TSP dinámico para la toma de decisiones en escenarios de e-commerce.

José Adrián Galván García
Departamento de Estudios de Posgrado e Investigación Tecnológico Nacional de México, Instituto Tecnológico de León
León, Guanajuato
M93240319@leon.tecnm.mx

Dr. Zamudio Rodríguez Víctor M.
Departamento de Estudios de Posgrado e Investigación Tecnológico Nacional de México, Instituto Tecnológico de León
León, Guanajuato
vic.zamudio@leon.tecnm.mx

Dr. Héctor José Puga Soberanes
Departamento de Estudios de Posgrado e Investigación Tecnológico Nacional de México, Instituto Tecnológico de León
León, Guanajuato
pugahector@yahoo.com

Dr. Juan Martín Carpio Valadez
Departamento de Estudios de Posgrado e Investigación Tecnológico Nacional de México, Instituto Tecnológico de León
León, Guanajuato
juanmartin.carpio@leon.tecnm.mx

Dr. Carlos Lino Ramírez
Departamento de Estudios de Posgrado e Investigación Tecnológico Nacional de México, Instituto Tecnológico de León
León, Guanajuato
carloslino@leon.tecnm.mx

***Resumen*— El problema del vendedor ambulante dinámico (DTSP) en un ambiente experimental es estocástico y dinámico. La capacidad de cambio requiere que el algoritmo tenga la capacidad de adaptarse rápidamente. La mayoría de los científicos llaman la atención sobre la correlación entre la diversidad de la población y la convergencia al óptimo. El control de la variación de la población permite el control de una convergencia estable del algoritmo al óptimo y proporciona un buen mecanismo para evitar el estancamiento.**

En este trabajo se hace un análisis del algoritmo genético y herramientas para la resolución del problema TSP dinámico, incluyendo las librerías de TSPLIB que indica cual es la ruta óptima para poderla evaluar los algoritmos propuesto como el genético, como estrategia para la solución del problema.

Palabras claves: TSP, DTSP, optimización

I. INTRODUCCION

El problema del agente viajero (TSP) es uno de los problemas más intensamente estudiados en la optimización combinatoria. Dado un conjunto de objetivos, la tarea consiste en determinar un recorrido más corto a cada objetivo de forma precisa y se reencuentra en el inicio. Si la asistencia entre dos objetivos cualesquiera es igual a la distancia euclidiana, el problema se denomina problema del vendedor ambulante euclidiano (ETSP). El problema del vendedor viajero asimétrico (ATSP) es un problema donde la distancia entre dos blancos no es simétrica, pero depende de la dirección del trasversal. Otras variaciones del TSP incluyen el Problema del vendedor ambulante con los vecindarios (TSPN), donde cada objetivo del recorrido puede moverse en una región determinada, el Problema del vendedor ambulante de cuello de botella (BTSP), donde debe ser minimizada la mayor distancia entre los dos objetivos en los itinerarios, y otros.

Los problemas de planificación de rutas enumerados anteriormente tienen como objetivo encontrar los mejores recorridos posibles sin tener en cuenta las características del vehículo. Sin embargo, cuando se trabaja con vehículos del mundo real, hay que tener en cuenta las restricciones cinemáticas, como el radio de giro mínimo. Los vehículos con restricciones de movimiento impuestas por el mecanismo de dirección satisfacen una restricción. Tales vehículos no son capaces de seguir caminos obtenidos de las soluciones de los problemas clásicos de TSP. Los robots móviles similares a automóviles o los vehículos aéreos de ala fija que avanzan a una velocidad constante y giran con curvatura acotada superior pueden modelarse como un vehículo El problema del vendedor ambulante para un vehículo se llama generalmente Problema del vendedor viajero (DTSP) o TSP Curvatura.

El DTSP ha atraído considerable atención debido a muchas aplicaciones civiles y militares. Una configuración típica es monitorear una colección de objetivos distribuidos espacialmente por un vehículo aéreo no tripulado (UAV). Esto podría referirse al control del tráfico en lugares específicos, la recopilación de inteligencia y el reconocimiento de objetivos sospechosos para operaciones antiterroristas, misiones de seguridad y vigilancia de infraestructuras críticas y otros puntos de interés, el apoyo a las misiones de combate mediante operaciones de inteligencia, vigilancia y reconocimiento, la evaluación de los daños de batalla (confirmación de un objetivo y verificación de su destrucción), entre otros. Para otras aplicaciones [8]. Normalmente, cuando tenemos un problema de optimización, podemos recurrir a dos tipos de algoritmos para solucionar el problema, los algoritmos exactos y los algoritmos heurísticos, para el primer caso, hay algoritmos que pueden encontrar la mejor solución (solución óptima) de manera determinista para dicho problema, la desventaja de este tipo de algoritmos

es que, en primera, suelen ser difíciles de modelar e implementar, ya que requieren de una capacidad analítica, además de conocimientos sobre matemáticas aplicadas y programación, otra desventaja de estos métodos exactos, es que estos exigen una cantidad de recursos computacionales considerable, sin embargo, a pesar de todas estas desventajas, utilizar este tipo de algoritmos siempre que sea factible es la mejor opción, por otro lado, puede haber casos donde definitivamente el tamaño del problema sea muy grande y no tengamos la posibilidad de utilizar un método exacto. Estando en esta situación, la única opción que tenemos es optar por alguna técnica que nos brinde una solución que muy posiblemente no sea la óptima pero que al menos nos asegure que la solución brindada sea una solución de calidad, es aquí cuando los algoritmos heurísticos toman relevancia.

II. METODOLOGIA

Algoritmo genético es utilizar la tecnología de búsqueda de población para resolver la población como un conjunto de problemas y generar una nueva generación de población mediante la aplicación de una serie de operaciones genéticas como la selección, el cruce y la mutación de factores ambientales genéticos biológicos similares a la población actual. Y optimizar gradualmente la población a un estado que contenga la solución óptima aproximada.

El TSP Dinámico (DTSP) fue introducido en 1998 por Psaraftis, DTSP es un TSP en el cual ciudades pueden ser agregadas o eliminadas en tiempo real, o cuando el costo de viajar entre ciudades puede cambiar. Algunos investigadores han trabajado el DTSP con optimización con colonia de hormigas (ACO, Ant Colony Optimization). El DTSP tratado desde un enfoque de optimización de colonia de hormigas tiene problemas con la evaporación de feromonas, incluso en el autor propone una variante del algoritmo ACO con una función de evaporación de feromonas diferente a la tradicional para poder adaptar este algoritmo a DTSP.

Un TSP Dinámico es un TSP determinado por una matriz de costos dinámica, como se muestra a continuación:

$$\boldsymbol{D(t) = d_{ij}(t)_{n\times n}} \quad (1)$$

d_{ij} *Es el costo de viajar de la ciudad* C_i *a la ciudad* C_j

t = *Es un tiempo determinado*

Donde $d_{ij}(t)$ is el costo desde la ciudad (nodo) c_i a la ciudad c_j, y t es el tiempo real. En esta definición, el número de ciudades $n(t)$ y el costo de la matriz son tiempos dependientes. El problema dinámico del agente viajero es encontrar el mínimo costo de la ruta que contiene todos los $n(t)$ nodos.

Po dados todos $n(t)$ $\left(P_1, P_2, \dots, P_{n(t)}\right)$ puntos, y la correspondencia de costo matriz $D = \{d_{ij}(t)\}, i, j = 1,2, \dots, n(t)$, encontrar un mínimo costo de la ruta que contiene todos los $n(t)$ puntos, donde t se encuentra en el momento del tiempo t se encuentra por la distancia entre el punto del objetivo P_i y el punto objetivo P_j, y $d_{ij}(t) = d_{ji}(t)$.

$$Min\left(d(T(t))\right) = \min\left(\sum_{i=1}^{n(t)} d_{T_i,T_{i+1}}(t)\right) \quad (1)$$

Donde $T \in \{1,2, \dots, n(t) if i \neq, then T_i \neq T_j, T_{n(t)+1} = T$
En la definición consideremos el cambio del DTSP de la matriz de costos con el tiempo continuo del proceso. Prácticamente discretizamos este proceso de cambio, Así, un DTSP se convierte en una serie de problemas de optimización.

$$D(t_k) = \{d_{ij}(t_k)\}_{n(t_k)\times n(t_k)} \quad (2)$$

$k = 0,1,2, \dots, m-1$, con la ventana de tiempo $[t_k, t_{k+1}]$, donde $\{t_k\}_{i=0}^{m}$ es una secuencia de puntos de puntos de muestreo de tiempo del mundo real.

III. EVALUACIÓN

Los algoritmos de AG son potentes herramientas para resolver problemas de TSP, y se tomara como base para resolver el problema del DTSP con la estrategia de almacenar la última ruta optima durante su trayecto introducirlo a la población y no forzar el algoritmo y obtener rápido el resultado.

Se utilizo el Algoritmo Genético con elitismo (SGA).

Algoritmo Pseudocódigo de un AG	
1:	/* Parámetros de un AG */
2:	t=0 // número de generación
3:	P (0)
4:	Evaluar P (0)
5:	While not (Condición de terminación) do
6:	P'(t) = seleccionar(P(t))
7:	P'(t) = aplicar operadores de cruce(P(t))
8:	P(t+1) = reemplazar (P(t), P'(t))
9:	Evaluar P(t+1)
10:	t=t+1
11:	return Mejor Solución Encontrada
12:	end while

A. Resultados

Nuestro algoritmo en pruebas en TSPLIB. (CPU: Intel core i7 RAM:16GB). La productividad fue evaluada en una

simulación en código Python 3.10.6, librería Turtle (gráficos) para evaluar los experimentos. Los experimentos son evaluados en el algoritmo genético para resolver el TSP dinámico.

Población 50, Probabilidad de cruza 0.9, Probabilidad de mutación 0.5, 2000 generaciones.

TABLE I. EVALUACIÓN DEL TSP, EXPERIMENTOS 33 VECES Y EL RESULTADO ES EL SIGUIENTE

TSPLIB	**Evaluación TSP**					
	Distancia Optima	***Distancia optima (AG)***	***Diferencia distancia optima y AG***	***Generación***	***Tiempo de ejecución***	***Tiempo Concord***
CH150	6528	6528	0	1491	42.7644	7.48
Kroa100	21282	21282	0	1963	43.0102	1.37
Kroc100	20749	20753	4	2000	45.9550	0.80
Lin105	14379	14392	13	2000	43.2640	0.80

TABLE II. EVALUACIÓN DEL TSP DINAMICO, EXPERIMENTO DEL RESULTADO ES EL SIGUIENTE

TSPLIB	**Evaluación TSP Dinámico**			
	Ch150	***Kroa100***	***Kroc100***	***Lin105***
Tiempo de inserción	155.0285	91.1992	93.2149	99.5681
Tiempo de borrado	38.5920	21.9458	20.9421	21.9743

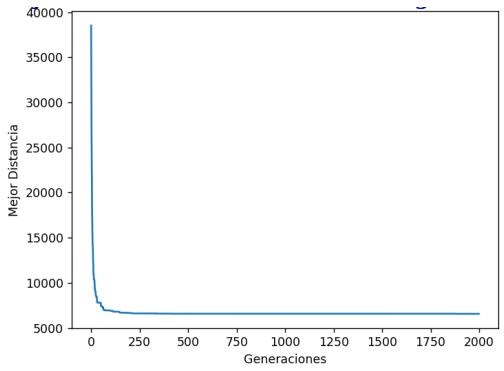

Fig. 1. Mejor distancia contra el número de generaciones

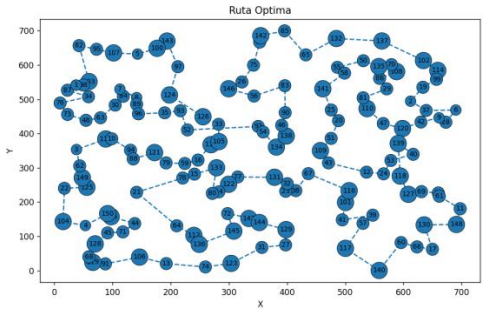

Fig. 2. Tsplib ch150.tsp, representa la ruta optima

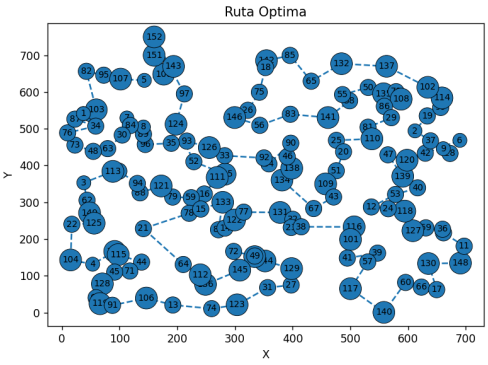

Fig. 3. Añadiendo 2 puntos P1(160,700) y P2(160,750

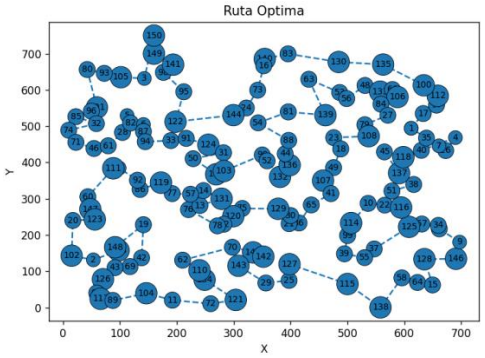

Fig. 4. Borrando los 2 primeros puntos P1(37.43935,541.20906) y P2(612.17595,494.31668)

IV. DISCUCION

Con la experimentación del algoritmo genético al finalizar las generaciones nos muestra la ruta optima y almacenando la ruta optima al momento de añadir o quitar elemento podemos definir que realizamos un menor esfuerzo en obtener la ruta porque tenemos ya una distancia menor que es lo que queremos encontrar , podemos obtener los puntos de las tiendas con su menor costo de los productos que estamos buscando, significa que si estamos buscando productos con distintos precios el precio más bajo nos representaría una

tienda y el recorrido nos mostrará el algoritmo genético con la ruta especificada ya que el comprador puede moverse las tiendas se presentan en un radio y se añaden los puntos en movimiento y se eliminan , se realizó una simulación el Python un radio fijo para tener controlado los puntos y movimientos al azar y se obtiene una respuesta optima de la ruta en movimiento, para que nos represente ya en la simulación se le especifica el tiempo de paro de las interacciones de las generaciones para que nos muestre la ruta optima a visitar las tiendas.

V. CONCLUSIONES Y/O PROYECTOS FUTUROS

En el presente trabajo se presenta el problema del agente viajero en el contexto de e-commerce. Este problema es importante porque nos permite identificar los puntos de venta de un producto que estamos buscando para indicarnos la ruta más corta para poderlo adquirir, ya que el agente se mueve los puntos de venta aparecerán dependiendo de la ruta donde se encuentre y se presenta el problema del agente viajero dinámico.

Se propuso el algoritmo genético con elitismo (SAG) para el TSP y el DTSP.

De la revisión del algoritmo genético podemos concluir que se encuentra la ruta más corta indicándonos los puntos donde se pueden visitar.

VI. AGRADECIMIENTO.

El autor agradece al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el apoyo económico brindado a través de becas de posgrado CVU 1109353. También al Tecnológico Nacional de México, en particular al Departamento de Posgrado e Investigación (DEPI) del Instituto Tecnológico de León donde se desarrolló este trabajo.

REFERENCIAS

[1] Babel, L. (2020). New heuristic algorithms for the Dubins traveling salesman problem. Journal of Heuristics, 1-28.

[2] Boryczka, U., & Strąk, Ł. (2015, March). Diversification and entropy improvement on the DPSO algorithm for DTSP. In Asian Conference on Intelligent Information and Database Systems (pp. 337-347). Springer, Cham.

[3] Chura, H. E. T., Delgado, C. A. S., Gonzales, E. E. A., & Espinoza, E. F. (2015). Aplicación del algoritmo de colonia de hormigas al problema del agente viajero. Ciencia & Desarrollo, (20), 98-102.

[4] Guntsch, M., & Middendorf, M. (2001, April). Pheromone modification strategies for ant algorithms applied to dynamic TSP. In Workshops on applications of evolutionary computation (pp. 213-222). Springer, Berlin, Heidelberg.

[5] Guntsch, M., Middendorf, M., & Schmeck, H. (2001, July). An ant colony optimization approach to dynamic TSP. In Proceedings of the 3rd annual conference on genetic and evolutionary computation (pp. 860-867).

[6] Restrepo, J. H., & Sánchez, J. J. (2004). Aplicación de la teoría de grafos y el algoritmo de Dijkstra para determinar las distancias y las rutas más cortas en una ciudad. Scientia et technica, 10(26), 121-126

[7] Riaño, E. R., Toro, G. M. M., & Rico-Bautista, D. (2018). Árbol de caminos mínimos: enrutamiento, algoritmos aproximados y complejidad. REVISTA COLOMBIANA DE TECNOLOGIAS DE AVANZADA (RCTA), 1(31), 11-21.

[8] S. J. Russell and P. Norvig, Inteligencia artificial. Un enfoque moderno. Madrid:Pearson, 2 ed., 2004.

[9] S. Dokania, S. Bagga and R. Sharma, "Opportunistic Self Organizing Migrating Algorithm for real-time Dynamic Traveling Salesman Problem," 2017 51st Annual Conference on Information Sciences and Systems (CISS), 2017, pp. 1-6, doi: 10.1109/CISS.2017.7926065.

[10] Smart Delivery systems , Solving complex vehicle Routing Problems, intelligent Data centric System, Series Fatos Xhafa, Edit by Jakub Nalepa, Ed. Elservier

[11] Talbi, E.-G. (2009). Metaheuristics: From design to implementation. Hoboken, N.J: John Wiley & Sons. Chura, H. E. T., Delgado, C. A. S., Gonzales, E. E. A., & Espinoza, E. F. (2015). Aplicación del algoritmo de colonia de hormigas al problema del agente viajero. *Ciencia & Desarrollo*, (20), 98-102.

[12] Riaño, E. R., Toro, G. M. M., & Rico-Bautista, D. (2018). Árbol de caminos mínimos: enrutamiento, algoritmos aproximados y complejidad. *REVISTA COLOMBIANA DE TECNOLOGIAS DE AVANZADA (RCTA)*, *1*(31), 11-21.

[13] Restrepo, J. H., & Sánchez, J. J. (2004). Aplicación de la teoría de grafos y el algoritmo de Dijkstra para determinar las distancias y las rutas más cortas en una ciudad. *Scientia et technica*, *10*(26), 121-126