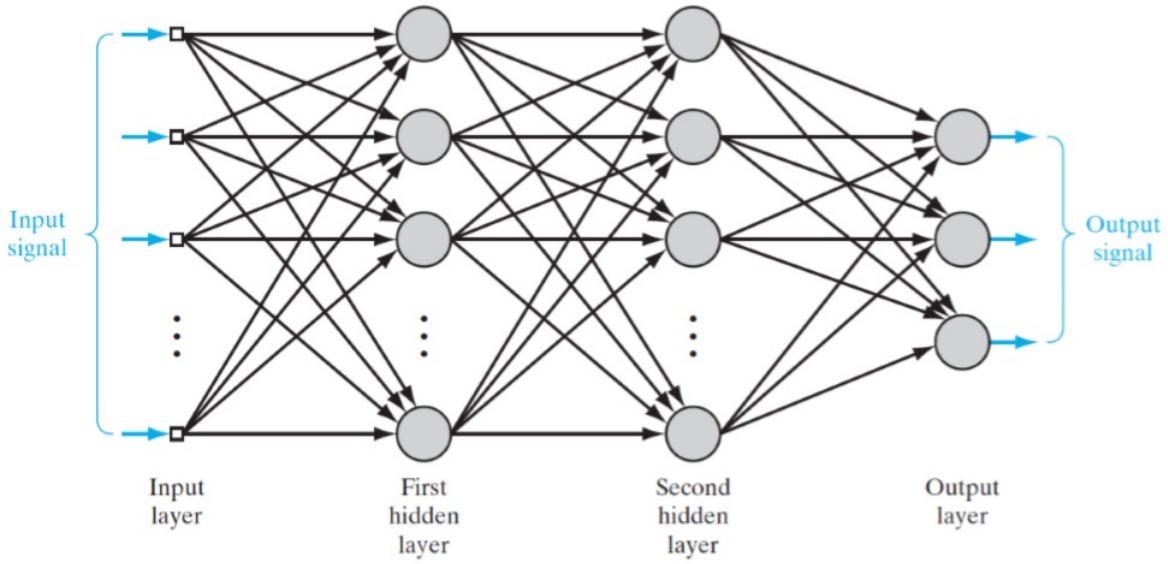


Figura 9. Red Neuronal Multicapa.
Fuente: ([Arredondo,2008])

2.1.3.2 **Según el aprendizaje**

El objetivo del entrenamiento de una RNA es conseguir que una aplicación determinada, para un conjunto de entradas, produzca el conjunto de salidas deseadas o mínimamente consistentes. El proceso de entrenamiento consiste en la aplicación secuencial de diferentes conjuntos o vectores de entrada para que se ajusten los pesos de las interconexiones según un procedimiento predeterminado. Durante la sesión de entrenamiento los pesos convergen gradualmente hacia los valores que hacen que cada entrada produzca el vector de salida deseado

[Ohlsson,1992]. Los algoritmos de entrenamiento o los procedimientos de ajuste de los valores de las conexiones de las RNA se pueden clasificar en dos grupos:

- Aprendizaje Supervisado y,
- Aprendizaje No Supervisado.

2.1.3.3 **Aprendizaje Supervisado**

El aprendizaje supervisado se caracteriza porque el proceso de aprendizaje se realiza mediante un entrenamiento controlado por un agente externo (supervisor, maestro) que determina la respuesta que debería generar la red a partir de una entrada determinada. El supervisor controla la salida de la red y en caso de que ésta no coincida con la deseada, se procederá a modificar los pesos de las conexiones, con el fin de conseguir que la salida obtenida se aproxime a la deseada [Ohlsson,1992]. En este tipo de aprendizaje se suelen considerar, a su vez, tres formas de llevarlo a cabo, que dan lugar a los siguientes aprendizajes supervisados:

1. Aprendizaje por corrección de error,
2. Aprendizaje por refuerzo y,
3. Aprendizaje estocástico.

2.1.3.4 **Aprendizaje No Supervisado**

Las redes con aprendizaje no supervisado (también conocido como autosupervisado) no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas. La red no recibe ninguna información por parte del entorno que le indique si la salida generada en respuesta a una determinada entrada es o no correcta. Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada. Existen varias posibilidades en cuanto a la interpretación de la salida de estas redes, que dependen de su estructura y del algoritmo de aprendizaje empleado.

En algunos casos, la salida representa el grado de familiaridad o similitud entre la información que se le está presentando en la entrada y las informaciones que se le han mostrado hasta entonces (en el pasado). En otro caso, podría realizar una clusterización (clustering) o establecimiento de categorías, indicando la red a la salida a qué categoría pertenece la información presentada a la entrada, siendo la propia red quien debe encontrar las categorías apropiadas a partir de las correlaciones entre las informaciones presentadas [Arredondo,2008].

En cuanto a los algoritmos de aprendizaje no supervisado, en general se suelen considerar dos tipos, que dan lugar a los siguientes aprendizajes:

1. Aprendizaje hebbiano y,

2. Aprendizaje competitivo y comparativo.

La selección de una red se realiza en función de las características del problema a resolver. La mayoría de éstos se pueden clasificar en aplicaciones de Predicción, Clasificación, Asociación, Conceptualización, Filtrado y Optimización.

Tabla 3
Evolución de las Redes Neuronales Artificiales

Red	Creador	Año	Aplicación	Caracteristicas
Adaline y Madaline	Bernard Widrow	1960	Predicción	Técnicas de Adaptación para el Reconocimiento de Patrones
Adaptive Resonance Theory Networks (ART)	Carpenter, Grossberg	1960-86	Conceptualización	Reconocimiento de Patrones y Modelo del Sistema Neuronal. Concepto de Resonancia Adaptativa.
Back-Propagation	Rumelhart y Parker	1985	Clasificación	Solución a las limitaciones de su red predecesora el Perceptron
Bi-Directional Associative Memory (BAM) Networks	Bart Kosko	1987	Asociación	Inspirada en la red ART
The Boltzmann Machine	Ackley, Hinton y Sejnowski	1985	Asociación	Similar a la red Hopfield
Brain-State-in a Box	James Anderson	1970-86	Asociación	Red Asociativa Lineal
Cascade-Correlation-Networks	Fahhman y Lebiere	1990	Asociación	Adición de nuevas capas ocultas en cascada
Counter-Propagation	Hecht-Nielsen	1987	Clasificación	Clasificación Adaptativa de Patrones
Delta-Bar-Delta (DBD) Networks	Jacobb	1988	Clasificación	Métodos Heurísticos para Acelerar la Convergencia
Digital Neural Network Architecture (DNNA) Networks	Neural Semiconductor Inc	1990	Predicción	Implementación Hardware de la función Sigmoid

Sigue en la página siguiente.

Red	Creador	Año	Aplicación	Caracteristicas
Directed Random Search (DRS) Networks	Maytas y Solis	1965-81	Clasificación	Técnica de valores Random en el mecanismo de Ajuste de Pesos
Functional-link Networks (FLN)	Pao	1989	Clasificación	Versión mejorada de la red Backpropagation
Hamming Networks	Lippman	1987	Asociación	Clasificador de vectores binarios utilizando la Distancia Hamming
Hopfield Networks	Hopfield	1982	Optimización	Concepto de la red en términos de energía
Learning Vector Quantization (LVQ) Networks	Kohonen	1988	Clasificación	Red Clasificadora
Perceptron Networks	Rosenblatt	1950	Predicción	Primer modelo de sistema Neuronal Artificial
Probabilistic Neural Network (PNN)	Spetcht	1988	Asociación	Clasificación de Patrones utilizando métodos estadísticos
Recirculation Networks	Hinton y McClelland	1988	Filtrado	Alternativa a la red Backpropagation
Self-Organizing Maps (SOM)	Kohonen	1979-82	Conceptualización	Aprendizaje sin supervisión
Spatio-Temporal-Pattern Recognition (SPR)	Grossberg	1960-70	Asociación	Red clasificadora Invariante en el espacio y tiempo

Fuente: ([Ohlsson,1992])

2.2 Red Neuronal HOPFIELD

La red Hopfield fue propuesta en 1982 por el físico John Hopfield. Se basa en:

- Permite recuperar patrones a partir de una información incompleta, con la implementación de la denominada memoria asociativa, que no es más que recuperar información a partir de conocimiento parcial y,
- Todas las neuronas están conectadas con todas las demás a excepción de consigo misma.

Siendo una red monocapa de aprendizaje no supervisado de tipo hebbiano en la que las neuronas tienen valores binarios y están conectadas entre sí, en esta red la salida de cada neurona se calcula y retroalimenta como entrada hasta llegar a un punto de estabilidad donde ya no existe cambio [Vasantha ,2009].

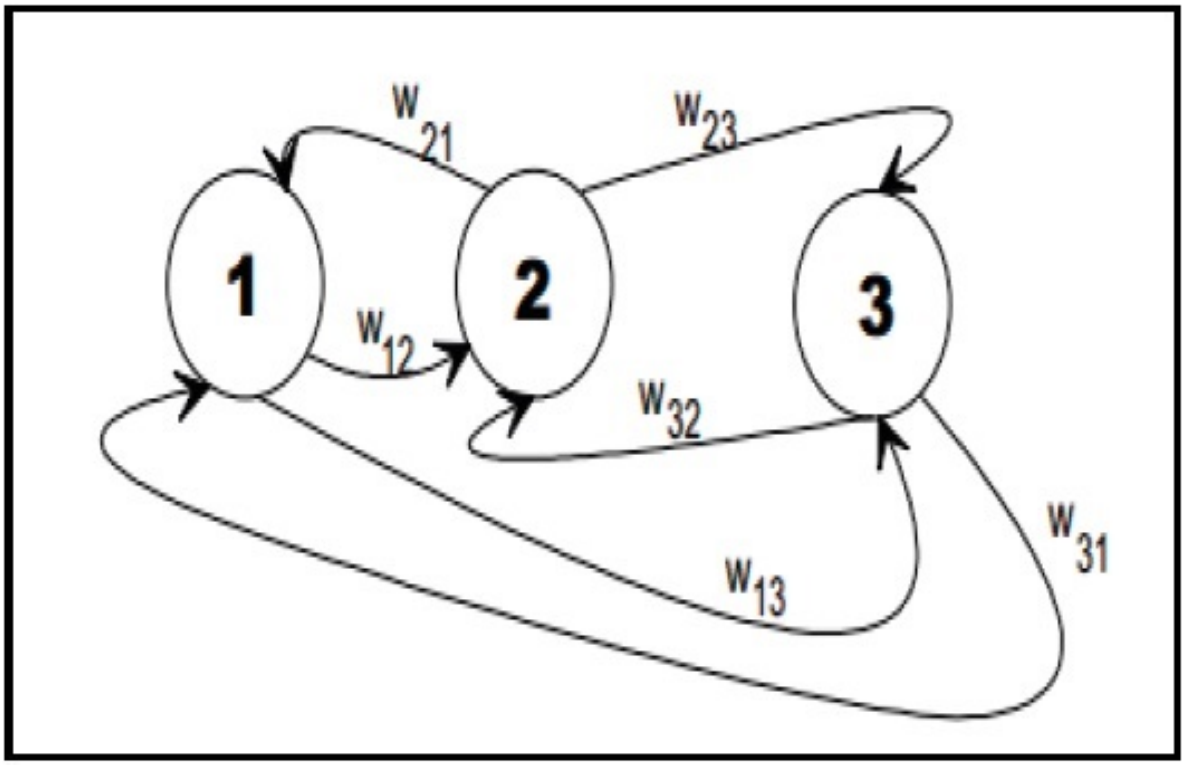


Figura 10. Red Neuronal Hopfield.
Fuente: ([Vasantha ,2009])

La finalidad de esta red es almacenar o aprender un conjunto de patrones, de tal manera que al aparecer un nuevo patrón se lo reconozca y se lo asigne a uno de los ya aprendidos, en virtud de ser una red autoasociativa.

2.2.1 Funcionamiento

El funcionamiento de una red Hopfield tiene algunas etapas, las que se detalla a continuación:

2.2.1.1 Fase de almacenamiento

El modelo Hopfield está basado en una red monocapa en la cual las neuronas están conectadas entre sí menos consigo mismas, por lo tanto la estructura del aprendizaje o almacenamiento es una matriz de pesos cuadrada, simétrica y donde la diagonal principal es 0.

2.2.1.2 Fase de recuperación

Al finalizar la fase de almacenamiento existe la fase de recuperación en la que se comprueba la eficacia del aprendizaje o almacenamiento. En esta fase la red debe recuperar el patrón almacenado más próximo al patrón de prueba. Se calculan los estados de la red y se espera hasta que la red alcance la estabilidad, de no producirse la estabilidad se debe reiniciar la fase de almacenamiento o aprendizaje reduciendo el número de patrones [Kleinberg ,2005].

2.2.1.3 **Función de energía o de error**

Una de las principales aportaciones de John Hopfield en su visión del estado de una red fue la función de energía que en cada estado de la red se describe durante el proceso de convergencia, es decir la energía global de la red se hace cada vez más pequeña hasta que finalmente alcance el estado estable [Kleinberg ,2005].

La Red Hopfield consta de un número de neuronas simétrica e íntegramente conectadas, como ya se mencionó anteriormente. Esto significa que si existe una conexión desde la neurona N_i a la neurona N_j, también existe la conexión desde N_j a N_i;ambas exhibendo el mismo peso ($w_{ij} = w_{ji}$). De igual manera se establece que la conexión de una neurona con sí misma no está permitida. El conjunto permitido de valores de entrada y salida es $\{0,1\}$ (o en alguna oportunidad $\{$-1, 1$\}$); o sea, es un conjunto binario. De esta manera todas las neuronas en una Red Hopfield son binarias, tomando solamente uno de los dos estados posibles: activo (1) o inactivo (-1 o 0). En una Red Hopfield los pesos se pueden calcular y se mantienen fijos durante el aprendizaje de los patrones. Solamente cambia el estado de las neuronas. Para calcular el peso de una conexión cualquiera, w_{ij} (y por simetría para la conexión w_{ij}i), en una Red Hopfield se utiliza la siguiente ecuación [Wolfe,1999]:

$$W_{ij} = \sum_{q=1}^{Q} (2 * e_{qi} - 1) * (2 * e_{qj} - 1), \quad i \neq j$$

Donde *Q,* es el número de patrones y e_{qi} es la entrada a la neurona N_i.

Con respecto al número de ceros y unos, el umbral de cada neurona puede utilizarse para regular esto, distinguiéndose así dos casos posibles:

1. Si hay más 0s que 1s el umbral tiene que disminuirse, porque que las neuronas tienen una probabilidad más alta para hacerse inactivas que para hacerse activas.

2. Si hay más 1s que 0s el umbral tiene que incrementarse, porque las neuronas tienen una probabilidad más alta para hacerse activas que para hacerse inactivas.

La premisa fundamental de este método es que el problema de optimización puede formularse como una función energética. Por lo tanto hallar el óptimo global implica encontrar el mínimo de dicha función energética. La misma tiene un aspecto genérico de esta forma:

$$E = -0{,}5 * \sum_{i} \sum_{j} (W_{ij} * out_i * outj) + \sum_{i} (\Theta_i * outi)$$

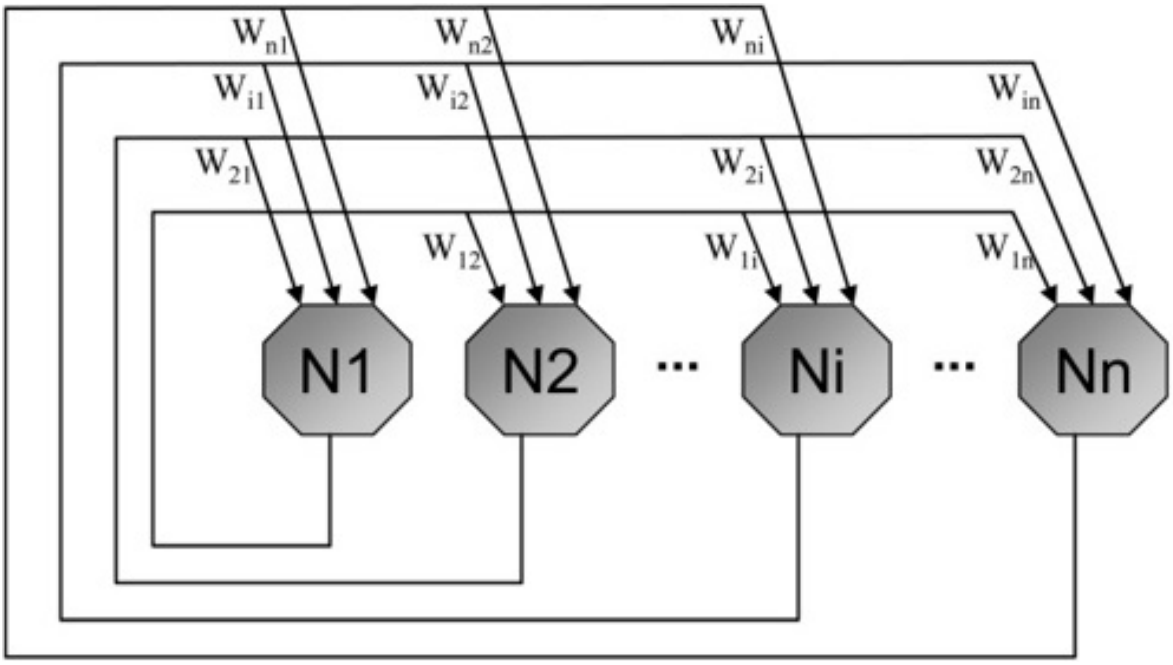


Figura 11. Estructura Hopfield.
Fuente: ([Wolfe,1999])

2.2.1.4 **Aplicaciones de la red Hopfield**

La red Hopfield puede ser aplicada tanto en problemas de asociación de memoria como en los problemas de optimización los cuales quedan exitosamente resueltos cuando la red alcanza un estado estable en un mínimo de energía.

2.2.1.5 **Ventajas y limitaciones de la red Hopfield**

Las redes Hopfield presentan una rápida capacidad computacional la cual se debe a la naturaleza altamente paralela del proceso de convergencia. Las limitaciones que podrían presentarse es el problema de la estabilización de la energía cuando se almacenan demasiadas memorias lo que hace que una red de N neuronas pueda tener muchos más estados que 2N.

2.3 **Algoritmo de KNAPSACK**

El Algoritmo de Knapsack o comúnmente conocido como el Problema de la Mochila define el proceso para la optimización, por un lado se tiene la mochila en la que se van a introducir los objetos, y por otro lado se tiene la lista de N objetos, teniendo los siguientes datos [Pisinger,2005]:

- W= capacidad máxima de la mochila.
- v_i= valor del i-ésimo objeto.
- w_i= peso del i-ésimo objeto.

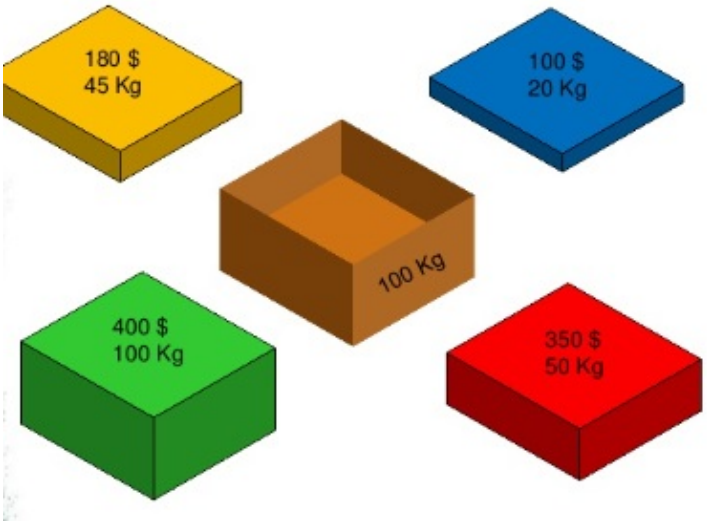


Figura 12. Problema de Knapsack.
Fuente: ([Pisinger,2005])

Tanto la capacidad como los valores y los pesos serán siempre enteros positivos, el objetivo entonces consiste seleccionar un subconjunto de N tal que la ganancia total de esos objetos dada por los valores que suman sean maximizados, y el total de los pesos de objetos seleccionados no exceda el valor límite de la capacidad de la mochila, matemáticamente podemos expresar este problema de la siguiente manera:

$$maximizar \sum_{i=1}^{n} v_i x_i$$

$$sujeto\ a:$$

$$\sum_{i=1}^{n} v_i x_i \leq W$$

$$x_i = 0\ o\ 1, \qquad i = 1, ..., n$$

La solución al problema viene dada por el vector de variables binarias $X(x_1, x_2, ..., x_n)$. Estas variables binarias tienen el siguiente significado:

$$X_i = \begin{cases} 1, & \text{si el objeto } i \text{ es seleccionado} \\ 0, & \text{si el objeto } i \text{ no es seleccionado} \end{cases}$$

las variantes del problema de la Mochila son:

2.3.1 El problema de la mochila binaria

Está versión del problema de la mochila es conocida como el problema de la mochila binaria, también llamada mochila 0/1. Este nombre viene dado por el hecho de que para cada ítem disponible solo hay dos opciones, elegir si cogerlo o no, siendo esta una decisión binaria

[Bartholdi,2008]. Las principales razones porque esta versión del problema de la mochila es tan importante y es de las más estudiadas son las siguientes:

1. Puede ser vista como el problema de programación lineal más simple
2. Aparece como subproblema en muchos problemas más complejos y
3. Es ampliamente usada en situaciones prácticas

2.3.2 El problema de la mochila acotado

El problema de la mochila se puede generalizar eliminando la restricción de que solo hay una unidad de cada ítem, por lo que el mismo ítem se puede escoger varias veces. Para ello se tiene que conocer otra característica de cada ítem. b_i para i= 1,...,n es la cota superior para el ítem i, el número máximo de copias de ese ítem que se pueden meter en la mochila.

En está instancia del problema el vector $X(x_1, x_2, ..., x_n)$ ya no es un vector de variables binarias, sino un vector de variables enteras, donde x_i representa el número de copias del i-ésimo objeto introducidas en la mochila. La formulación matemática de esta versión del problema es la siguiente [Bartholdi,2008]:

$$maximizar \sum_{i=1}^{n} v_i x_i$$
$$sujeto\ a:$$
$$\sum_{i=1}^{n} v_i x_i \leq W$$
$$0 \leq x_i \leq b_i, \qquad i = 1, ..., n$$

2.3.3 El problema de la mochila no acotado

El problema puede ser generalizado aún más en base a la instancia anterior, en este caso eliminando la restricción al número de copias de cada ítem que pueden ser insertadas en la mochila, pudiendo introducir un número ilimitado de copias del mismo ítem en la mochila. En está instancia las variable b_i para i= 1,...,n desaparecen. El significado del vector solución no cambia respecto al problema de la mochila acotado, x_i representa el número de copias del i-ésimo objeto introducidas en la mochila. El problema se define de la siguiente forma:

$$maximizar \sum_{i=1}^{n} v_i x_i$$
$$sujeto\ a:$$
$$\sum_{i=1}^{n} v_i x_i \leq W$$
$$0 \leq x_i \leq \infty, \qquad i = 1, ..., n$$

2.3.4 Formulación Lineal

Una de las técnicas matemáticas que se puede utilizar para la resolución de este problema es la programación lineal. Estableciendo las siguientes variables:

- c, capacidad de la mochila.
- v_i, como el beneficio unitario obtenido por ingresar el producto i en la mochila.
- w_i= peso del producto i.
- n como la cantidad de productos.
- c, p_i y w_i como valores enteros y positivos.

Se tiene entonces:

$$maximizar\ z = \sum_{i=1}^{n} v_i x_i$$

$$sujeto\ a:$$

$$\sum_{i=1}^{n} w_i x_i \leq c$$

$$0 \leq x_i \leq \infty, \qquad i = 1, ..., n$$

No obstante, el problema que tiene esta técnica es que no siempre se puede resolver debido a su complejidad matemática. En esas ocasiones, es necesario recurrir a heurísticas, o algoritmos de aproximación.

2.4 Teoría de Grafos

En matemáticas y en ciencias de la computación, la teoría de grafos (también llamada teoría de las gráficas) estudia las propiedades de los grafos (también llamadas gráficas). Un grafo es un conjunto, no vacío, de objetos llamados vértices (o nodos) y una selección de pares de vértices, llamados aristas (edges en inglés) que pueden ser orientados o no. Típicamente, un grafo se representa mediante una serie de puntos (los vértices) conectados por líneas (las aristas) [Paredes,2011].

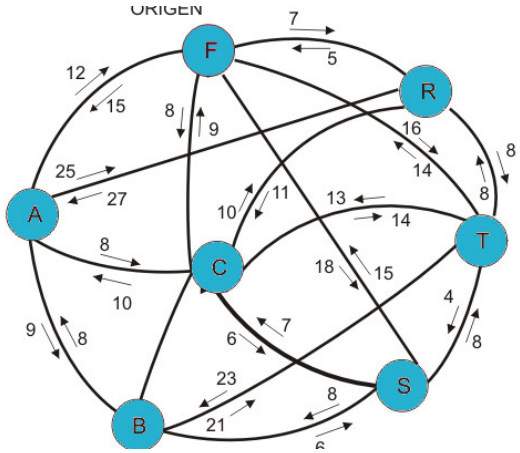


Figura 13. Teoría de Grafos.
Fuente: ([Paredes,2011])

2.4.1 Grafos

Un grafo es un par $G = (V, E)$, donde V es el conjunto de vértices del grafo, siendo su cardinal $|V| = n$ entero positivo, y $E \subset VxV$ son los arcos o las aristas del grafo $(e = \{i, j\} = \{j, i\}$ es una arista, por lo que no existe orientación).

$$V = \{1, 2, 3, 4\}$$

$$U = \{(1, 2), (2, 4), (4, 3), (3, 2)\}$$

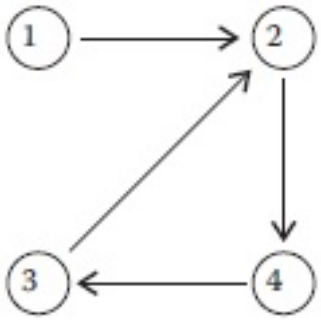


Figura 14. Grafo Dirigido.
Fuente: ([Paredes,2011])

2.4.2 Tipos y Características de los Grafos

Según el sentido y posición de las aristas un Grafo puede ser [Novo,2004]:

- ***Orientado o Dirigido***, es aquel en el que los arcos son pares ordenados; el arco o la arista (i, j) empieza en el nodo x y termina en el nodo y.

- ***No orientado o no dirigido***,es un grafo en el que (i, j) y (j, i) representan el mismo arco o arista.
- ***Multigrafo***, es un grafo con varias aristas entre dos vértices.
- ***Pseudografo***, tiene aristas cuyos extremos coinciden (origen y fin en el mismo vértice), tales aristas se denominan lazos.
- ***Digrafos***, cuando E está formado por pares ordenados (es decir, importa cual de los 2 vértices del par se coloca primero) de elementos de V, sus elementos son llamados arcos y se habla de grafo dirigido o dígrafo:

$$D = (V, E)$$

.

- ***Isomorfismo***, dos grafos son isomorfos, si cada par de vértices adyacentes, corresponde con un par de vértices adyacentes del otro.
- ***Subgrafo***, un subgrafo de un grafo G es un grafo cuyos conjuntos de vértices y aristas son subconjuntos de los de G.
- ***Grafos simples***, un grafo es simple si una arista cualquiera es la única que une dos vértices específicos.
- ***Grafos completos***, un grafo es completo si existen aristas uniendo todos los pares posibles de vértices. Es decir, todo par de vértices (i, j) debe tener una arista que los une.
- ***Grafos bipartitos***, un grafo G es bipartito si puede expresarse como $G = \{V_1 \bigcup V_2, A\}$(es decir, sus vértices son la unión de dos grupos de vértices).
- ***Árboles***, un grafo que no tiene ciclos y que conecta a todos los puntos.

2.5 Problema del Agente Viajero TSP

El Problema del Agente Viajero, conocido también con sus siglas inglesas TSP (Traveling Salesman Problem), trata de buscar la menor ruta posible que recorre una serie de ciudades (o nodos) de las que se conoce la distancia entre cada par de ellas. Dicha ruta debe visitar cada ciudad exactamente una vez y regresar a la ciudad origen. De una manera más informal: "Si un viajante parte de una ciudad, conociendo las distancias entre las distintas ciudades por las que ha de pasar ¿cuál es la ruta óptima que debe elegir para visitar todas las ciudades una

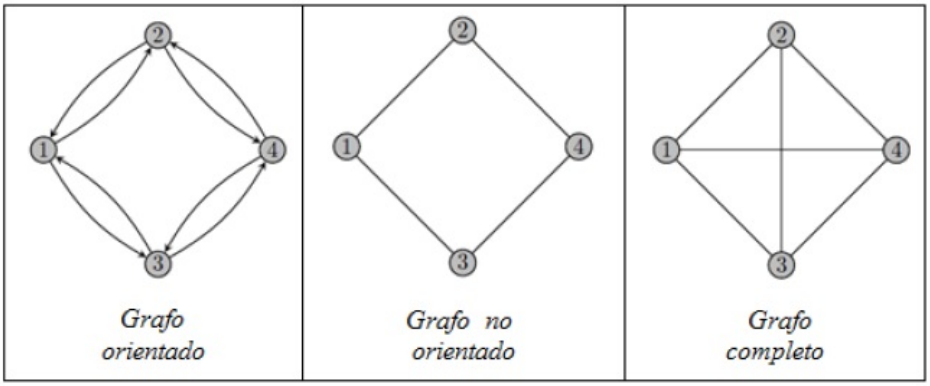


Figura 15. Tipos de Grafos.
Fuente: ([Novo,2004])

sola vez y volver a la ciudad de partida [Quirós,2013]. El problema del viajante es uno de los problemas más famosos y estudiados dentro de la optimización combinatoria1. A pesar de la aparente sencillez de su enunciado, el TSP es uno de los más complejos de resolver, lo que ha suscitado durante los últimos 60 años gran interés entre investigadores relacionados con las matemáticas, física, o biología entre otras.

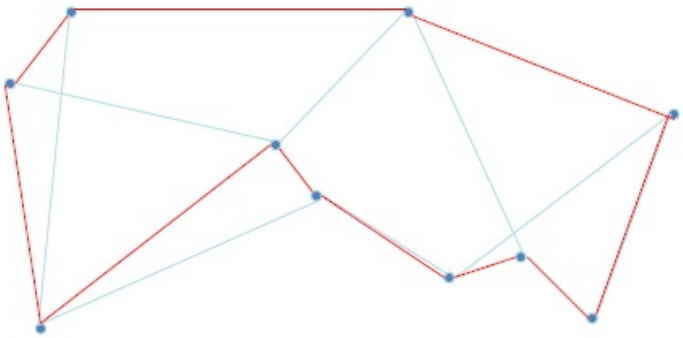


Figura 16. Ruta Agente Viajero.
Fuente: ([Quirós,2013])

El TSP es un problema NP-duro (nondeterministic polynomial time). Igual que ocurre con otros problemas en el ámbito de la Organización Industrial, el TSP se caracteriza por ser fácilmente descriptible pero difícil de resolver, por lo que ha despertado un gran interés a lo largo de la historia no sólo en su estudio sino también en la obtención de heurísticas para su resolución. Su dificultad de resolución radica en que el número de posibles soluciones aumenta significativamente al aumentar el tamaño de la muestra (n), donde el número total de rutas o soluciones posibles que han de ser evaluadas son [Toro,2014]:

$$(n-1)!$$

2.5.1 Aplicaciones

El TSP se puede emplear en cualquier situación que requiera seleccionar nodos en cierto orden de cara a reducir los costos. Hasta el día de hoy, éste se ha aplicado sobre una gran variedad de problemas. A continuación, se comentan brevemente algunas de las aplicaciones más importantes del Problema del Agente Viajero [Garrido,2000].

- ***Logística:*** Las aplicaciones más directas y más abundantes del TSP se centran en el campo de la logística. El flujo de personas, mercancías y vehículos en torno a una serie de ciudades o clientes se adapta perfectamente a la filosofía del TSP. Dentro de este campo encontramos:

  - Vendedores y turistas: La mayoría de las agencias de viajes o agentes comerciales utilizan algún planificador de rutas para determinar cuál es el mejor camino para visitar los puntos que desean y regresar al punto de origen, ya sea el hotel o la oficina. Estos planificadores generalmente incluyen algún algoritmo de resolución del TSP.
  - Horarios de transportes laborales y/o escolares. Actualmente, muchas empresas dedicadas al transporte de personas adquieren software de resolución de TSP que les permite reducir gastos de una manera significativa.
  - Empresas de paquetería o servicios de comida a domicilio: en ocasiones el reparto de mercancía puede modelizarse como un TSP. Se trata de los casos en los que las casas están muy alejadas unas de otras o cuando sólo se debe visitar algunas de ellas.

- ***Industria:*** Las aplicaciones en industria no son tan numerosas como las anteriores, pero la aplicación del TSP a este ámbito también ha dado lugar a una significativa reducción de los costes. Entre estas aplicaciones encontramos:

  - Producción de placas de circuito impreso: Estas placas normalmente contienen numerosos agujeros para hacer posteriormente las conexiones oportunas. Dichos agujeros son producidos por perforadoras automáticas que se mueven entre los puntos especificados creando un agujero tras otro. Por tanto, la aplicación del TSP en este campo consiste en, tomando como ciudades cada una de las posiciones donde debe realizarse una perforación y las distancias entre ellas como el tiempo que necesita la máquina en trasladarse de una a otra, minimizar el tiempo que pierde la taladradora en moverse de una posición a otra. Esto no sólo ocurre con las perforadoras, también cuando es necesario realizar soldaduras o grabados [Possani,2013].

  - Microprocesadores personalizados: La misma clase de aplicación, pero a una escala mucho menor, fue la desarrollada en los Laboratorios Bell a mediados de los 80. Estos investigadores desarrollaron una técnica para la rápida producción de microprocesadores a medida. Este proceso comienza con un microprocesador básico, el cual está formado por una red de puertas lógicas sencillas. Después, porciones de esta red son cortadas con láser para crear grupos individuales de puertas que permiten al chip desarrollar las funciones especificadas por el cliente. En este caso, el TSP sirve para guiar al laser a través de los puntos que requieren ser cortados, lo que permite reducir el tiempo hasta un 50 % [Possani,2013].
  - Secuenciación de tareas: Se refiere al orden en el cual n trabajos tienen que ser procesados de tal forma que se minimice el costo total de producción. Supongamos que una maquina debe realizar una serie de tareas en el mínimo tiempo posible y sin importar el orden de las mismas. Supongamos que se tarda un tiempo t_{ij} en poner a punto la máquina para realizar la tarea j si la última tarea que realizó fue la i. En ese caso, podemos aplicar el TSP suponiendo que cada tarea es uno de los nodos a visitar y han de realizarse todas las tareas para producir el producto considerando que la distancia entre ellos es t_{ij}. El nodo origen y destino serán el estado de la maquina cuando empieza o termina el producto. Dado que el tiempo que se emplea en realizar cada tarea no depende del orden, no será necesario incluir estos tiempos en el modelo [Possani,2013].

- ***Búsqueda de planetas:*** A pesar de que normalmente asociamos el TSP con aplicaciones que requieren la visita física a lugares remotos, éste también puede ser empleado cuando los sitios se pueden observar desde la distancia sin necesidad de viajar realmente. Un ejemplo de ello puede darse cuando los nodos considerados son planetas, estrellas o galaxias, y las observaciones se deben hacer con algún tipo de telescopio. Interesantes ejemplos del TSP han sido considerados en el trabajo de planificación de misiones de telescopios espaciales de la NASA. Éste es usado para determinar la secuencia de observaciones que deberá hacer el telescopio una vez esté en su posición.

Cada día, las áreas donde trabaja el TSP son más innovadoras. Entre los últimos proyectos exitosos en referencia a la aplicación de éste último encontramos: planificación de caminos de senderismo, minimización de gasto de papel o patrones de corte en la industria cristalera.

2.5.2 Formulación matemática

Previamente se aclaran los siguientes conceptos [Wilson,1988]:

- Un ***circuito***, es una secuencia de aristas en la que el primer vértice y el último son el mismo y además no hay más vértices coincidentes que estos dos.

- Un ***camino hamiltoniano***, es un camino que recorre todos los nodos del grafo una y sólo una vez.

- Un ***circuito hamiltoniano***, es un circuito que recorre todos los nodos del grafo una y sólo una vez.

Conociendo estas definiciones, el problema del TSP puede ser descrito según la teoría de grafos de la siguiente manera: Sea $G = (V, E)$, un grafo completo, donde $V = 1, \ldots, n$ es el conjunto de nodos o vértices y E es el conjunto de arcos. Los nodos $i = 2, \ldots, n$ se corresponden con los lugares a visitar mientras que el nodo 1 es considerado la ciudad de origen y destino. A cada arco (i, j) se le asocia un valor no negativo $d_{i,j}$, que representa la distancia del vértice i al j. El uso de los arcos (i, i) no está permitido, por lo que se impone que $d_{i,i} = 1, \quad \forall i \in V^2$.

Se diferencian dos tipos de problema en función del tipo de grafo G:

- ***ATSP: (TSP asimétrico):*** Cuando G es un grafo dirigido. Lo que significa que la matriz de distancias es asimétrica; es decir, la distancia de ida no tiene porqué ser igual a la de vuelta.

- ***STSP: (TSP simétrico):*** Cuando G es un grafo no dirigido, es decir; $d_{i,j} = d_{i,j} \forall (i, j) \in E$, y por tanto la matriz de distancias es simétrica o lo que es igual, la distancia de ida es igual a la de vuelta.

Como ya se indicó el objetivo del Problema del Agente Viajero es encontrar una ruta que, comenzando y terminando en un mismo punto, en este caso denotado como 1, pase una sola vez por cada uno de los puntos restantes y minimice la distancia total recorrida. Si definimos las variables binarias de decisión X_{ij} para todo $(i, j) \in E$, de forma que tomen el valor 1 si el arco (i, j) forma parte de la solución y 0 en otro caso; tenemos que el problema de programación lineal asociado al Problema del Agente Viajero consiste en minimizar la siguiente función objetivo:

$$\sum_i \sum_j d_{ij} x_{ij}$$

Donde:

$$\sum_i x_{ij} = 1, \quad \forall j$$

$$\sum_i x_{ij} = 1, \quad \forall i$$

$$x_{ij} \in \{0, 1\}, \quad \forall ij$$

Estas restricciones son necesarias pero no suficientes, pues pueden dar lugar a varios subtours o subcircuitos, como se puede observar en la Figura siguiente. Obsérvese que $x_{16} = x_{65} = x_{51} = x_{24} = x_{43} = x_{32} = 1$, por lo que no se viola ninguna de las restricciones.

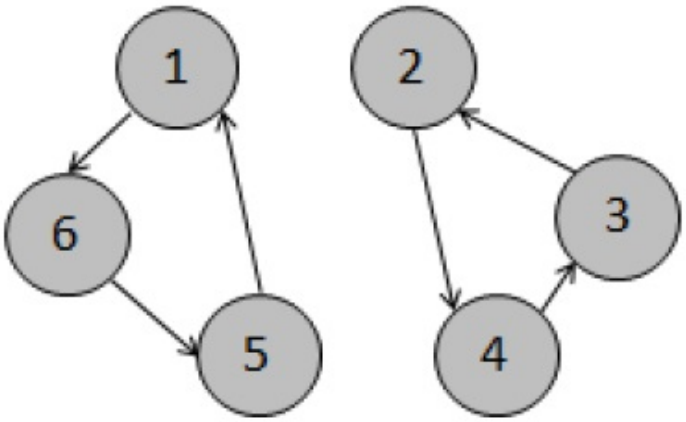


Figura 17. Formación de Subcircuitos.

Por tanto, para que el modelo proporcione una solución factible se debe añadir restricciones para eliminar los subtours o subcircuitos. De acuerdo con el proceso de resolución implementado se deben añadir estas restricciones que garanticen la eliminación de subcircuitos en la ruta óptima final

En el siguiente capítulo se iniciará con el planteamiento del problema a solucionar y las condiciones con las que se debe trabajar en la implementación del software que permita optimizar el desempeño de las rutas de mantenimiento con la utilización de la Red Neuronal Hopfield.

Cada día sabemos más y entendemos menos.
Albert Einstein

CAPÍTULO
3

Formulación del Problema

El servicio de mantenimiento preventivo de equipos y maquinarias que posee una empresa o industria con varias sucursales a nivel nacional, consiste en brindar visitas periódicas a cada una de sus sucursales con el fin de cumplir las rutinas establecidas para cada tipo de equipo a atender. Esto involucra que se deba mantener una planificación mensual, trimestral, semestral y anual con varios grupos de trabajo y así poder atender de una manera adecuada el funcionamiento los equipos. Para este proyecto se ha planteado trabajar con una institución financiera que posee más de 300 puntos de atención a nivel nacional, distribuidos geográficamente de una manera dispersa y no uniforme. A continuación se describirá cada parte que interviene en el proceso del servicio de mantenimiento preventivo.

3.1 Puntos o Nodos de Atención

Como puntos o nodos de atención se entiende a cada uno de los lugares a donde tendrán que acudir el grupo de mantenimiento para ejecutar las rutinas sobre los equipos y maquinarias, estos pueden ser:

3.1.1 Agencia Matriz

Se denomina agencia matriz a aquel lugar donde la institución financiera posee una infraestructura muy grande que involucra una gran cantidad de equipos y maquinarias que utiliza para su funcionamiento. En una agencia matriz se tiene ubicada gran parte de la infraestructura administrativa de la institución financiera y generalmente se ubicarán en las ciudades más importantes del país.

3.1.2 Agencia Sucursal

Se denomina agencia sucursal a aquel lugar en donde la institución financiera posee una infraestructura medianamente grande con equipos y maquinarias para su funcionamiento. En una agencia sucursal se encuentra ubicada la parte administrativa de una provincia o localidad, generalmente una agencia sucursal está ubicada en la capital de una provincia importante a nivel comercial o industrial. Para nuestro caso las agencias sucursales están ubicadas en:

- Ibarra
- Ambato
- Esmeraldas
- Manta
- Machala
- Latacunga
- Santo Domingo de los Tzachillas
- Quevedo
- Riobamba

3.1.3 Agencia Nodo de Comunicación

Se denomina agencia nodo de comunicación a aquel lugar en donde la institución financiera posee un centro de comunicaciones y servidores informáticos que permiten la operación de varias agencias esclavas que transmiten y reciben información a través de estas, por lo que su importancia radica en el enlace de comunicación que mantiene a nivel nacional. Para nuestro caso las agencias nodos de comunicación están ubicadas en:

- Quito
- Guayaquil
- Cuenca
- Babahoyo
- Tulcán
- Ibarra
- Ambato
- Esmeraldas
- Manta
- Machala
- Latacunga
- Santo Domingo de los Tzachillas
- Quevedo
- Riobamba
- Loja
- Portoviejo

3.1.4 Agencia Centro de Acopio

Se denomina agencia centro de acopio a aquel lugar en donde la institución financiera posee bóvedas centralizadas de acopio de sus recursos, de estos lugares se distribuyen los

mismos hacia el resto de agencias a nivel nacional.

3.1.5 Agencia Normal

Se denomina agencia normal a aquel lugar en donde la institución financiera posee una infraestructura pequeña o moderada con no muchos equipos y maquinarias y que están distribuidas en ciudades, parroquias, centros comerciales, etc., a nivel nacional. En estas se ejecutan todas las transacciones operativas de los clientes diariamente.

3.2 Prioridad de Atención

Como parte del servicio del mantenimiento preventivo, es necesario priorizar las atenciones a las diferentes agencias de la institución financiera dentro de la planificación por cada intervención, de tal manera que en una jornada de trabajo cada equipo atienda la mayor cantidad de agencias teniendo en cuenta su importancia o prioridad, la misma que está dada según el siguiente cuadro:

Tabla 4
Prioridad de Atención

Tipo de Agencia	Prioridad
Agencia Matriz	10
Agencia Nodo de Comunicación	8
Agencia Centro de Acopio	6
Agencia Sucursal	4
Agencia Normal	2

Como se puede observar en el cuadro anterior la prioridad establece un orden inicial por el cual se deberán atender los puntos o nodos dentro de la planificación, buscando siempre que en cada intervención o visita del grupo de mantenimiento se obtenga el valor más alto posible de la sumatoria de prioridades de los nodos atendidos.

3.3 Distribución geográfica de los puntos o nodos de atención

La distribución geográfica nacional de los puntos o nodos de atención no son uniformes y dependen de varios temas como por ejemplo el número de habitantes, industrias existentes, centros comerciales, puertos marítimos, etc.[1] A continuación, como método de planteamiento del problema se ejemplifica una zona específica del país, para este caso se ha tomado la Zona

[1] ANEXO 1.

de la provincia de Manabí en la formulación del problema, entendiendo que lo mismo se replica en todas las zonas a nivel nacional.

Tabla 5
Distribución de Nodos en Manabí

N°	Agencia	Provincia	Prioridad
1	EL CARMEN	Manabí	4
2	CALLE 13	Manabí	6
3	CALCETA	Manabí	2
4	BANCA COMUNAL CHONE	Manabí	6
5	CHONE	Manabí	2
6	TOSAGUA	Manabí	6
7	PICHINCHA	Manabí	2
8	PORTOVIEJO	Manabí	8
9	PEDERNALES	Manabí	2
10	SAN VICENTE	Manabí	4
11	BAHIA CARAQUEZ	Manabí	2
12	PASEO SHOPPING MANTA	Manabí	2
13	MANTA	Manabí	10
14	TARQUI	Manabí	6
15	BANCA COMUNAL PORTOVIEJO	Manabí	4
16	REALES TAMARINDOS	Manabí	2
17	MALL DEL PACIFICO	Manabí	2
18	JIPIJAPA	Manabí	2
19	PUERTO LOPEZ	Manabí	6
20	12 DE MARZO	Manabí	2

3.4 Equipos y maquinaria existentes

En cada agencia de la institución financiera se pueden encontrar varios equipos y maquinarias que intervienen para su correcto funcionamiento, los cuales deben ser intervenidos periódicamente con rutinas de mantenimiento preventivo, entre los más importantes se encuentran:

3.4.1 Aires Acondicionados

Son equipos eléctricos de refrigeración que permiten mantener una temperatura de confort a las áreas de atención al cliente, administrativas y de equipos.[2]

[2] ANEXO 2.

Figura 18. Equipos de Climatización.

3.4.2 Grupos electrógenos

Un grupo electrógeno o generador eléctrico es una máquina compuesta por un motor de combustión y un generador asíncrono que en conjunto se encargan de proporcionar energía eléctrica el momento que la red de suministro público falla.

Figura 19. Grupo Electrógeno.

3.4.3 UPS

Los UPS´S son equipos eléctricos que proporcionan energía estabilizada e ininterrumpida a las cargas sensibles y críticas que están dentro de las agencias.[3]

3.4.4 Recontadoras monedas / billetes

Son equipos eléctricos que permiten realizar conteo, clasificación e identificación de monedas y billetes falsos dentro de las agencias.

[3] ANEXO 2.

Figura 20. Equipos UPS's.

Figura 21. Recontadoras de Monedas.

Figura 22. Recontadoras de Billetes.

3.4.5 **Sistema de bombeo hidrosanitario y sistema contraincendios**

Todas las agencias poseen equipos de bombeo para los sistemas hidrosanitarios y también para los sistemas contra incendios.[4]

Figura 23. Sistema Hidroneumático.

Figura 24. Sistema contra Incendios.

[4]ANEXO 2.

3.4.6 **Tableros eléctricos**

Todas las agencias poseen armarios o tableros eléctricos donde se encuentran las acometidas principales de los grupos electrógenos o de los Ups.[5]

Figura 25. Tableros Eléctricos.

En el siguiente cuadro se describe el tiempo promedio de atención de los equipos y maquinarias anteriormente descritos.

Tabla 6
Tiempo de Atención por equipo

Equipo	Tiempo de atención
Aire Acondicionado	2 horas
Grupo Electrógeno	3 horas
UPS	1.5 horas
Recontadoras Monedas/Billetes	0.8 hora
Bombas sistema Hidrosanitario	1 hora
Bombas sistema contra Incendio	1 hora
Tableros Eléctricos	1 hora

3.5 **Grupos de trabajo**

Para atender a todos los equipos y maquinarias dentro de las agencias de la institución financiera se cuenta con 10 grupos de trabajo de mantenimiento, los cuales están formados

[5] ANEXO 2.

por 3 personas: 1 supervisor y 2 técnicos. Estos grupos de mantenimiento se encuentran distribuidos geográficamente según la afluencia de agencias que existe en cada zona. En la Figura 3.9 se muestra la distribución geográfica de los 10 grupos de trabajo.

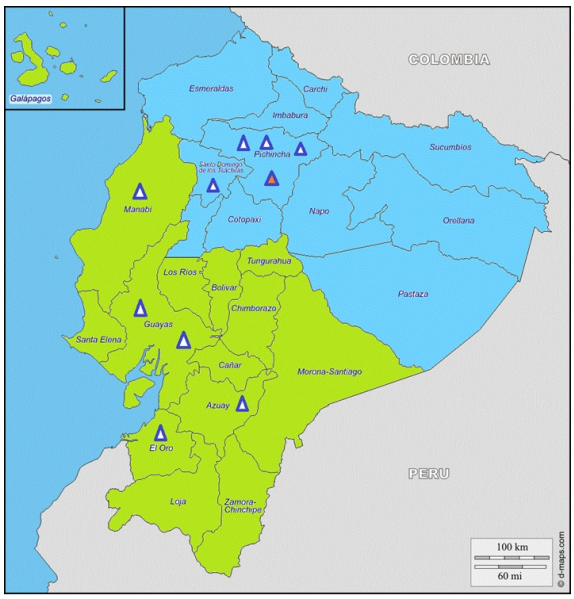


Figura 26. Distribución de Grupos de Trabajo.

Como se puede observar cada grupo de mantenimiento tiene a su cargo una zona de cobertura geográfica dentro de la cual deben movilizarse siguiendo la ruta más óptima de manera que los tiempos y costos de movilización y estadía sean los mínimos.[6]

A continuación, se muestra la gestión del grupo de la provincia de Manabí.

3.5.1 Grupos ME08

Este grupo de trabajo tiene como Ciudad Base : MANTA

Tabla 7
Distribución Geográfica ME08

Agencia	Provincia	Días de Atención	Longitud	Latitud
EL CARMEN	Manabí	5	-79,46477	-0,27143
CALLE 13	Manabí	2	-80,72699	-0,94841
CALCETA	Manabí	3	-80,1597	-0,84066
BANCA COMUNAL CHONE	Manabí	1	-80,09584	-0,6917
CHONE	Manabí	4	-80,09437	-0,69761

Sigue en la página siguiente.

[6] ANEXO 3.

Agencia	Provincia	Días de Atención	Longitud	Latitud
TOSAGUA	Manabí	2	-80,23434	-0,78422
PICHINCHA	Manabí	2	-79,82688	-1,04433
PORTOVIEJO	Manabí	8	-80,4525	-1,05472
PEDERNALES	Manabí	2	-80,0514	0,07311
SAN VICENTE	Manabí	2	-80,40205	-0,60472
BAHIA CARAQUEZ	Manabí	2	-80,42737	-0,61866
PASEO SHOPPING MANTA	Manabí	2	-80,70538	-0,96648
MANTA	Manabí	12	-80,70892	-0,96765
TARQUI	Manabí	3	-80,7159	-0,96448
BANCA COMUNAL PORTOVIEJO	Manabí	1	-80,46746	-1,0572
REALES TAMARINDOS	Manabí	3	-80,4687	-1,04304
MALL DEL PACIFICO	Manabí	1	-80,73209	-0,9421
JIPIJAPA	Manabí	2	-80,58273	-1,35256
PUERTO LOPEZ	Manabí	4	-80,80456	-1,55393
12 DE MARZO	Manabí	3	-80,4439	-1,05696

Donde los **Días de Atención**, corresponden al tiempo que el grupo de trabajo debe permanecer ejecutando sus actividades programadas, y el mismo depende del número de equipos que tiene cada nodo.

3.6 Jornadas de trabajo

Para el desarrollo de la planificación de las rutas de atención a las agencias es necesario conocer que las jordanas laborales de los grupos de trabajo pueden variar de acuerdo a las necesidades de cada agencia y a los viajes fuera de su ciudad base que deban ejecutar los grupos de mantenimiento. En el siguiente cuadro se establecen las jornadas de trabajo más comunes que pueden asignarse a un determinado equipo de trabajo.

Tabla 8
Jornadas de trabajo Típicas

Tipo	Días Laborables	Días Descanso
Jornada 1	10	4
Jornada 2	5	2
Jornada 3	20	8

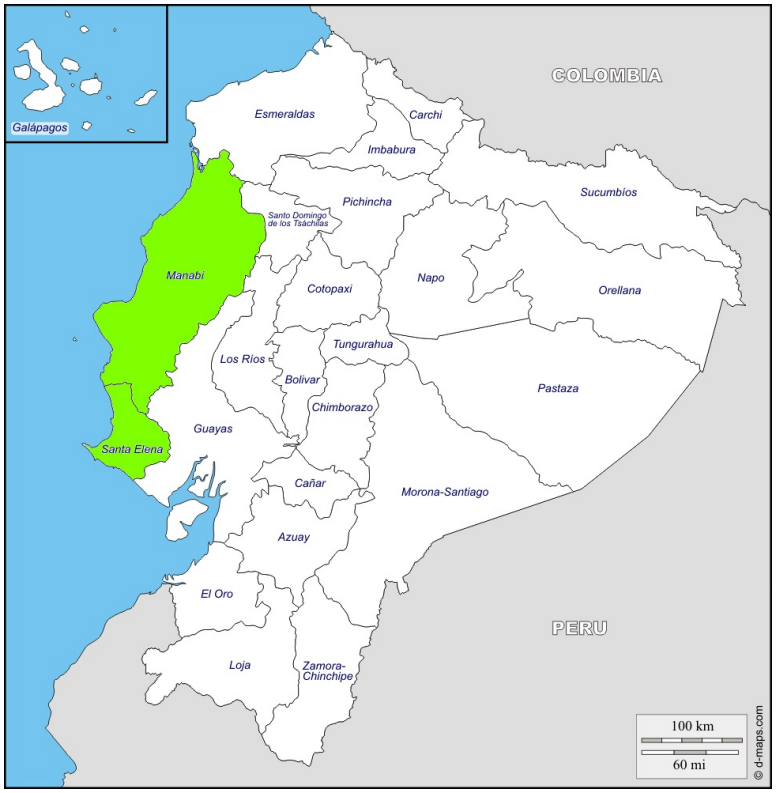


Figura 27. Distribución Geográfica ME08.

Para una jornada adicional se debe tener en cuenta que por cada 5 días laborables se deberán considerar 2 días de descanso de esta manera se establece esta relación:

Para n días laborables los dias de descanso son:

$$d_d = \frac{2n}{5}$$

Una vez que se conocen todos los nodos o puntos de atención, las prioridades de los mismos, su distribución geográfica, el tiempo que durara la intervención según el número de equipos existentes, la designación de los grupos de trabajo y el tiempo de duración de cada jornada laboral, es necesario desarrollar la planificación con las rutas de mantenimiento que indican el orden y cuáles son las agencias seleccionadas para cada grupo de trabajo, para esta actividad se necesita la utilización de métodos computacionales que consigan el objetivo.

3.7 Aplicación de la Teoría de Grafos

Para iniciar la formulación del problema se utilizan los 20 nodos que se encuentran en la provincia de Manabí y se los posiciona según sus coordenadas en un plano cartesiano donde las abscisas corresponden a la latitud de las coordenadas geográficas y las ordenadas corresponden a su longitud.

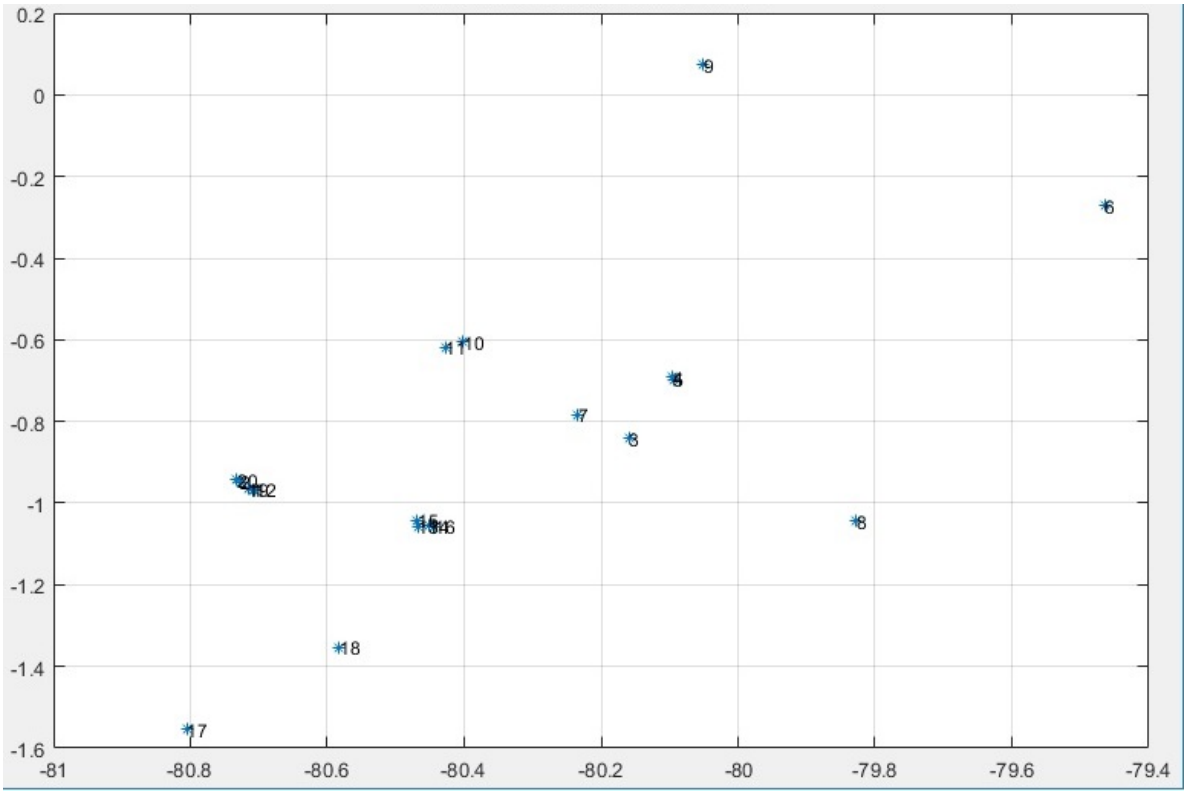


Figura 28. Distribución Nodos del grupo de mantenimiento ME08.

En la figura, el nodo etiquetado como 20, corresponde a ”12 de Marzo”, el nodo etiquetado como 19, corresponde a ”Puerto López”, el nodo etiquetado como 18, corresponde a ”Jipijapa”, el nodo etiquetado como 17, corresponde a ”Mall del Pacífico”, el nodo etiquetado como 16, corresponde a ”Reales Tamarindos”, el nodo etiquetado como 15, corresponde a ”Banca Comunal Portoviejo”, el nodo etiquetado como 14, corresponde a ”Tarqui”, el nodo etiquetado como 13, corresponde a ”Manta”, el nodo etiquetado como 12, corresponde a ”Paseo Shopping Manta”, el nodo etiquetado como 11, corresponde a ”Bahía de Caraquez”, el nodo etiquetado como 10, corresponde a ”San Vicente”, el nodo etiquetado como 9, corresponde a ”Pedernales”, el nodo etiquetado como 8, corresponde a ”Portoviejo”, el nodo etiquetado como 7, corresponde a ”Pichincha”, el nodo etiquetado como 6, corresponde a ”Tosagua”, el nodo etiquetado como 5, corresponde a ”Chone”, el nodo etiquetado como 4, corresponde a ”Banca Comunal Chone”, el nodo etiquetado como 3, corresponde a ”Calceta”, el nodo etiquetado como 2, corresponde a ”Calle 13”, y el nodo etiquetado como 1, corresponde a ”El Carmen”.

La ruta que recorra todos los nodos y mantenga un grafo no orientado, tiene muchas alternativas, pero para poder establecer la óptima es necesario utilizar algún método de cálculo diferente al de la fuerza bruta que llevaría a establecer todos los recorridos e identificar el menor de todos, esto se complica al aumentar el número de nodos hasta hacerlo imposible

por la cantidad de alternativas existentes, para el caso de los 20 nodos pueden existir varias rutas, para ejemplificar se han tomado dos de manera aleatoria buscando las distancias más cercanas entre nodos, de la siguiente manera:

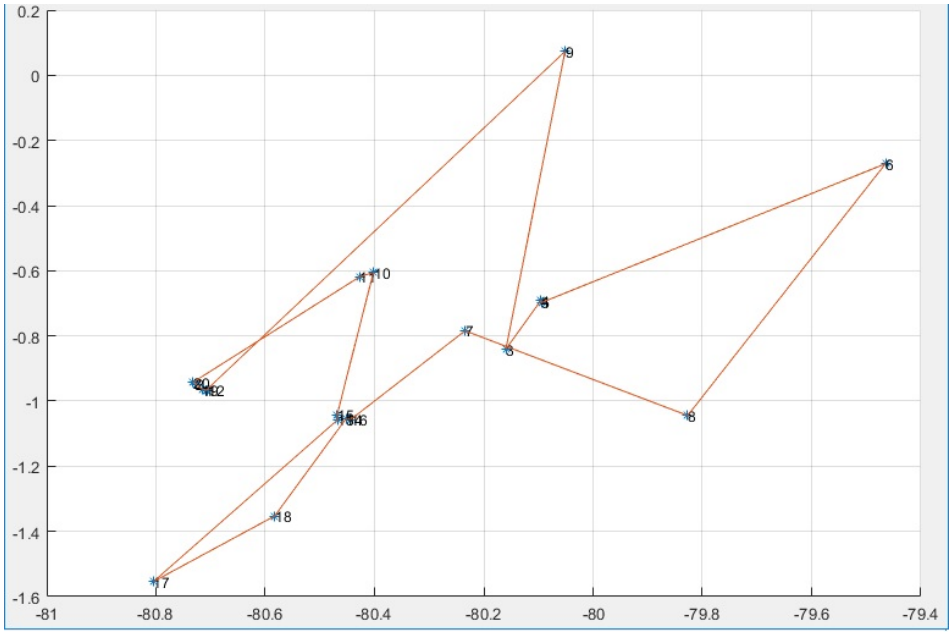


Figura 29. Alternativa 1 de ruta para nodos Manabí.

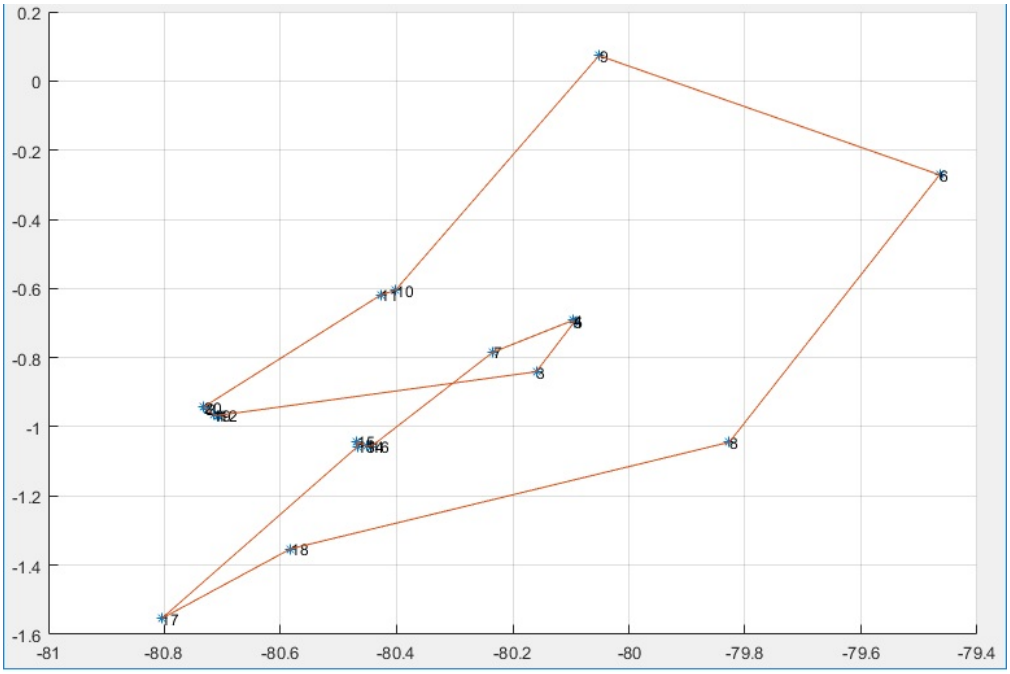


Figura 30. Alternativa 2 de ruta para nodos Manabí.

Dentro del cálculo de la ruta óptima hay que tomar en cuenta las variables adicionales que existen en proceso de mantenimiento y que son:

- El tiempo de estadía de los grupos de mantenimiento en cada nodo
- Las conexiones entre nodos que no pueden existir
- Los días disponibles para una jornada laboral de un grupo de mantenimiento
- La prioridad de atención de los nodos que siempre deberá ser la máxima posible

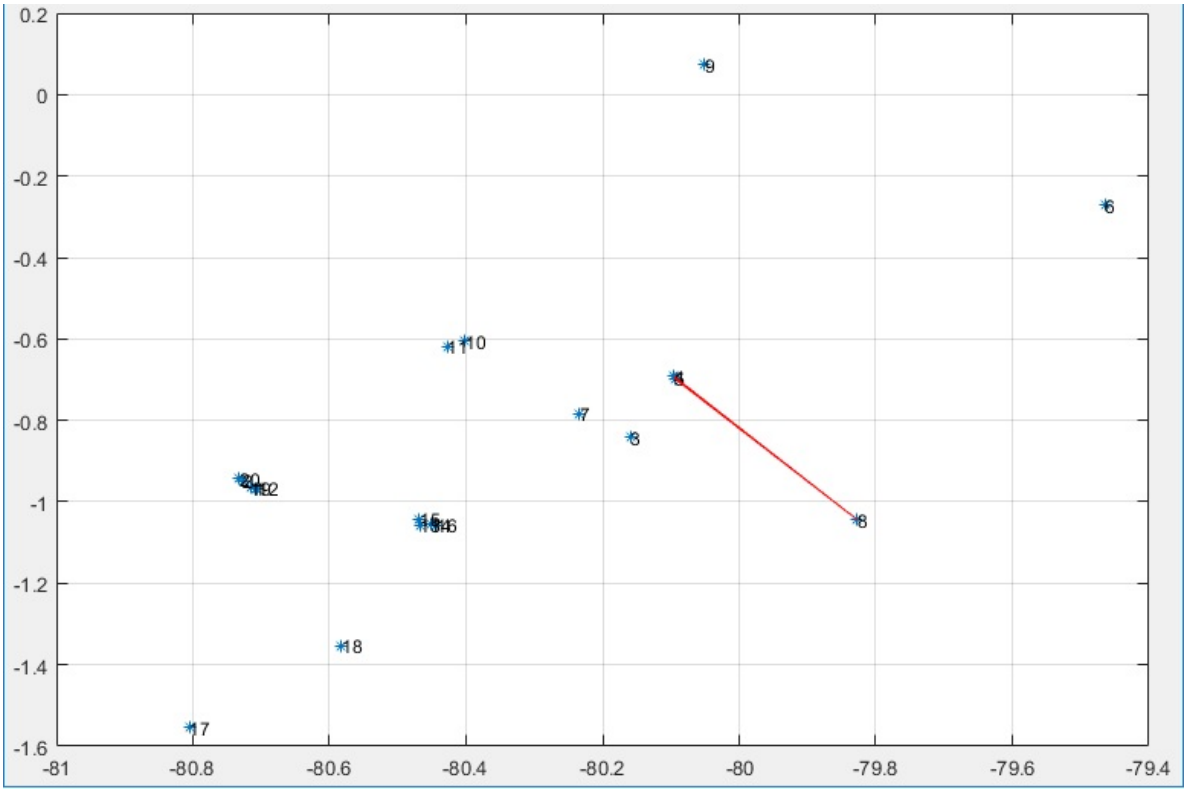


Figura 31. Conexiones que hipoteticamente no pueden producirse en la ruta de los nodos de Manabí.

Con estas restricciones previo al cálculo de la ruta óptima podemos utilizar el modelo del TSP para la resolución del problema.

Para ejemplificar en el caso de Manabí se ha dispuesto una hipotética restricción de acceso entre 2 grupos de nodos, situación que puede darse por las condiciones geográficas que no permiten el traslado entre los mismos (ríos, montañas, mar) y también por falta de vías de acceso.

3.8 Red Hopfield aplicada al TSP

Como se ha mencionado en el capítulo anterior el TSP o problema del Agente viajero mantiene una elevada complejidad de resolución y más si los nodos utilizados van creciendo en número, por esto para la resolución se ha planteado utilizar una Red Neuronal Artificial

tipo Hopfield y aplicarla al TSP. El algoritmo utilizado para la implementación de la RNA Hopfield es el denominado CHNN (Competitive Hopfield Neural Network) [Wen,2008].

Los enlaces de la red Hopfield están fuertemente conectando las neuronas y pueden resolver con éxito los problemas de optimización. La estructura de una red y los pesos de las conexiones entre las neuronas dependen de las restricciones específicas de un problema. Para cada neurona en la red, la entrada y las potenciales salidas se pueden definir, denotados por u y v, respectivamente [Hong,2005].

Para el caso que se ejemplifica, de los nodos existentes en Manabí, la matriz u inicia con valores randómicos:

$$u_{20*20} = [rand]_{20*20.} \tag{3.1}$$

En el Modelo de Hopfield, la función de activación más usual es la sigmoidea, para obtener la matriz v se utiliza la siguiente ecuación:

$$v(u) = \frac{1}{2}[1 + tanh(\alpha u)], \tag{3.2}$$

donde: α es la ganancia de la función que incluye el valor umbral Θi. Para valores grandes de α, v adopta el valor binario aproximado de:

$$v(u) = \begin{cases} 1, & \text{si } u > 0\text{; si se utiliza el camino} \\ 0, & \text{si } u < 0\text{; si no se utiliza el camino} \end{cases} \tag{3.3}$$

Aquí v es una matriz cuadrada compuesta por ceros y unos, en la cual en cada fila existe solamente un único número uno y el resto ceros, de igual manera en cada columna se tiene un uno y el resto ceros. Esta matriz se consigue al utilizar la función de activación sigmoidea en la Red Neuronal. Esta matriz crea la ruta resultante para los nodos que intervienen:

$$v_{20*20} = \begin{pmatrix} 0 & 0 & ... & 1 \\ 1 & 0 & ... & 0 \\ ... & ... & ... & ... \\ 0 & 1 & ... & 0 \end{pmatrix} \tag{3.4}$$

En una red compuesta de n neuronas, una función de energía E de la red está dada de la siguiente forma:

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} v_i v_j \tag{3.5}$$

Para este caso:

$$E = -\frac{1}{2}\sum_{i=1}^{20}\sum_{j=1}^{20} t_{ij} v_i v_j \tag{3.6}$$

$t_{ij} \quad (i, j = 1, ..., n)$ es el peso de la conexión entre la salida de la $j - th$ neurona y la entrada de la $i - th$ neurona. Todos los t_{ij} forman una matriz de pesos de conexión, pueden ser

positivos (estímulo excitador) o negativos (estímulo inhibidor) o igual a cero, es decir, no hay conexión desde la neurona j a la neurona i. La entrada potencial u_i de la $i-th$ neurona esta definida por la ecuación: [Tan,2005]

$$u_i = -\frac{\partial E}{\partial v_i} \qquad (i = 1, ..., n); \tag{3.7}$$

por lo tanto

$$u_i = \sum_{j=1}^{n} t_{ij} v_j \qquad (i = 1, ..., n) \tag{3.8}$$

En base a la terminología de la teoría de grafos, el TSP se define como un grafo Kn y una matriz simétrica que representa los pesos de los arcos o aristas de Kn, se debe entonces buscar el ciclo hamiltoniano en Kn de longitud mínima.

La resolución de problemas de optimización con la red Hopfield requiere una cuidadosa y adecuada elección de la función de energía, es decir, ponderaciones t_{ij}. La función E debe determinarse de tal manera que sus mínimos corresponden a las soluciones del problema considerado [Tan,2005].

La funcion de energía es de la forma:

$$E = \underbrace{\frac{A}{2} \sum_{x=1}^{n} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} v_{xi} v_{xj}}_{E_1} + \underbrace{\frac{B}{2} \sum_{i=1}^{n} \sum_{x=1}^{n} \sum_{\substack{y=1 \\ y \neq x}}^{n} v_{xi} v_{yi}}_{E_2} + \underbrace{\frac{C}{2} (\sum_{x=1}^{n} \sum_{i=1}^{n} v_{xi} - (n + \sigma))^2}_{E_3} +$$

$$\underbrace{\frac{D}{2} \sum_{x=1}^{n} \sum_{\substack{y=1 \\ y \neq x}}^{n} \sum_{i=1}^{n} d_{xy} v_{xi} (v_{y,i+1} + v_{y,i-1})}_{E_4} \tag{3.9}$$

donde

- E1: inhibición de fila, favorece solo 1 ciudad en una fila
- E2: inhibición de columna, favorece solo 1 ciudad en una columna
- E3: Inhibición global, favorecer el estado de que todas las ciudades están presentes
- E4: Inhibición de distancia, favorece la distancia mínima de la ruta

n corresponde al número de nodos, d_{xy} corresponde a la matriz de distancias entre los nodos x y y, A, B, C, D, y σ son constantes de calibracion de la red neuronal.

Como se dijo anteriormente, el punto crucial del diseño de la red es la elección correcta de ponderaciones de conexión o, de forma equivalente, constantes $A; B; C; D; \sigma; \alpha$. Las

condiciones finales de iteración teóricamente suceden cuando la función de energía de la red ya no se reduce y cuando la red alcanza el estado óptimo, pero en la práctica, si la función energética cambia para determinar el final del programa, existen problemas potenciales (como la complejidad del cálculo de las funciones energéticas o los errores que conducen a juicios inexactos, etc.). Por lo tanto, el experimento utiliza el número de iteraciones para controlar el final del programa, y de esta manera se evitarán errores o lazos infinitos. El ajuste de parámetros en esta investigación encontró que, cuando los parámetros de red inicialmente tomados fueron $(A = B = D) > C$, con σ pequeño , los resultados no fueron exitosos, por tal motivo, se debió determinar las constantes óptimas que lograrán calibrar la RNA; para ello, se realizaron distintas simulaciones hasta encontrar las que minimizaban el error en la solución.

Para el desarrollo de la RNA los parametros requeridos son:

- Número de Nodos.
- Coordenadas de las posiciones de los nodos.
- Matriz de distancias entre los nodos, entendiendo que no se trata de distancia Euclideana, sino la distancia que establece una ruta de vehículo.
- Matriz de permutación.
- Matriz de estado inicial con valores aleatorios.
- Matriz de restricciones.

Una vez determinados estos parámetros se debe calibrar la RNA hasta conseguir el resultado deseado, es decir buscar las contantes adecuadas que reduzcan el error de cálculo y permitan un resultado aproximado al real.

3.9 Optimización Knapsack

Como parte de las restricciones existentes en el desarrollo del software, se tiene que la ruta además de ser la óptima en logística y tiempo de traslado, debe abarcar el mayor número de nodos posibles tomando en cuenta la prioridad de los mismos y no superar el tiempo establecido para cada jornada laboral. Para este objetivo se aplica la resolución del problema de Knapsack mediante la programación dinámica que se basa en el llamado principio de óptimo, enunciado por Bellman en 1957 y que dice: [Bruno,2013]

> *"En una secuencia de decisiones óptima toda subsecuencia ha de ser también óptima".*

Hay que observar que, aunque este principio parece evidente no siempre es aplicable y por tanto es necesario verificar que se cumple para el problema en cuestión. El diseño de un algoritmo de Programación Dinámica consta de los siguientes pasos:

1. Planteamiento de la solución como una sucesión de decisiones y verificación de que ésta cumple el principio de óptimo.
2. Definición recursiva de la solución.
3. Cálculo del valor de la solución óptima mediante una tabla en donde se almacenan soluciones a problemas parciales para reutilizar los cálculos.
4. Construcción de la solución óptima haciendo uso de la información contenida en la tabla anterior.

Suponiendo que un problema se resuelve tras tomar un secuencia d_1, d_2, ..., d_n de decisiones. Si hay d opciones posibles para cada una de las decisiones, una técnica de fuerza bruta exploraría un total de d^n secuencias posibles de decisiones. La técnica de programación dinámica evita explorar todas las secuencias posibles por medio de la resolución de subproblemas de tamaño creciente y almacenamiento en una tabla de las soluciones óptimas de esos subproblemas para facilitar la solución de los problemas más grandes.

Dado el conjunto S de n objetos, sea S_k el conjunto de los k primeros objetos (de 1 a k). Podemos definir $B(k, w)$ como la ganancia de la mejor solucion obtenida a partir de los elementos de S_k para el problema de Knapsack de capacidad w [Moya,2017].

Definición recursiva de $B(k, w)$:

$$B(k,w) = \begin{cases} B(k-1,w), & \text{si } x_k = 0 \\ B(k-1, w-w_k) + b_k, & \text{si } x_k = 1 \end{cases} \tag{3.10}$$

Para la solución optima se debe seleccionar el valor maximo:

$$B(k,w) = max\{\underbrace{B(k-1,w)}_{V_1}, \underbrace{B(k-1,w-w_k)+b_k}_{V_2}\} \tag{3.11}$$

- V1: valor anterior calculado siempre que sea menor que la restrición de capacidad w
- V2: valor del cálculo actual mas el valor anterior siempre que sea menor que la restricción de capacidad w

donde: b_k corresponde al valor del cálculo actual, y k corresponde número de nodos utilizados, mientras que $B(k-1, w)$ corresponde al valor anterior calculado.

Para el caso de esta investigación b_k constituye el número días de la solución actual, w es el número de días disponibles que no puede ser sobrepasado(restricción), B es el valor de la prioridad obtenida, y k es el número de nodos que forman la solución óptima.

En el siguiente capítulo se desarrollará e implementará el software sobre la plataforma de **Matlab®**, en el cual se ingresarán todas las variables y restricciones del modelo para la obtención de las rutas óptimas de mantenimiento, mediante el uso de la Red Neuronal Hopfield.

Haz las cosas lo más simple que puedas, pero no te limites a lo simple.

Albert Einstein

CAPÍTULO

4

Implementación de Software y Resultados

Para el desarrollo e implementación del software que calcule la ruta optima de los grupos de trabajo de este proyecto se ha utilizado la plataforma de **Matlab**®, que es un software de Mathworks.inc, el cual por su robustez permitirá crear y simular una Red Neuronal Artificial tipo Hopfield y aplicarla en este problema de optimización, de igual manera se utilizará los criterios del problema de Knapsack con el fin de integrar restricciones al desarrollo del software y que éste sea más flexible y adaptable a las necesidades reales de la institución financiera. Como es de esperar el software debe contener interfaces gráficas hombre-máquina (HMI) que permitan un manejo e ingreso de información eficiente y así mismo presenten resultados gráficos que muestren las rutas obtenidas. Como herramienta adicional se ha enlazado el software de **Matlab**® con **Microsoft**® Excel para permitir que el usuario ingrese la información directamente de un archivo o plantilla previamente desarrollada y cargarla para la ejecución de los resultados. Para finalizar en el caso que fuere necesario se ha implementado adicionalmente un ADD-IN o función adicional del programa Excel para que de requerirlo se trabaje todo el análisis en Excel sin utilizar el software desarrollado en **Matlab**®. Ambas aplicaciones son de tipo ejecutable lo que indica que no es necesario que la computadora huésped donde se trabaje tenga instalado el programa **Matlab**®, siendo el único requisito que el ordenador tenga una plataforma Windows y también el software Excel de Office. A continuación, iniciamos el desarrollo e implementación del software

4.1 Implementación del Software

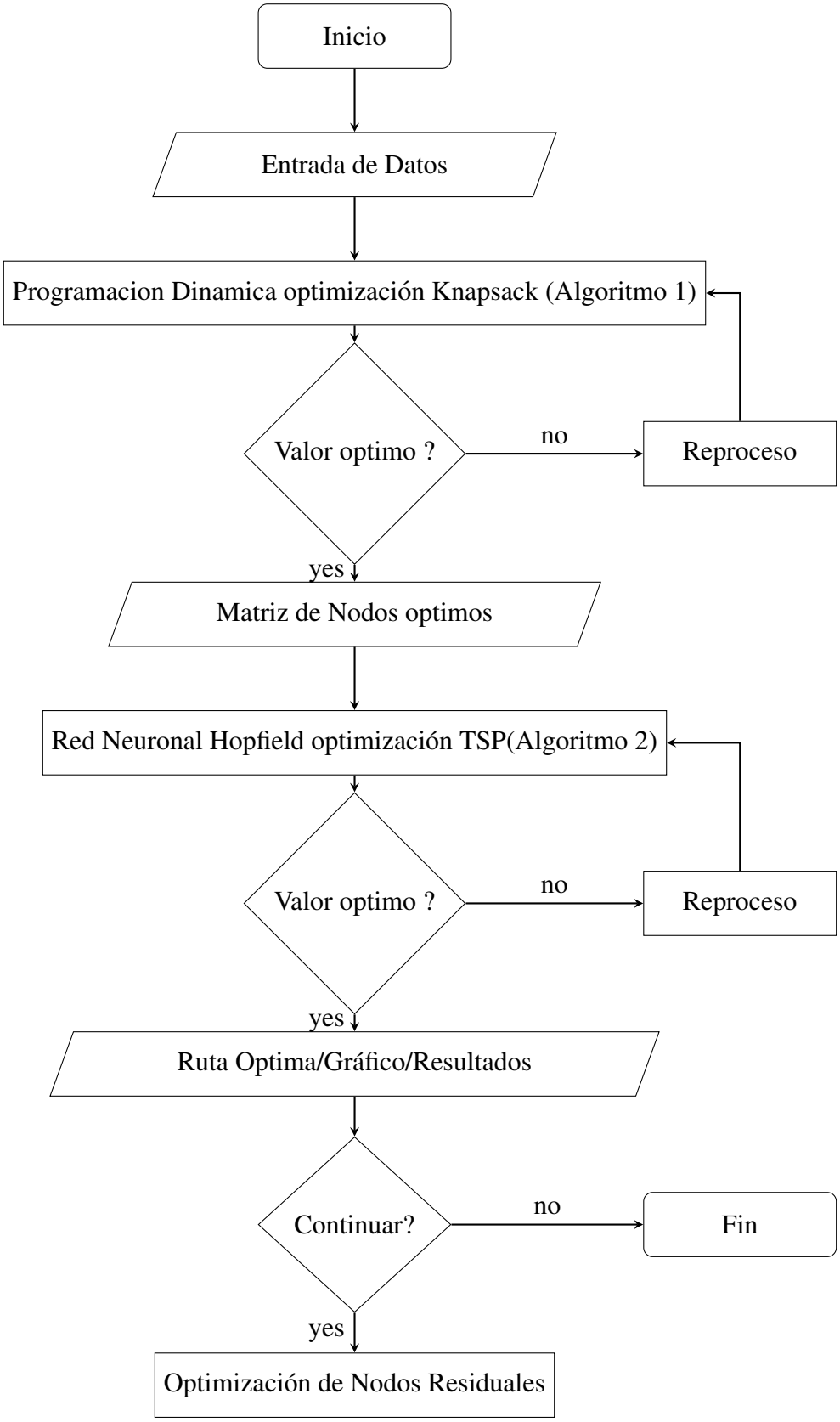
Para el desarrollo del software se ha utilizado la plataforma de **Matlab**®, sobre ella se ejecutará una aplicación instalable con interfaces gráficas que permiten el cálculo de la ruta

óptima de cada grupo de trabajo para el mantenimiento de las agencias de la institución financiera. La implementación del software inicia con el ingreso y recopilación de datos que los ejecuta el usuario, y en donde se encuentran las variables que se manejarán dentro del programa,así como las constantes de la función de energía de la red neuronal Hopfield. Como primer proceso de ejecución en el software se optimizarán, mediante la resolución del problema de Knapsack, un número de nodos de acuerdo con las jornadas de trabajo ingresadas y la prioridad que tenga cada uno.

Este proceso dará como resultado un conjunto de nodos previamente seleccionados que intervendrán en la ruta a optimizar, la información de este grupo de nodos serán los datos ingresados hacia la red Hopfield para la optimización final y objetivo principal del programa. Una vez optimizada la ruta se presentará gráficamente el recorrido con el orden de visitas de los nodos dentro de la ruta. Como información adicional el software calculará el tiempo de intervención total de recorrido, los nodos que no intervienen en la ruta, y la sumatoria de la prioridad total obtenida con la optimización.

Como implementación adicional el software posee una aplicación add – in que puede ser cargada en la plataforma **Microsoft**® Excel de office de manera que el usuario podrá utilizar el desarrollo de este investigación como una función más de este software en una hoja de cálculo cualquiera.

A continuación, se desarrollan los diagramas de flujo que intervienen en el proceso de cálculo:


Inicio
Entrada de Datos
Programacion Dinamica optimización Knapsack (Algoritmo 1)
Valor optimo ?
no
Reproceso
yes
Matriz de Nodos optimos
Red Neuronal Hopfield optimización TSP(Algoritmo 2)
Valor optimo ?
no
Reproceso
yes
Ruta Optima/Gráfico/Resultados
Continuar?
no
Fin
yes
Optimización de Nodos Residuales

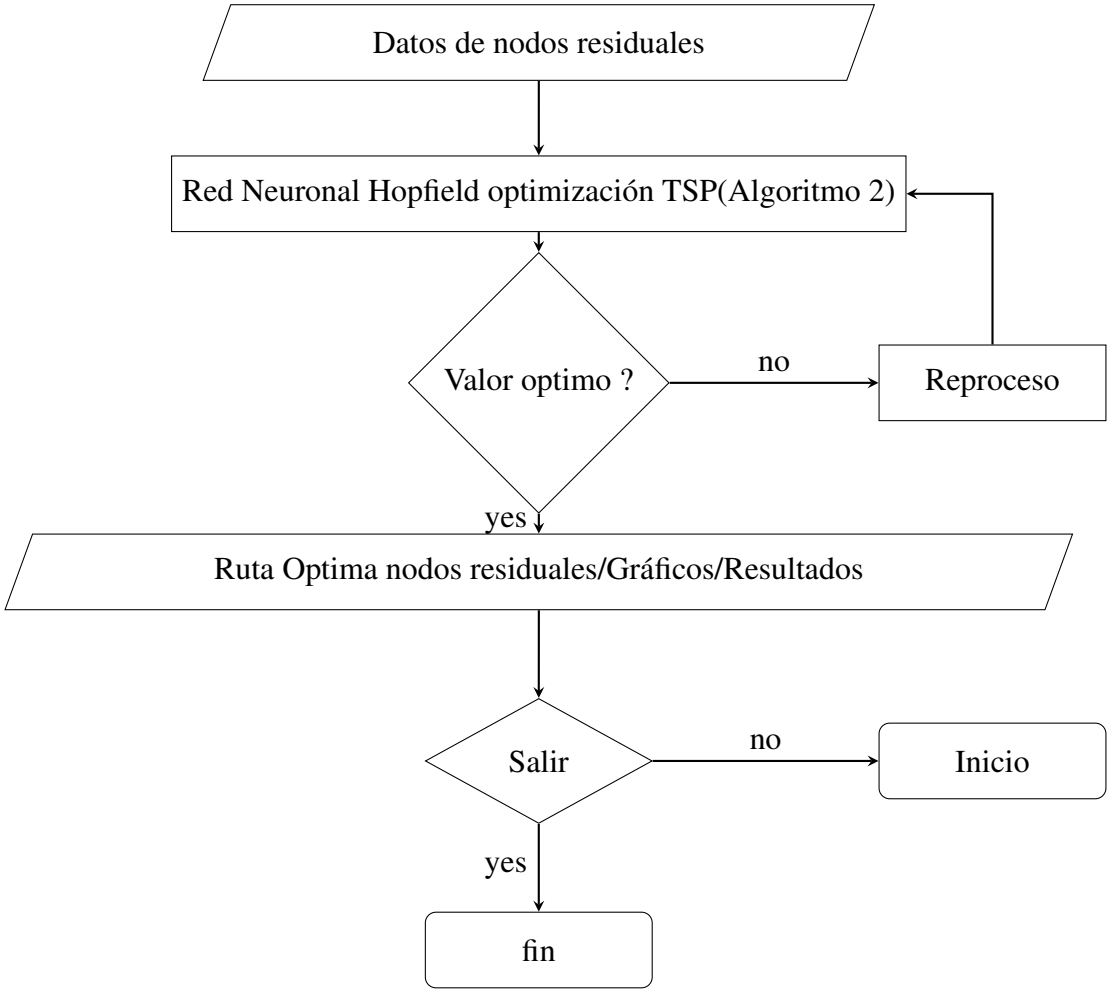


Figura 32. Diagrama de flujo software Optima Ruta.

Algoritmo 1: Optimización Knapsack

Entrada: Prioridad(values), Días de estadia (weights), Días de la Jornada (W).
Salida : Grupo de nodos optimizados (best), Prioridad obtenida (amount)

$$A = zeros(length(weights) + 1, W + 1)$$

for $j = 1$ ***to*** $length(weights)$ **do**
  for $Y = 1$ ***to*** W **do**
    if $weights(j) > Y$ **then**
      $$A(j+1, Y+1) = A(j, Y+1)$$
    else
      $A(j+1, Y+1) =$
       $max(A(j, Y+1), values(j) + A(j, Y - weights(j) + 1))$
$best = A(end, end)$
$amount = zeros(length(weights), 1)$
$a = best$
$j = length(weights)$
$Y = W$
while $a > 0$ **do**
  while $A(j+1, Y+1) == a$ **do**
    $j = j - 1;$
  $j = j + 1;$
  $amount(j) = 1$
  $Y = Y - weights(j)$
  $j = j - 1$
  $a = A(j+1, Y+1)$
return *best, amount*

Algoritmo 5: Subrutina nueva matriz de entrada U "Newu_ad"

Entrada: Matriz de entrada(u), Matriz de salida(v), Matriz de distancias (d), Número de nodos (n), Constantes de la Red Hopfield(A,B,C,D)
Salida : Nueva Matriz U (u)

for $xk = 1$ ***to*** n **do**
  for $k = 1$ ***to*** n **do**
    $u(xk, k) = u(xk, k) + A * lan + B * lan$ **for** $yk = 1$ ***to*** n **do**
      $u(xk, k) = u(xk, k) - v(xk, yk) * A * lan$
      $u(xk, k) = u(xk, k) - v(yk, k) * B * lan$
      if $(k == n)$ **then**
        $u(xk, k) = u(xk, k) - d(xk, yk) * v(yk, 1) * D * lan$
      else
        $u(xk, k) = u(xk, k) - d(xk, yk) * v(yk, k+1) * D * lan$
return u

Algoritmo 2: Optimización TSP Red Neuronal HOPFIELD

Entrada: Coordenadas Nodos(ciudad), Número de nodos (n), Constantes de la Red Hopfield(A,B,C,D), Número de Iteraciones (iteraciones)
Salida : Ruta optima (ruta), Distancia optima (kmtotal)

$cycle = floor(sqrt(iteracion))$
$test = cycle$
$summin = 0$
$dis = zeros(1, test)$
$u_0 = 0{,}02$
$lan = 1{,}0000e-005$
$delta = 0{,}1$
$place = ciudad$
$betterpath = ones(1, n+1)$
$u = zeros(n)$
for $tk = 1$ ***to*** $test$ **do**
  $del\_pre = 0$
  $u = 1/n + (2 * rand(n) - 1) * delta/n$
  $v = zeros(n)$
  for $ck = 1$ ***to*** $cycle$ **do**
    $[del, v] = Newv(u, v, u0, n)$ %subrutina Newv
    if $(abs(del - del\_pre) < (1e-15))$ **then**
      $break$
    $del\_pre = del$
    $[u] = Newu\_ad(u, v, d, n)$ %subrutina Newu_ad
  $[path, n\_fre, n\_ill, betterpath, dis(tk), summin] =$
  $Check(tk, ck, v, d, cycle, n, betterpath, summin)$ %subrutina Check
$kmtotal = summin * 100$
$ruta = betterpath$
return *ruta, kmtotal*

Algoritmo 3: Subrutina nueva matriz de salida V "Newv"

Entrada: Matriz de entrada(u), Matriz de salida(v), constante (u_0), Número de nodos (n)
Salida : Nueva matriz V (v), constante de calculo (del)

$del = 0$
for $k = 1$ ***to*** n **do**
  for $k1 = 1$ ***to*** n **do**
    $tmp = v(k, k1)$
    $v(k, k1) = 0{,}5 * (1 + tanh(u(k, k1)/u0))$ % se aplica funcion sigmodea
    $del = del + (v(k, k1) - tmp)^2$
return *del, v*

Algoritmo 4: Subrutina verifica resultados funcion Energía "Check"

Entrada: Ruta(betterpath), Número de nodos (n), Distancia mínima(summin), Matriz de salida(v), Matriz de distancias (d), Número de Iteraciones (iteraciones), constantes de cálculo(tk,ck)
Salida : Ruta(betterpath), Distancia mínima(summin), Sum de Distancias (sumdis), constantes de cálculo(path,n_fre,n_ill)

```
1  n_ill = 0
2  n_fre = 0
3  sumdis = 0
4  vi = zeros(1, n)
5  vj = zeros(1, n)
6  path = ones(1, n + 1)
7  ispath = ones(1, n)
8  for jk = 1 to n do
9      for k = 1 to n do
10         if (jk = k) then
11             if ispath(k) = 0 then
12                 if v(k, jk) >= vj(jk) then
13                     vj(jk) = v(k, jk)
14                     path(jk) = k
15     ispath(path(jk)) = 0
16 for k = 1 to n do
17     if (ispath(k) == 1) then
18         for jk = 1 to n do
19             if path(jk) == 1 then
20                 path(jk) = k
21                 ispath(k) = 0
22                 break
23 for k = 1 to n do
24     if vi(k) < 0,8 then
25         if ck > cycle then
26             n_fre = n_fre + 1
27         else
28             n_ill = n_ill + 1
29 path(n + 1) = path(1)
30 for k = 1 to n do
31     sumdis = sumdis + d(path(k), path(k + 1))
32 if tk == 1 then
33     summin = sumdis
34     betterpath = path
35 if sumdis < summin then
36     summin = sumdis
37     betterpath = path
38 return betterpath, summin, sumdis, path, n_fre, n_ill
```

4.2 Optima Ruta V1.0

Optima Ruta V1.0 es el software desarrollado en la plataforma de MATLAB, se basa en interfaces gráficas y permite ingresar los datos tabulados mediante un archivo de Excel, los datos que se ingresan como parámetros para la operación del software son:

Figura 33. Software Optima Ruta V1.0.

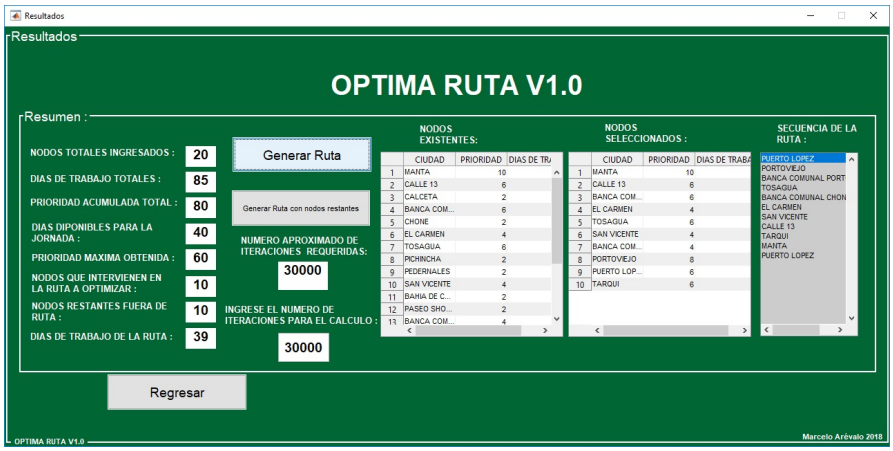


Figura 34. Software Optima Ruta V1.0 - Resultados.

- Coordenadas de la ubicación geográfica de los nodos.
- Prioridad de cada nodo ingresado.
- Días de estancia en cada nodo ingresado.
- Matriz de distancias de rutas entre los nodos, esta matriz es simétrica con su diagonal igual a 0.
- Rutas inexistentes o rutas que son imposibles de utilizar entres nodos por su condición geográfica y/o existencia de carreteras.

Una vez ingresados los datos se procesa la información con un parámetro adicional que es el número de días que se tienen disponibles para la ruta. De esta manera se tiene completa la información requerida para la obtención de los resultados

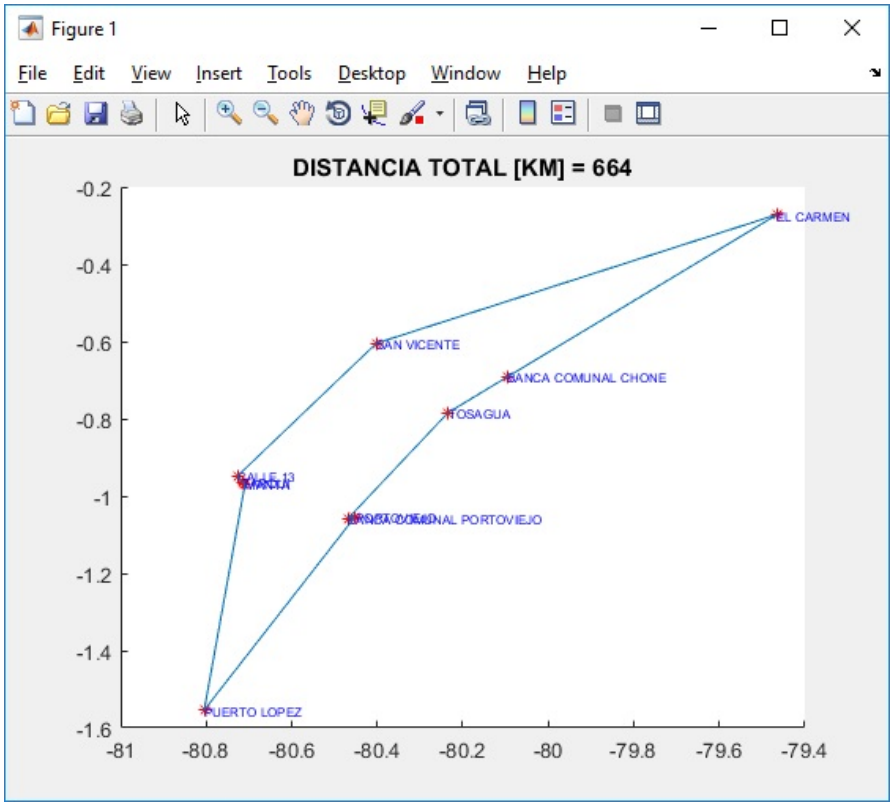


Figura 35. Software Optima Ruta V1.0 - gráfico ruta.

4.3 Resultados

De acuerdo con simulaciones preliminares, se establecen como mejor alternativa los siguientes valores de parámetros:

Ajuste de parámetros: En las simulaciones se encontró que cuando los parámetros de calibración de la red $A = B = D = 500, C = 200, \sigma = 0,02$, casi ninguna solución es legal u óptima. Esto significa que, en la función de energía E, el parámetro C es menor con respecto a A,B y D, por lo que el parámetro básico C se ajusta a 1000.

Con esos parámetros se inicializan las coordenadas de los nodos, los parámetros de red y el número de nodos, para la matriz de estado y permutación (u, v) se inician con números aleatorios. En el caso en el que el número de ciudades n y el paso de actualización de la red es fijo, el tiempo para resolver cada matriz de salida v es fijo, de modo que la comparación del tiempo para cada solución legal puede producirse comparando el número de ciclos.

Las constantes A, B y C son los pesos para garantizar la solución óptima, A es el coeficiente de peso para la obtención del valor de uno para cada fila, B es el coeficiente de peso para la

obtención del valor de uno para cada columna y C es el coeficiente de peso que asegura la longitud total más corta de la ruta Cuando C es menor que A y B $(A = B = 500, C = 200)$, el experimento difícilmente encontrará la solución óptima, en la mayoría de los casos la matriz de salida presenta dos o mas filas de ceros , y el programa es a menudo atrapado en un bucle infinito. Esto demuestra que la tercera parte de la legitimidad de la solución no recibió suficiente atención. Por lo tanto, se debe aumentar gradualmente el parametro C y observar los resultados experimentales, cuando C es 500, la situación aún no mejora significativamente, cuando C es 1000, la frecuencia de la solución óptima mejoró significativamente.

Se concluye entonces que cuanto menor es el valor de C no se puede garantizar la legitimidad de la solución, y la frecuencia de la solución óptima es obviamente mejor cuando C es grande.

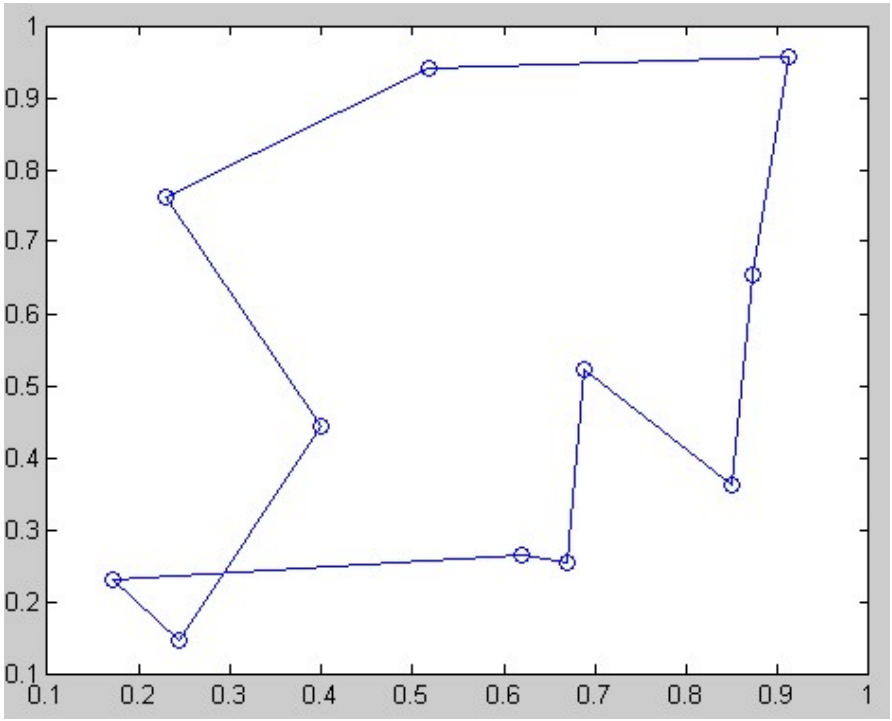


Figura 36. Resultado para un grupo de 11 nodos.

El coeficiente de peso D refleja la proporción de la longitud de la ruta en la función energética. Cuando D se toma como 200, el promedio de cada 1,5 veces se obtiene una solución óptima, pero la longitud de la ruta es muy grande, cuando D es de 500, el promedio de cada 6,7 veces se obtiene una solución óptima, cuando D es 600, la frecuencia de la aparición de una solución óptima ha disminuido, cuando D es 700, se requiere 151 veces para obtener una mejor solución, cuando D se toma como 1000, el programa está casi atrapado en un ciclo sin fin, la probabilidad de una solución óptima es muy baja. En el caso de un valor bajo de D, la legitimidad de la solución es relativamente fuerte, por lo que la frecuencia de

V =

0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0

Figura 37. Matriz de salida.

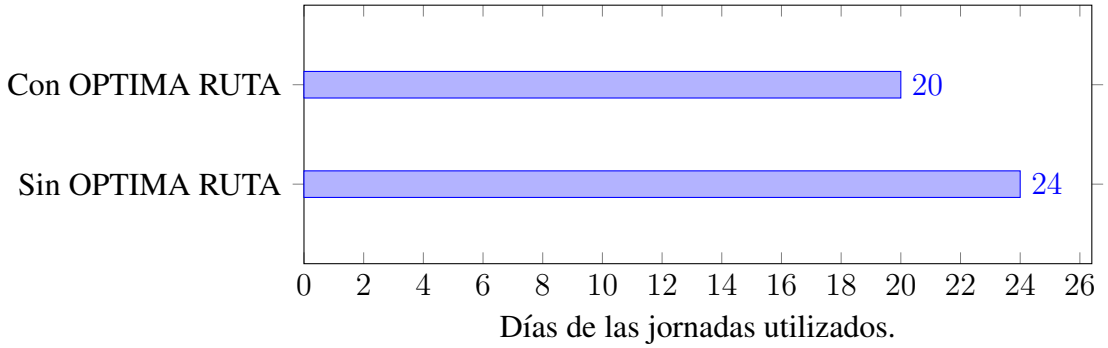
la solución óptima es grande, pero la longitud de la ruta es muy grande cuando D es grande, la frecuencia de la solución óptima se reduce, pero la longitud de la ruta es obviamente menor, La probabilidad de la solución óptima es relativamente mayor, y cuando D es demasiado grande, debido al énfasis excesivo en la longitud de la ruta, es difícil de encontrar una solución legítima u óptima.

Cuando σ es $0,0001$, el resultado es como se describe en la Figura 4.5. Cuando σ se toma como $0,001$, la solución promedio se produce cada 2,5 veces. Se puede observar que cuando σ es grande, la matriz de estado cambia mucho, lo que aumentará la frecuencia de la solución óptima, pero cuando σ es demasiado grande, la matriz del estado será difícil de obtener.

4.3.1 Comparación de resultados

Una vez implementado el software OPTIMA RUTA V1.0 se pueden validar los resultados del cálculo de las rutas de mantenimiento al compararlos con la planificación normal previa sin el software, a continuación se detallan los siguientes casos:

- ***Cálculo con una jornada de 20 días:***



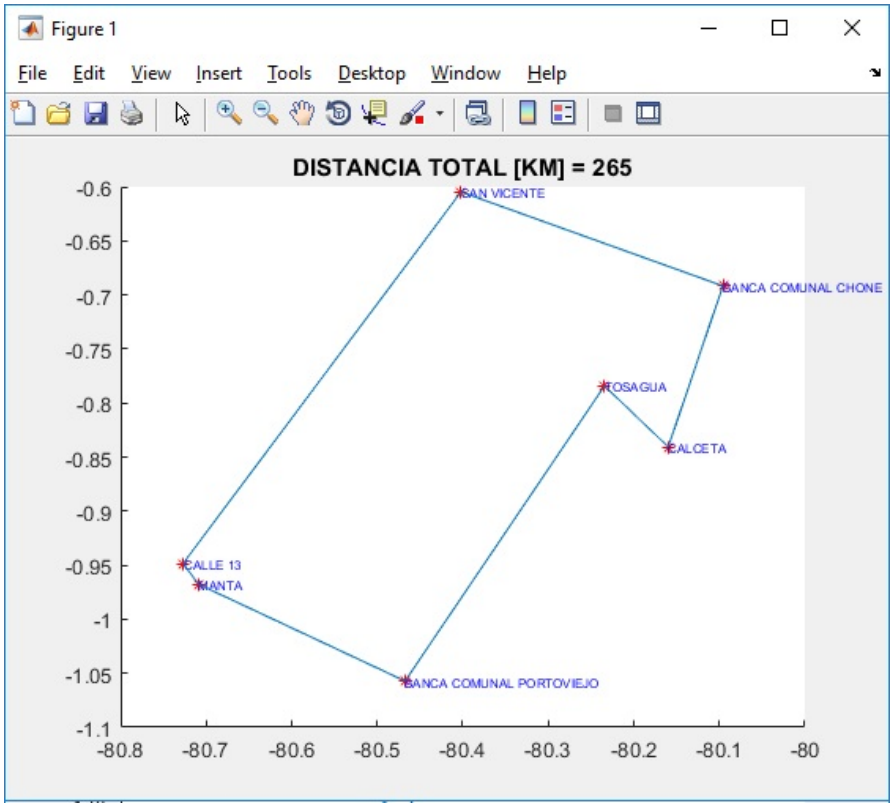


Figura 38. Cálculo de ruta con OPTIMA RUTA V1.0
Jornada 20 días

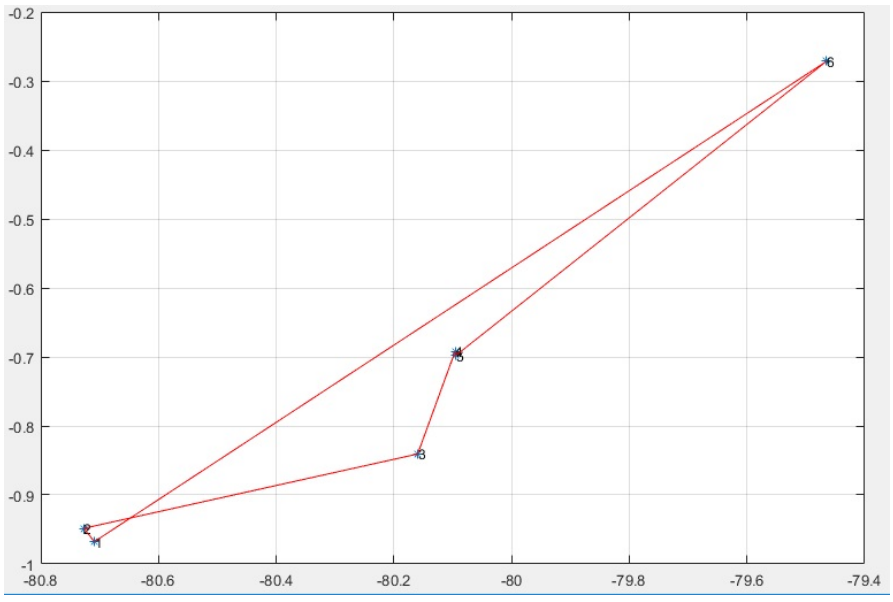
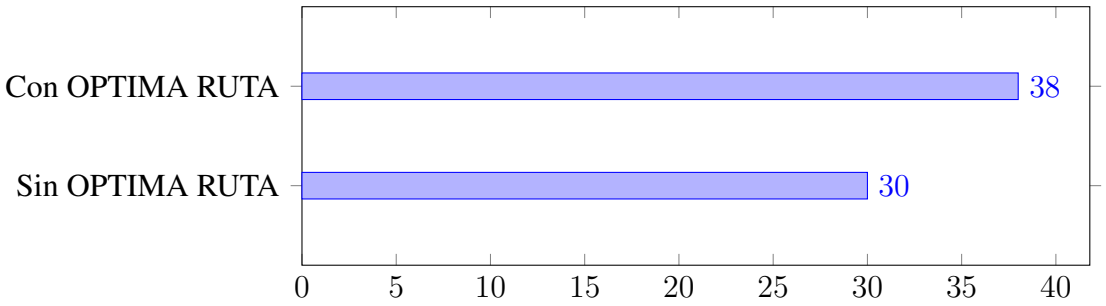
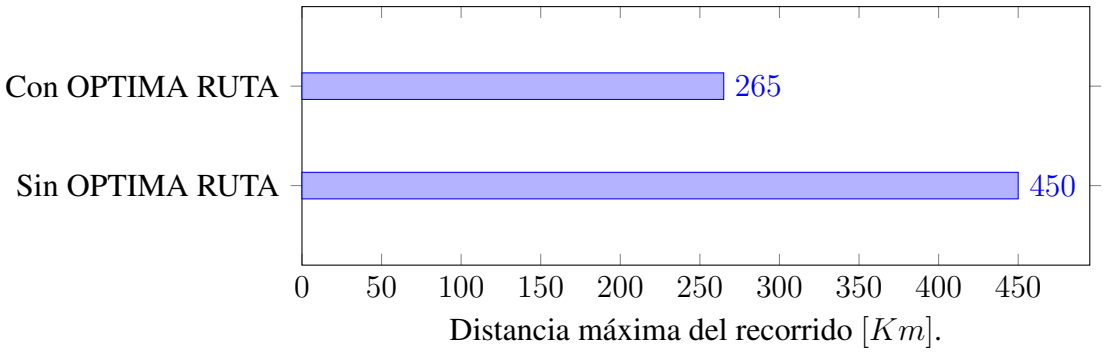
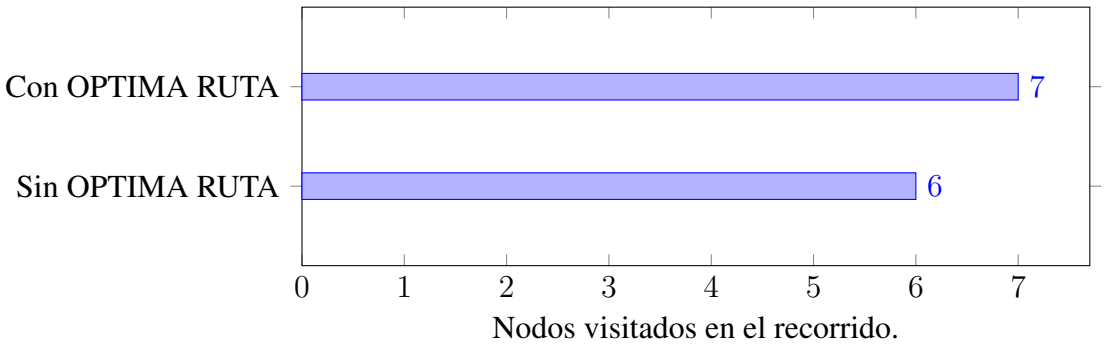


Figura 39. Cálculo de ruta SIN OPTIMA RUTA V1.0
Jornada 20 días







Al comparar la ruta durante una jornada de 20 días se observa que el software OPTIMA RUTA V1.0 consigue aprovechar todos los 20 días de la programación, lo que no sucede en la planificacion manual que requiere 4 días adicionales lo que implica una ruta incompleta. De igual manera con el software podemos obtener una prioridad mayor, una distancia de recorrido menor y un mayor número de nodos visitados. Con esto se valida la efectividad del software OPTIMA RUTA V1.0.

- ***Calculo con una jornada de 40 días***

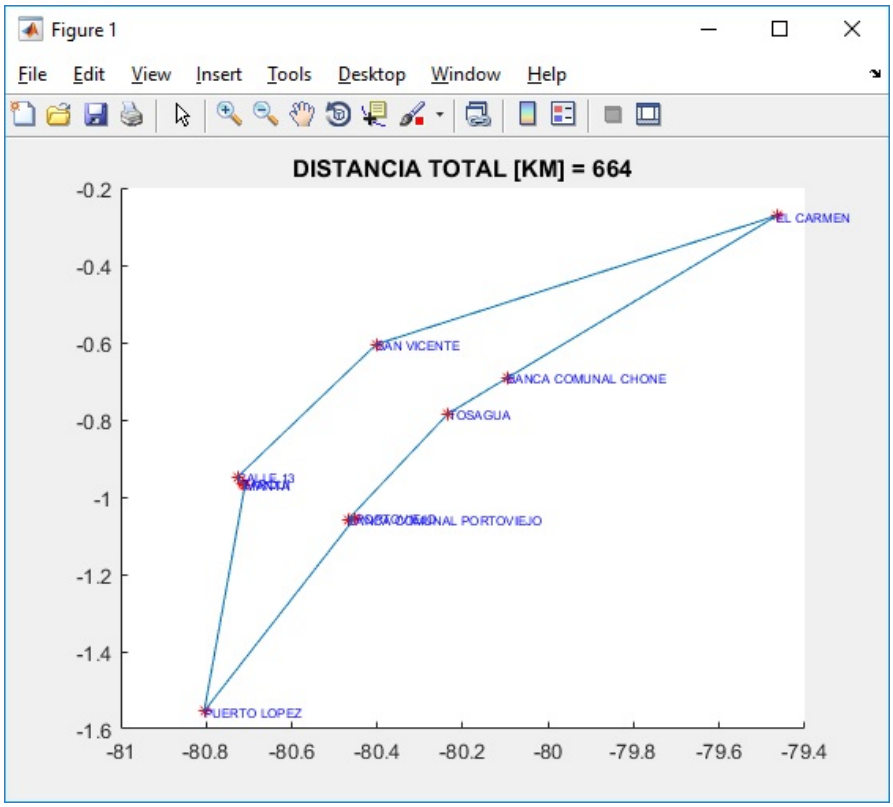


Figura 40. Cálculo de ruta con OPTIMA RUTA V1.0
Jornada 40 días

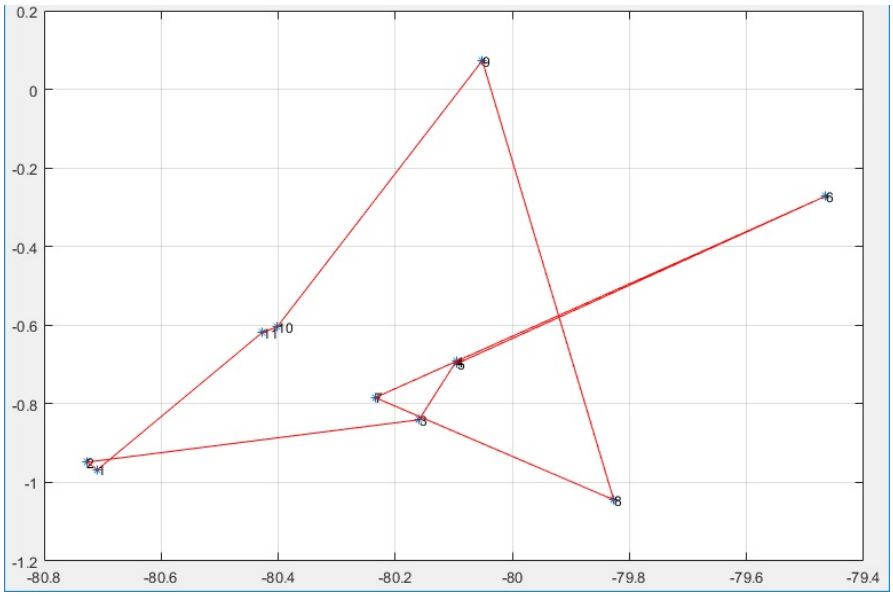
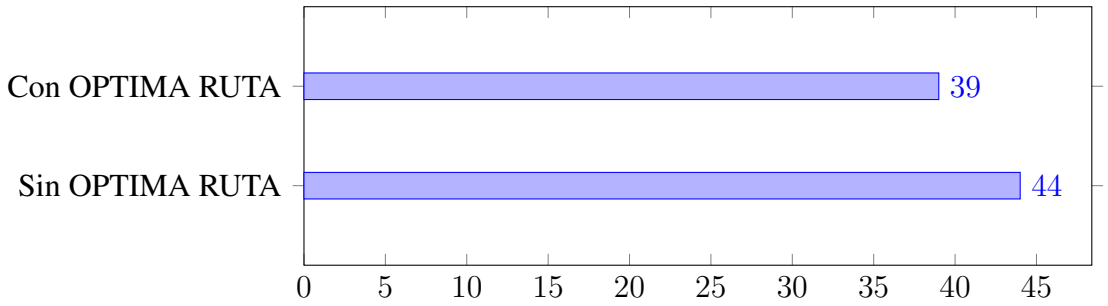
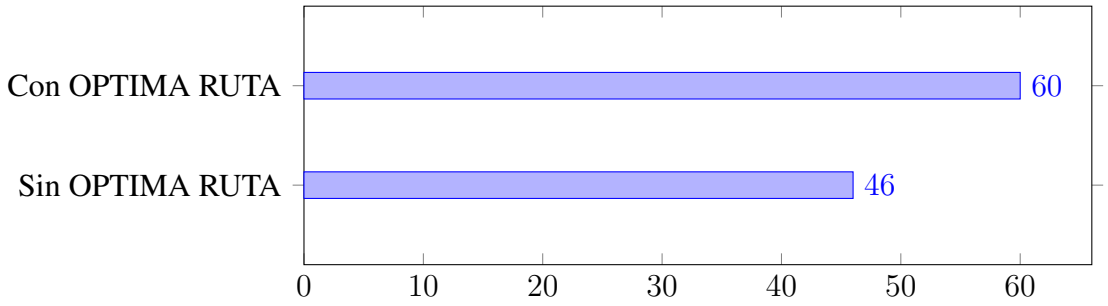


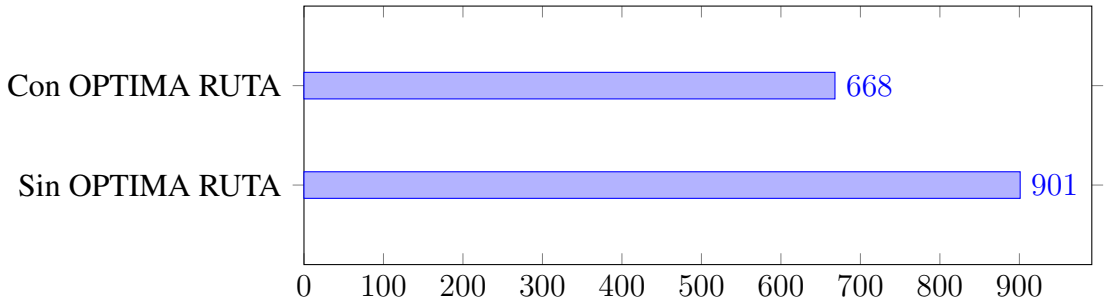
Figura 41. Cálculo de ruta SIN OPTIMA RUTA V1.0
Jornada 40 días



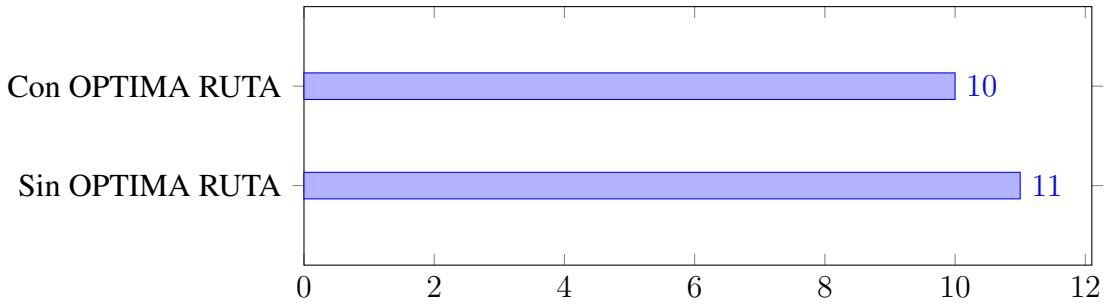
Días de las jornadas utilizados.



Prioridad máxima obtenida.



Distancia máxima del recorrido $[Km]$.



Nodos visitados en el recorrido.

Al comparar la ruta durante una jornada de 40 días se observa que el software OPTIMA RUTA V1.0 consigue aprovechar todos los días de la programación e incluso termina la jornada 1 día antes de lo requerido, lo que no sucede en la planificacion manual que requiere 4 días adicionales lo que implica una ruta incompleta. De igual manera con el software podemos obtener una prioridad mayor, una distancia de recorrido menor. Con esto se valida la efectividad del software OPTIMA RUTA V1.0.

- ***Calculo con una jornada de 80 días***

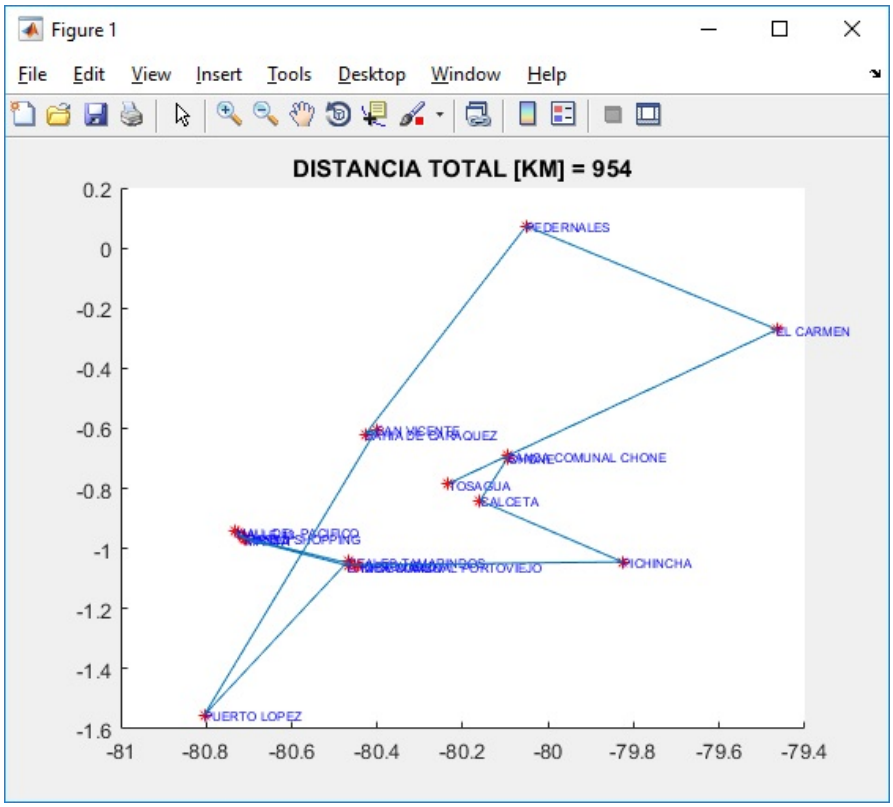


Figura 42. Cálculo de ruta con OPTIMA RUTA V1.0
Jornada 80 días

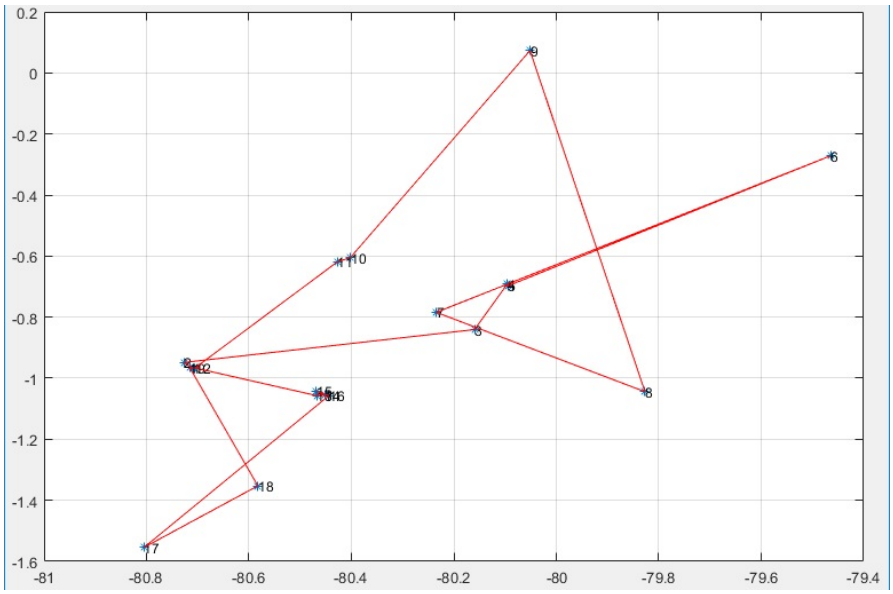
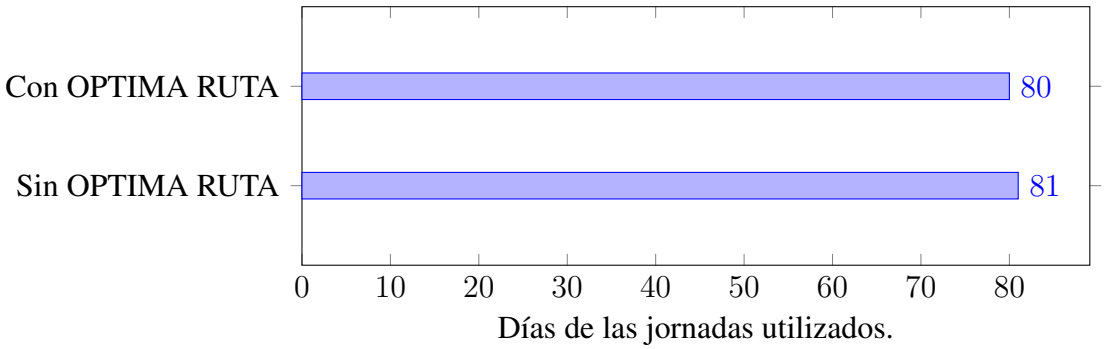
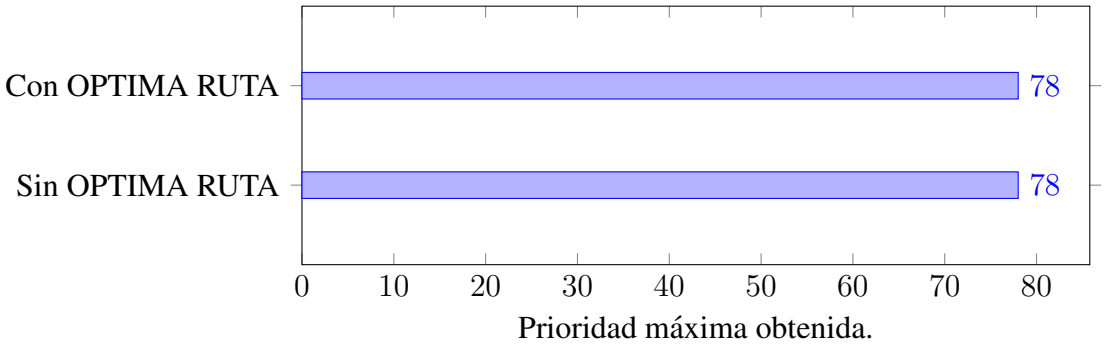


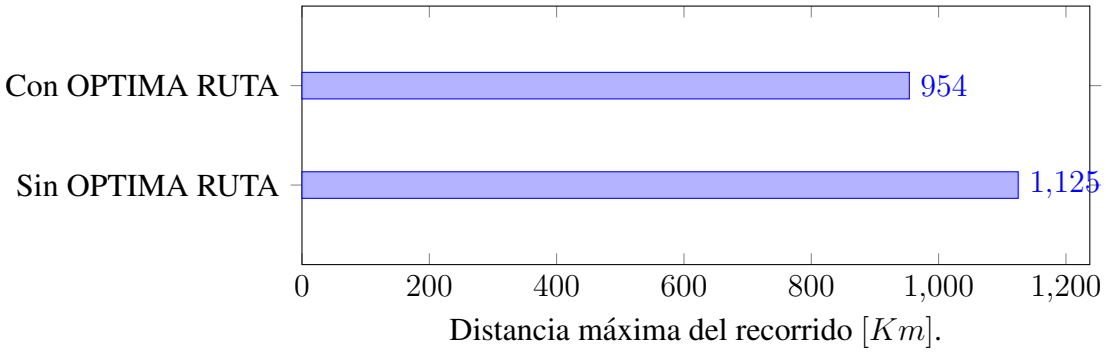
Figura 43. Cálculo de ruta SIN OPTIMA RUTA V1.0
Jornada 80 días



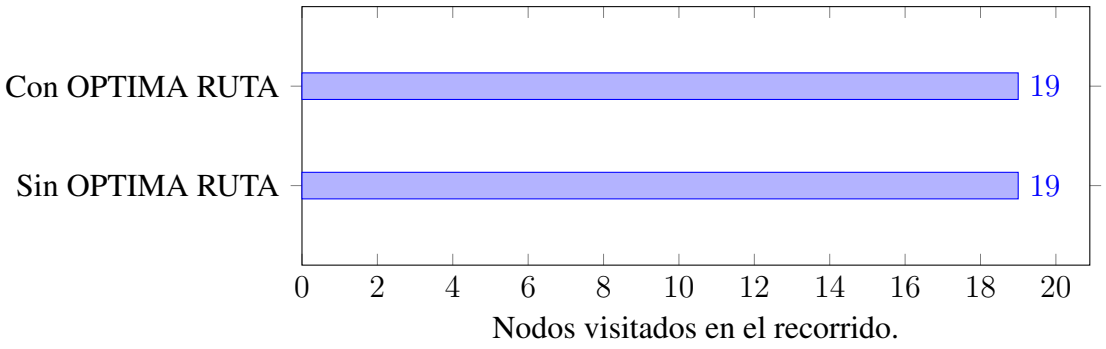
Días de las jornadas utilizados.



Prioridad máxima obtenida.



Distancia máxima del recorrido $[Km]$.



Nodos visitados en el recorrido.

Al comparar la ruta durante una jornada de 80 días se observa que el software OPTIMA RUTA V1.0 consigue aprovechar todos los 80 días de la programación, lo que no sucede en la planificacion manual que requiere 1 días adicionales lo que implica una ruta incompleta. De igual manera con el software podemos obtener una distancia de recorrido menor. Con esto se valida la efectividad del software OPTIMA RUTA V1.0.

Tabla 9
Resultados Rutas sin Optimizar

Tiempo disponible para la jornada	Número de nodos Ruta Normal	Prioridad Máxima	Días Excedentes	Km
5*	2	16	2	20
8	3	18	3	171
10	3	18	1	171
20	6	30	4	450
30	8	38	1	598
35	9	40	1	897
40**	11	46	4	901
50	13	52	0	975
60	15	62	3	988
70	17	70	1	1154
80	19	78	1	1125

*** :** Para este caso particular la optimización parecería no ser necesaria al momento de visualizar el dato de la distancia recorrida, pero es importante profundizar el análisis pues se trata de una condición especial ya que por el tiempo reducido de la jornada disponible que en este caso son 5 días, los únicos nodos que intervienen son apenas 2 y se han tomado los más cercanos sin tomar en cuenta ningún razonamiento adicional, pero está a pesar de ser una ruta supuestamente más corta no cumple con los día requeridos por ejecución y sobrepasa el tiempo de ejecución en 2 días laborables, haciéndola NO OPTIMA.
*** * :** Para este ejemplo, con una jornada de 40 días se tiene 11 nodos en la ruta, alcanzando una prioridad máxima de 46 y con un exceso de 4 días más de los requeridos, esta ruta tendría un recorrido de 901 [Km], por lo que es una ruta NO OPTIMA.

Tabla 10
Resultados Optima Ruta V1.0

Tiempo disponible para la jornada	Número de nodos Ruta Normal	Prioridad Máxima	Días Excedentes	Km
5	2	16	0	178
8	4	24	0	208
10	4	26	0	208
20	7	38	0	265
30	8	50	-1	424
35	9	56	-1	423
40*	10	60	-1	668

Sigue en la página siguiente.

Tiempo disponible para la jornada	Número de nodos Ruta Normal	Prioridad Máxima	Días Excedentes	Km
50	13	66	-1	810
60	15	70	0	963
70	17	74	0	917
80	19	78	0	861

*** :** Para este ejemplo, con una jornada de 40 días se tiene 10 nodos en la ruta, alcanzando una prioridad máxima de 60 y con un 1 día menos a los disponibles, constituyendo una optimización adicional, esta ruta tendría un recorrido de 668 [Km], por lo que es una ruta OPTIMA.

El software OPTIMA RUTA V1.0 puede ser aplicado a cualquiera de los grupos de trabajo manteniendo los valores de sus constantes de calibración, pues la red neuronal Hopfield funciona a través de las coordenadas de los nodos, sus restricciones y los días disponibles para cada jornada laboral.

4.3.2 Resumen Financiero

Para completar el análisis se muestra a continuación un resumen financiero en el cual se analiza el beneficio económico obtenido al utilizar el software OPTIMA RUTA V1.0, para los diferentes escenarios que se han experimentado:

Tabla 11
Comparación de días utilizados para las rutas con y sin optimización, según el tiempo de cada jornada

Días para la jornada	20 Días	30 Días	40 Días	50 Días	60 Días
Sin Optimización	24 días	31 días	44 días	50 días	63 días
Con Optimización	20 días	29 días	39 días	49 días	60 días
% de Ahorro	20 %	7 %	13 %	2 %	5 %

Tabla 12
Comparación de Km recorridos en las rutas con y sin optimización, según el tiempo de cada jornada

Km recorridos en la ruta	20 Días	30 Días	40 Días	50 Días	60 Días
Sin Optimización	450 Km	598 Km	901 Km	975 Km	988 Km
Con Optimización	265 Km	424 Km	668 Km	810 Km	963 Km
% de Ahorro	70 %	41 %	35 %	20 %	2.6 %

Teniendo como referencia los costos generados por cada grupo de mantenimiento en función de su desplazamiento, alojamiento y alimentación, se puede cuantificar los porcentajes de ahorro determinados en las tablas anteriores. El combustible utilizado en el vehículo se contabiliza según el kilometraje y tiene un valor de 1,20 USD por cada 20 Km. Los costos aproximados que genera un grupo de mantenimiento diariamente durante su gestión son:

Tabla 13
Costos que genera un grupo de trabajo diariamente

Descripción	Costo	Cantidad	Total
Alojamiento	25 USD	3	75 USD
Alimentación	10 USD	3	30 USD
Vehículo	93 USD	1	93 USD

Tabla 14
Cuantificación del ahorro de las rutas con y sin optimización, según el tiempo de cada jornada

Costo Ahorrado en la ruta	20 Días	30 Días	40 Días	50 Días	60 Días
Por Viáticos	$ 792	$ 396	$ 990	$ 198	$ 594
Por Combustible	$ 11.1	$ 10.44	$ 13.98	$ 9.90	$ 1.5
Total	$ 803.1	$ 406.44	$ 1003.98	$ 207.9	$ 595.5

Para finalizar el análisis se presenta una proyección del ahorro en una jornada de 20 días que es la más comúnmente utilizada en la planificación de mantenimiento por los 10 grupos de trabajo que atienden las agencias de la institución financiera a nivel nacional.

Tabla 15
Costo ahorrado a Nivel Nacional con los 10 grupos de trabajo en una jornada de 20 días

Costo Ahorrado a Nivel Nacional	20 Días	N° de equipos	Total
Por Viáticos	$ 792	10	$ 7920
Por Combustible	$ 11.1	10	$ 111
Total	$ 803.1	10	**$ 8031**

El objetivo de la implementacion del software esta cumplido con exito, en el siguiente capitulo se terminará este trabajo concluyendo los resultados y generando recomendaciones para futuros trabajos.

Ahora este no es el final. No es ni siquiera el principio del fin. Pero es, quizá, el fin del principio.
Winston Churchill

CAPÍTULO
5

Conclusiones

- La utilización de redes neuronales artificiales, específicamente la Red Neuronal Hopfield para la resolución del cálculo de una ruta óptima en la planificación de mantenimiento y atención a nodos remotos y dispersos geográficamente ofrece una alternativa interesante y muy cercana a la solución ideal, debido a que las redes neuronales son capaces de hallar patrones en sus datos de entrada que les permiten predecir la solución óptima.

- La red neuronal Hopfield constituye una red neuronal artificial capaz de aplicarse a cualquier problema de optimización, pues por su estructura y su retroalimentación directa con su salida puede mejorar constantemente su respuesta optimizándola permanentemente.

- Este proyecto de optimización alcanzó su objetivo general el cual fue combinar técnicas de optimización e implementar un software capaz de optimizar las rutas de mantenimiento mediante el uso de una red neuronal, programación lineal, y la teoría de grafos.

- Para el buen desempeño y confiabilidad del funcionamiento de una red neuronal artificial es muy importante tomar en cuenta la importancia de la Calidad de los datos que son ingresados a la misma, es más importante tener una cantidad suficiente de datos en la entrada que sean confiables, y no una cantidad excesiva de información que no lo sea y que causará señales de ruido durante su implementación y entrenamiento.

- El momento de implementar y desarrollar una red neuronal artificial en un sistema computacional es necesario que se tome en cuenta la capacidad informática que se requiere tanto en memoria de procesamiento, como en capacidad y velocidad de cálculo.

- Las ventajas de la utilización de una red neuronal sobre otros métodos de cálculo radican en la confiabilidad y rapidez de respuesta que se puede obtener debido a que una red neuronal busca emular al comportamiento del cerebro humano, el cual se distingue por sobre las máquinas convencionales de cálculo en utilizar el razonamiento y asociación de sus neuronas, utilizando incluso la experiencia anteriores para poder resolver problemas de red cualitativos de manera muy sencilla entre estos el reconocimiento deserción elección y optimización de procesos.

- El desarrollo e implantación de redes neuronales en la actualidad se mantiene aún en estudio y cada día son más las aplicaciones en las cuales se puede utilizar las redes neuronales con resultados exitosos, sin que aún las mismas puedan completar el objetivo de trabajar con la perfección del cerebro humano.

- Durante el desarrollo de este proyecto se profundizó en los conceptos que enmarcan el diseño de una red neuronal Hopfield para la optimización de rutas de mantenimiento al igual que los conceptos que definen la programación lineal en la resolución del problema de Knapsack y la aplicación de la Teoría de Grafos en los modelos de la optimización específicamente el Problema del Agente Viajero, culminando con la obtención de un producto final como es la del software OPTIMA RUTA V1.0, el cual fue implantado sobre la plataforma de Matlab y permite obtener una ruta óptima para el mantenimiento y atención de nodos remotos y dispersos geográficamente.

- El software OPTIMA RUTA V1.0 puede ser instalado en cualquier ordenador que posea la plataforma Windows 7 o superior, para el ingreso de la información y el desarrollo de los cálculos se puede utilizar un proceso manual y / o la carga de una hoja de cálculo previamente desarrollada en el software Excel de Office. Sus interfaces gráficas le permiten al usuario visualizar los resultados numéricos y el gráfico de la ruta optimizado.

- Como desarrollo adicional se ha implementado una función add- in para **Microsoft®** Excel de office que puede ser instalado en cualquier computador con plataforma Windows 7 o superior y que trabajará directamente en una hoja de cálculo de Excel como una función adicional de este software.

- Este proyecto establece el inicio de varias aplicaciones futuras y desarrollos adicionales que pueden ejecutarse tanto en el área de las redes neuronales, como su aplicación para la industria en el desarrollo de respuestas a varios problemas prácticos que nos afecta constantemente y reducen la productividad, eficiencia y tiempos en procesos industriales y administrativos. Se sugiere continuar con el desarrollo e investigación en redes neuronales y sus aplicaciones.

Bibliografía

[Arredondo,2008] T. Arredondo , Introducción A Las Redes Neuronales , Universidad Técnica Federico Santa María , 7 - 64 , (2008) 10, 14, 15, 20, 21, 22

[Bartholdi,2008] J. J. Bartholdi , The Knapsack Problem , Springer , 20 - 31 , (2008) 29

[Basogain,1998] X. Basogain , Redes Neuronales Artificiales Y Sus Aplicaciones , Publicaciones De La Escuela De Ingenieros, Escuela Superior De Ingeniería De Bilbao , 13 - 59 , (1998) 6, 7, 11, 12, 13

[Bondy,1976] J. Bondy, U. Murty , Graph Theory And Applications , Elsevier Science Publishing Co. Inc. , 42 - 96 , (1976)

[Bruno,2013] T. Bruno , Pr0Blema De La Mochila , Universidad De Buenos Aires , 2 - 16 , (2013) 54

[Campos,1998] J. Campos - C.P.S. , Esquemas Algorítmicos - Programación Dinámica , , 7 - 16 40 - 47 , (2000) 17, 19

[Campos,1998] A. Campos , Proceso De Distribución Aplicando Redes Neuronales Artificiales Con Supervisión , Universidad Autonoma De Nuevo León , 31 - 67 , (1998) 17, 19

[De Castro,2002] F. De Castro, M.C. De Castro , Redes Neurais Artificiais , Pontificia Universidad Católica Do Rio Grande Do Sul , 1 - 25 , (2002) 10

[Espinoza,2006] D. Espinoza , Problema Del Vendedor Viajero (Tsp) Y Programación Entera (Ip) , Universidad De Chile , 6 - 140 , (2006) 5, 6

[Garrido,2000] A. Garrido, E. Onaindía , Un Algoritmo Para La Optimización De Rutas De Transporte , Universidad Politécnica De Valencia , 1 - 9 , (2000) 34

[Guerequeta,1998] R. Guerequeta, A. Vallecillo , Técnicas De Diseño De Algoritmos , Servicio De Publicaciones De La Universidad De Málaga , 177 - 209 , (1998)

[Hong,2005] Q. Hong, Y. Zhang, X. Xiaolin , Theoretical Analysis And Parameter Setting Of Hopfield Neural Networks , Springer - Verlag Berlín Heidelberg , 739 - 744 , (2005) 52

[Kiselev,2014] A. Kiselev, E. Zhukova , Introducción A La Teoría De Grafos , Universidad Estatal De San Petersburgo , 1 - 17 , (2014)

[Kleinberg ,2005] J. Kleinberg, E. Tardos , Algorithm Design , Pearson International Edition , 1 - 61 127 - 178 , (2005) 25, 26

[Lasarte,2017] E. Lasarte , Entrenamiento De Una Red Neuronal Hardware Desde Matlab (Hardware In The Loop) , Universidad Politécnica De Madrid , 3- 80 , (2017) 11, 14, 16

[Ledesma,2000] S. Ledesma , Las Redes Neuronales: Implementación Y Consideraciones Prácticas , Fifth Mexican International Conference On Artificial Intelligence , 1 - 17 , (2000)

[Mandziuk,2000] J. Mandziuk , Optimization With The Hopfield Network Based On Correlated Noises: Experimental Approach , International Computer Science Institute , 4 - 28 , (2000) 6

[Matich,2011] D. Matich , Redes Neuronales , Universidad Tecnológica Nacional de Rosario , 4 - 54 , (2011)

[Moya,2017] A. Moya , Problema De La Mochila Con Capacidad Variable , Universidad Miguel Hernández , 7 - 19 , (2017) 11, 14

[Novo,2004] E. Novo, A. Méndez , Aplicaciones De La Teoría De Grafos A Algunos Juegos De Estrategia , Suma: Revista Sobre Enseñanza Y Aprendizaje De Las Matemáticas , 31 - 36 , (2004) 55

[Ohlsson,1992] M. Ohlsson, C. Peterson, B. Söderberg , Neural Networks For Optimization Problems With Inequality Constraints - The Knapsack Problem , University Of Lund Sölvegatan , 1 -10 , (1992) 31, 33

[Pacheco,2003] J. Pacheco , Evolución De Indicadores Asociados A La Medicion De La Conectividad Y Utilidad De Las Redes De Transporte , Universidad Politécnica De Catalunya Barcelonatech , 26 - 48 , (2003) 6, 13, 22, 24

[Paredes,2011] J. Paredes , Teoría De Grafos: Estructuras Discretas , Universidad Alas Peruanas , 1 - 60 , (2011)

[Pérez De Vargas,2015] B. Pérez De Vargas , Resolución Del Problema Del Viajante De Comercio (Tsp) Y Su Variante Con Ventanas De Tiempo (Tsptw)U Usando Métodos Heurísticos De Búsqueda Local , Universidad De Valladolid , 5 -90 , (2015) 30, 31

[Pisinger,2005] D. Pisinger , Where Are The Hard Knapsack Problems? , Computer And Operations Research 32 , 2271 - 2284 , (2005) 3, 4

[Possani,2013] E. Possani , El Problema Del Agente Viajero: Un Recorrido Sobre Su Historia, Sus Aplicaciones Y Problemas Relacionados , Instituto Tecnológico Autónomo De México , 3 - 65 , (2013) 27, 28

[Potvin,1993] J. Potvin , The Traveling Salesan Problem: A Neural Network Perspective , Orsa Journal On Computing , 9 - 30 , (1993) 34, 35

[Quirós,2013] A. Quirós, M. López, L. Montiel, G. Martínez, A. Alcaraz , Modelo Matemático Del Problema Del Agente Viajero Para Encontrar La Ruta Óptima De Distribución , Investigación Gestión Organizacional , 9 - 26 , (2013) 8

[Rodríguez,2011] L. Rodríguez , Teoría De Redes O Grafos , Métodos Cuantitativos De Organización Industrial , 6 - 47 , (2011) 33

[Tan,2005] K. C. Tan, H. Tang, S. S. Ge , On Parameter Settings Of Hopfield Networks Applied To Traveling Salesman Problems , Ieee Transactions On Circuits And Systems—I: Regular Papers , 994 - 1001 , (2005) 7, 8

[Toro,2014] E. Toro, R. Bolaños, M. Granada , Solución Del Problema De Múltiples Agentes Viajeros Resuelto Mediante Técnicas Heurísticas , Universidad Tecnológica De Pereira , 174 - 181 , (2014) 53

[Vasantha ,2009] D. Vasantha Kalyani, R. Sundaramoorthy , Pattern Recognition Using Neural And Functional Networks , Springer - Verlag Berlín Heidelberg , 173 - 185 , (2009) 33

[Vázquez,2013] G. Vázquez , Aplicación De Algunas Heurísticas Al Problema De La Mochila , Benemérita Universidad Autónoma De Puebla , 1 - 119 , (2013) 24, 25

[Verdeguer,2016] D. Verdeguer , Solución En Paralelo Del Problema De La Mochila , Universitat Politécnica De Valencia , 13 - 36 , (2016)

[Wen,2008] Ue-Pyng Wen , Kuen-Ming Lan , Hsu-Shih Shih , A Review Of Hopfield Neural Networks For Solving Mathematical Progamming Problems , European Journal Of Operational Research , 1 - 14 , (2008)

[Wilson,1988] G. Wilson, G. Pawley , On The Stability Of The Travelling Salesman Problem Algorithm Of Hopfield And Tank , Springer - Verlag , 63 - 70 , (1988) 52

[Wolfe,1999] W. J. Wolfe , A Fuzzy Hopfield - Tank Traveling Salesman Problem Model , Informs Journal On Computing Vol 11 , 329 - 344 , (1999) 35
26, 27