[[1]](#footnote-1)

Implementación de la búsqueda de árbol de Monte Carlo en el problema del Agente Viajero (TSPTW)

Alfonso Caro Portillo, Estudiante de Maestría en Informática

Resumen – La teoría de la complejidad computacional, determina y se centra su estudio en problemas que requieren implementar algoritmos que buscan realizar una o más computaciones o cálculos para resolver dichos problemas, en muchos casos estas soluciones exigen una capacidad de recursos como CPU y memoria que limitan su implementación, presentar soluciones que optimicen los recursos y multipliquen inteligentemente las capacidades de cómputo actuales, en este documento encontraremos el análisis e implementación de una heurística que busca dar solución al clásico problema del agente viajero usando los principios de la complejidad computacional.

**Índice de Términos – Agente Viajero, Árbol de Búsqueda de Monte Carlo, Heurística, complejidad computacional**

# introducción

Al diseñar, desarrollar e implementar una solución orientada a crear un algoritmo que dé respuesta a problemas complejos, supone un especial cuidado en consideraciones de tiempo y espacio, esto se da al conocer de antemano si la respuesta dada por el algoritmo solución tardara un segundo, una hora o un año, de igual forma saber que la memoria requerida para la solución estará disponible o si por el contrario se requiere memoria casi infinita para su implementación, esto con el fin de buscar soluciones optimas que permitan acotar las restricciones presentadas y generar un “buen” algoritmo, para esto se han generado Heurísticas y metaheurísticas que contribuyen a validar las consideraciones sobre complejidad y están vinculadas con las estructuras de datos especiales que se pueden incorporar en diversas soluciones. Dedicaremos este documento al análisis de una Heurística conocida como el árbol de búsqueda anidada de Monte Carlo para encontrar soluciones optimas al problema del agente viajero en su variación con restricciones de ventanas de tiempo y determinar cómo los principios de esta heurística se pueden aplicar a un problema complejo conocido.

El problema del viajante con las ventanas de tiempo es conocido

Como un punto de referencia particularmente complejo para los algoritmos de optimización. En este problema, nos interesa la minimización del costo del viaje. Para solucionar este problema, se propone utilizar el algoritmo de Monte-Carlo anidado combinado con una estrategia de evolución y la autoadaptación. El objetivo es determinar como la heurística seleccionada puedes ser aplicada en esta solución y para muchos problemas en un corto período de tiempo.

Dentro de los algoritmos de aproximación y búsqueda se encuentran la metaheurísticas, las cuales son capaces de encontrar la solución a problemas muy complejos, inicialmente se presentan un gran número de posibles soluciones y mediante las interacciones y recorridos continuos se van optimizando basado en la evaluación de los resultados obtenidos con el objetivo de ir reduciendo el número de soluciones posibles y generar una solución óptima, esto basado en los principios principales como lo son resolver problemas rápidamente, resolver problemas grandes, producir algoritmos estables y que simultáneamente sean fáciles de implementar y que se puedan adaptar de forma fácil a muchos problemas.

# PROBLEMA DEL AGENTE VIAJERO

El problema del viajante es un problema logístico importante. Se utiliza para representar el problema de encontrar una ruta eficiente para visitar un cierto número de clientes, comenzando y terminando en un mismo punto. Existe una variante que adiciona una restricción con ventanas de tiempo y agrega la dificultad de que cada cliente debe ser visitado dentro de un período de tiempo determinado.

El objetivo es minimizar la duración del viaje. Este problema es un problema NP difícil e incluso encontrar una solución factible es NP-completo, inicialmente las primeras aproximaciones de solución ​​se basaron en ramificar y enlazar cada uno de los puntos y determinar sus costos individuales y la sumatoria global de los costos asociados. Posteriormente, Dumas. utilizó un método basado en la programación dinámica. Más recientemente, se han propuesto métodos basados ​​en la programación de restricciones. También se han considerado algoritmos basados ​​en heurística como el que analizaremos en este documento. Y existen grupos e instituciones donde se proporciona un estudio completo de los métodos más eficientes para resolver el TSPTW donde propone nuevos algoritmos como por ejemplo basados en colonias de hormigas que logra muy buenos resultados, Algoritmos Genéticos, redes neuronales entre otros que proporcionan un entorno claro para comparar algoritmos y soluciones con diferentes recursos y características de implementación donde el objetivo final en validar estas soluciones y mejorar las implementaciones.

Aplicaciones del problema del agente viajero

El problema del agente viajeros busca como fin último encontrar un camino que recorra un grupo de nodos, comenzando y finalizando en el mismo nodo (0), minimizando la suma de tiempos de espera y distancias recorridas hasta cada uno de los nodos y adicionalmente TSPTW adiciona un valor en términos de tiempo para llegar a cada uno de los nodos sin incumplir esta restricción. Este es un problema de optimización simple que se puede encontrar en múltiples situaciones de la vida real, por ejemplo:

#### Redes de computadoras, el problema de ruteo tiene aplicaciones en la búsqueda de información en una red determinada. Se puede visualizar como solución para recorrer los puntos donde se encuentra la información deseada y hallar las rutas que minimicen el tiempo de búsqueda.

#### Se emplea con mucha frecuencia en la diagramación de rutas de entregas para diferentes tipos de mercancías a múltiples destinos con su restricción de adicionar muchos vehículos a la ecuación.

#### Recientemente se han establecido algoritmos para vehículos no tripulados (Drones) donde se deben cumplir metas parciales y donde los algoritmos de solución cobran un papel mas importante ya que las metas a cumplir se hacen totalmente automatizadas, creando cálculos en tiempo real para resolver este tipo de situaciones.

SOLUCIONES PARA TSPTW

La importancia de hallar soluciones “buenas” a problemas complejos de optimización y especialmente a problemas de optimización combinatoria es alta por su implementación en campos industriales y científicas, y por esto de han desarrollado múltiples métodos para resolverlos, en la cuales podemos encontrar técnicas exactas, exhaustivas y aproximadas.

Las técnicas exactas hallan soluciones optimas para una instancia de un problema en un tiempo determinado, pero en el caso de los problemas NP-difíciles los tiempos de solución a nivel computacional son muy difíciles de calcular ya que estos tiempos crecen de forma exponencial con las variables que intervienen en el problema es decir que en cuanto el modelo de solución es iterado se hace necesarios recursos computacionales de forma exponencial para la búsqueda de la solución, lo que muchas veces incrementa los costos asociados al implementar estas técnicas y los tiempos se hacen inviables para su implementación.

Para estos casos se han desarrollado técnicas de aproximación donde se pueden encontrar soluciones factibles que pueden ser una buena aproximación a la solución optima esperada y donde la inversión en tiempo y recurso son mas razonables para la implementación de las computaciones asociadas.

Para las técnicas de solución aproximadas encontramos las heurísticas que son técnicas de indagación y descubrimiento mediante métodos no rigurosos y donde es factible encontrar buenas soluciones a los modelos de optimización. Estas a su vez se clasifican en heurísticas de construcción y de búsqueda. Por el lado de las metaheurísticas son métodos de aproximación especialmente creados para la solución de problemas de optimización combinatoria, donde se encuentran soluciones “optimas” muy acertadas utilizando procesos iterativos que orientan la heurística en un rango de soluciones factibles empleando principios de inteligencia artificial o procesos bilógicos naturales como la evolución o el comportamiento animal.

Figura 1. Clasificación de las principales técnicas de optimización

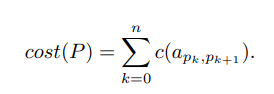
TSPTW se puede describir de la siguiente manera. Sea G un Grafo no dirigido.

G = (N, A)

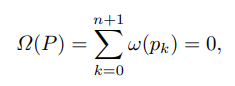
donde N = 0, 1, ..., n es un conjunto de nodos y A = N ∗ N es el conjunto de bordes entre los nodos. El nodo 0 representa el depósito. Los otros n nodos representan a los clientes.

Se da una función de costo c: A → R. la cual representa la distancia entre 2 clientes. Una solución a este problema es una secuencia de nodos.

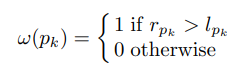
P = (p0, p1, ..., pn, pn + 1) donde p0 = pn + 1 = 0 y (p1, ..., pn) es una permutación de N \ {0}. El objetivo es minimizar la función:



En la versión con ventanas de tiempo, cada cliente i está asociado a un intervalo **[ei, li].** El cliente no debe ser atendido antes de **ei** o después de **li**. Se permite llegar a un nodo i antes de **ei** pero la hora de salida se convierte en **ei.** Sea **dpk** la hora de salida del nodo **pk, dpk = max (rpk, epk)** donde **rpk** es la hora de llegada al nodo **pk**. La función para minimizar es la misma, pero ahora se debe respetar un conjunto de restricciones. Sea Ω (P) el número de restricciones de ventanas violadas por el tour P. La optimización de f debe realizarse respetando la siguiente ecuación:



Donde:



Tenemos que:



Con la adición de las restricciones, el problema se vuelve mucho más complicado y los algoritmos clásicos usados para TSP ya no son eficientes. Es por eso que usaremos búsqueda de Monte-Carlo que se describe en la siguiente parte del documento.

implementación del árbol de búsqueda de monte carlo

El árbol de búsqueda Monte Carlo (MCTS) es un algoritmo de búsqueda heurístico para algunos tipos de proceso de toma de decisiones, sobre todo los que trabajan con juegos.

Un ejemplo destacado reciente es en los programas Go,​ y también se ha utilizado en otros juegos de mesa, así como en videojuegos en tiempo real y juegos no deterministas como el póquer.

El enfoque del árbol de búsqueda Monte Carlo se encuentra en el análisis de los movimientos más prometedores, ampliando el árbol de búsqueda basado en un muestreo aleatorio del espacio de búsqueda. La aplicación de búsqueda de árbol de Monte Carlo en los juegos se basa en muchos playoffs. En cada emisión, el juego, se juega de salida hasta el final mediante la selección de movimientos al azar.

El resultado final del juego de cada playout se utiliza para ponderar los nodos en el árbol del juego de manera que los mejores nodos son más propensos a ser elegidos en futuros playoffs.

La forma más básica de utilizar los playouts es aplicar el mismo número de playouts después de cada movimiento legal del jugador actual, a continuación, elegir el movimiento que llevó a la mayor cantidad de victorias.​ La eficacia de este método llamado Búsqueda Pura de Juego Monte Carlo - a menudo aumenta con el tiempo a medida que más playouts se asignan a los movimientos que han dado lugar con frecuencia a la victoria del jugador (en playouts anteriores). La plena búsqueda de árbol de Monte Carlo emplea este principio de forma recursiva en muchas profundidades del árbol de juego. Cada ronda de búsqueda de árbol de Monte Carlo consiste en cuatro pasos:

1. ***Selección:*** *empezar desde la raíz R y seleccionar nodos hijos sucesivos hasta alcanzar un nodo hoja L. La sección de abajo describe más de una manera de elegir nodos hijos, que permiten que el árbol de juego se expanda hacia movimientos más prometedores, que es la esencia del árbol de búsqueda Monte Carlo.*
2. ***Expansión:*** *a menos que L termine el juego con una victoria/pérdida para cualquiera de los jugadores, crear uno o más nodos hijos y elegir entre ellos un nodo C. Los nodos hijos son cualquier movimiento válido desde la posición del juego definida por L.*
3. ***Simulación****: jugar una reproducción aleatoria desde el nodo C.*
4. ***Retropropagación****: utilizar el resultado de la reproducción para actualizar la información en los nodos en el camino de C a R.*

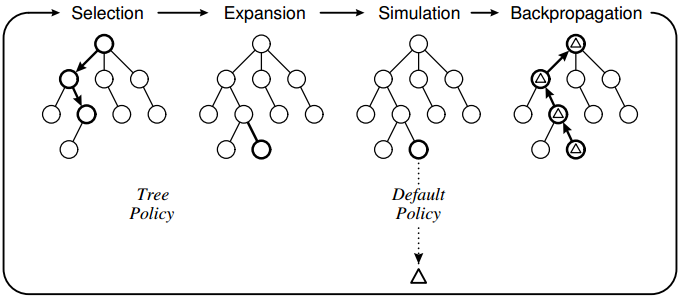


Figura 2, diagrama de implementación de la búsqueda de Montecarlo en un grafo.

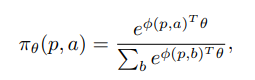
El algoritmo de Monte-Carlo (MCTS) es un algoritmo de búsqueda de árbol. Se supone que el árbol es grande y se pueden evaluar las hojas del árbol (posiciones finales del problema). No requiere ningún conocimiento sobre el problema y es bastante sencillo de implementar.

Es particularmente eficaz en problemas en los que las decisiones posteriores son tan importantes como las primeras. MCTS ha permitido establecer récords mundiales en juegos para un jugador como Morpion Solitaire o SameGame. Primero describimos el algoritmo MCTS y luego explicamos cómo introdujimos la heurística para obtener mejores resultados en el problema TSTPW.

El algoritmo de Monte-Carlo anidado. El algoritmo MCTS utiliza varios niveles. Cada nivel utiliza el nivel inferior para determinar qué acción se seleccionará en cada paso. El nivel 0 es una simulación de Montecarlo, es decir, una selección aleatoria de acciones hasta alcanzar una posición final. Más precisamente, en cada posición, una búsqueda de NMC de nivel n realizará una MCTS de nivel n - 1 para cada acción y luego seleccionará la que tenga la mejor puntuación. Por ejemplo, una búsqueda MCTS de nivel 1 hará una simulación de Monte-Carlo para cada acción (aquellas que alcanzan una posición final que se puede evaluar) y seleccionará la acción asociada con la evaluación más alta. Una vez que se ha seleccionado una acción, el problema está en una nueva posición y el método de selección se repite nuevamente hasta que se alcanza una posición final. El rendimiento del algoritmo se mejora enormemente al memorizar la mejor secuencia para cada nivel.

Esto proporciona un buen compromiso entre exploración y explotación. Es particularmente eficiente para juegos de un jugador y da buenos resultados incluso sin conocimientos de dominio. Sin embargo, los resultados se pueden mejorar mediante la adición de heurísticas.

Para la adaptación del problema del viajero con ventanas de tiempo. Es posible mejorar el rendimiento de MCTS modificando las simulaciones de Monte-Carlo. Una forma eficiente es seleccionar acciones basadas en heurísticas en lugar de una distribución uniforme. Sin embargo, se debe mantener cierta aleatoriedad para preservar la diversidad de las simulaciones. Para hacer eso, usamos una política Softmax de Boltzmann. Esta política se define por la probabilidad πθ (p, a) de elegir la acción a en una posición p:



Donde φ (p, a) es un vector de características y θ es un vector de ponderaciones de características. Las características que usamos son las heurísticas descritas en:

1. la distancia al último nodo:



1. la cantidad de tiempo que será necesario esperar si se selecciona a debido al comienzo de su ventana de tiempo:



1. la cantidad de tiempo que queda hasta el final de la ventana de tiempo de a si se selecciona a:



donde d es el último nodo seleccionado en la posición p, Tp es el tiempo utilizado para llegar a la situación p, ea es el comienzo de la ventana de tiempo para la acción a, la es el final de la ventana de tiempo para la acción a y c (d, a) es el viaje costo entre d y a. Los valores de la heurística se normalizan antes de usarse. Los valores que optimizaremos son los valores del vector θ (los pesos de las características).

References

1. G. O. Young, “Synthetic structure of industrial plastics (Book style with paper title and editor),” in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.
2. W.-K. Chen, *Linear Networks and Systems* (Book style)*.* Belmont, CA: Wadsworth, 1993, pp. 123–135.
3. H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.
4. B. Smith, “An approach to graphs of linear forms (Unpublished work style),” unpublished.
5. E. H. Miller, “A note on reflector arrays (Periodical style—Accepted for publication),” *IEEE Trans. Antennas Propagat.*, to be published.
6. J. Wang, “Fundamentals of erbium-doped fiber amplifiers arrays (Periodical style—Submitted for publication),” *IEEE J. Quantum Electron.*, submitted for publication.
7. C. J. Kaufman, Rocky Mountain Research Lab., Boulder, CO, private communication, May 1995.
8. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interfaces(Translation Journals style),” *IEEE Transl. J. Magn.Jpn.*, vol. 2, Aug. 1987, pp. 740–741 [*Dig. 9th Annu. Conf. Magnetics* Japan, 1982, p. 301].
9. M. Young, *The Techincal Writers Handbook.* Mill Valley, CA: University Science, 1989.
10. J. U. Duncombe, “Infrared navigation—Part I: An assessment of feasibility (Periodical style),” *IEEE Trans. Electron Devices*, vol. ED-11, pp. 34–39, Jan. 1959.
11. S. Chen, B. Mulgrew, and P. M. Grant, “A clustering technique for digital communications channel equalization using radial basis function networks,” *IEEE Trans. Neural Networks*, vol. 4, pp. 570–578, July 1993.
12. R. W. Lucky, “Automatic equalization for digital communication,” *Bell Syst. Tech. J.*, vol. 44, no. 4, pp. 547–588, Apr. 1965.
13. S. P. Bingulac, “On the compatibility of adaptive controllers (Published Conference Proceedings style),” in *Proc. 4th Annu. Allerton Conf. Circuits and Systems Theory*, New York, 1994, pp. 8–16.
14. G. R. Faulhaber, “Design of service systems with priority reservation,” in *Conf. Rec. 1995 IEEE Int. Conf. Communications,* pp. 3–8.
15. W. D. Doyle, “Magnetization reversal in films with biaxial anisotropy,” in *1987 Proc. INTERMAG Conf.*, pp. 2.2-1–2.2-6.
16. G. W. Juette and L. E. Zeffanella, “Radio noise currents n short sections on bundle conductors (Presented Conference Paper style),” presented at the IEEE Summer power Meeting, Dallas, TX, June 22–27, 1990, Paper 90 SM 690-0 PWRS.
17. J. G. Kreifeldt, “An analysis of surface-detected EMG as an amplitude-modulated noise,” presented at the 1989 Int. Conf. Medicine and Biological Engineering, Chicago, IL.
18. J. Williams, “Narrow-band analyzer (Thesis or Dissertation style),” Ph.D. dissertation, Dept. Elect. Eng., Harvard Univ., Cambridge, MA, 1993.
19. N. Kawasaki, “Parametric study of thermal and chemical nonequilibrium nozzle flow,” M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.
20. J. P. Wilkinson, “Nonlinear resonant circuit devices (Patent style),” U.S. Patent 3 624 12, July 16, 1990.
21. *IEEE Criteria for Class IE Electric Systems* (Standards style)*,* IEEE Standard 308, 1969.
22. *Letter Symbols for Quantities*, ANSI Standard Y10.5-1968.
23. R. E. Haskell and C. T. Case, “Transient signal propagation in lossless isotropic plasmas (Report style),” USAF Cambridge Res. Lab., Cambridge, MA Rep. ARCRL-66-234 (II), 1994, vol. 2.
24. E. E. Reber, R. L. Michell, and C. J. Carter, “Oxygen absorption in the Earth’s atmosphere,” Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0200 (420-46)-3, Nov. 1988.
25. (Handbook style) *Transmission Systems for Communications,* 3rd ed., Western Electric Co., Winston-Salem, NC, 1985, pp. 44–60.
26. *Motorola Semiconductor Data Manual,* Motorola Semiconductor Products Inc., Phoenix, AZ, 1989.
27. (Basic Book/Monograph Online Sources) J. K. Author. (year, month, day). *Title* (edition) [Type of medium]. Volume(issue). Available: <http://www.(URL>)
28. J. Jones. (1991, May 10). Networks (2nd ed.) [Online]. Available: <http://www.atm.com>
29. (Journal Online Sources style) K. Author. (year, month). Title. *Journal* [Type of medium]. Volume(issue), paging if given. Available: <http://www.(URL>)
30. R. J. Vidmar. (1992, August). On the use of atmospheric plasmas as electromagnetic reflectors. *IEEE Trans. Plasma Sci.* [Online]. *21(3).* pp. 876—880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar

**Biografía Autor(es)** (M'76-SM'81-F'87) y los otros autores pueden incluir las biografías al final de los documentos(papers) regulares. Por favor incluyan nombres y apellidos con los cuales puedan ser identificados al registrar sus artículos (Son registrados en la base de Publindex en Colciencias, entre otras). Si ha enviado documentos antes, no debe suponer que el Comité Editorial conoce o puede decidir cuál es o cuáles son los autores del artículo. *Cada envío debe ser completo en todos sus datos*. El primer párrafo debe contener la filiación institucional (por ejemplo, profesor asociado, Facultad de Ingeniería de Sistemas, Universidad El Bosque). Los grados deben listarse con el tipo de grado, en qué campo, en que institución, ciudad, estado o país.

El segundo párrafo usa el pronombre de la persona (él o ella) y no el apellido o nombre del autor. Lista la experiencia académica y laboral. Se ponen en mayúscula los títulos del trabajo. Pueden listarse cargos anteriores. Información que involucra las publicaciones anteriores puede ser incluida. Intente no listar más de tres libros o artículos publicados. El formato para listar a publicadores de un libro dentro de la biografía es: el título de libro (la ciudad, estado: el nombre del publicador, año) similar a una referencia. Los intereses de investigaciones actuales y anteriores terminan el párrafo.

El tercer párrafo empieza con el título del autor y apellido (por ejemplo, Dr. Smith, Prof. Jones, Sr. Kajor, Ms. Hunter). Finalmente, liste cualquier premio por trabajos y publicaciones. Proporcionar una fotografía es requisito para publicar su artículo: la biografía se dentará alrededor de ella. La fotografía se pone en la esquina superior izquierda de la biografía. Se quitarán las aficiones personales de la biografía.

1. [↑](#footnote-ref-1)