CIENTIFICA
TechNoLogica
ISSN:3119-7906, Vol.1Núm.2(2026): Abril-Junio
https://doi.org/10.17268/tech1.2.2026.02

Algoritmo de Prim Para la Obtención de Árboles de Expansión Mínima en Grafos Ponderados

Prim's Algorithm for Obtaining Minimum Spanning Trees in Weighted Graphs

Diaz Rodriguez Bagner Ivars[1], Gonzales Mondragon RuthLizeth[1], Pérez Gonzalez Alessandro Kathriel[1], Ramírez Quispe Elmer Joel[1], Sánchez Sangama Marco Antonio[1]

[1] Escuela de Ingeniería Informática, Universidad Nacional de Trujillo, La Libertad, Perú

*Autor correspondiente: bidiazr@unitru.edu.pe (Bagner. D)

RESUMEN

Los algoritmos voraces constituyen una de las estrategias algorítmicas más utilizadas para resolver problemas de optimización en informática. Entre ellos, el algoritmo de Prim destaca por su capacidad para construir árboles de expansión mínima en grafos ponderados conectados. El objetivo de esta investigación es analizar el funcionamiento, aplicación y eficiencia del algoritmo de Prim dentro del contexto de las estrategias algorítmicas. La metodología empleada consistió en una revisión bibliográfica de literatura especializada y en el análisis de ejemplos prácticos aplicados a grafos ponderados. Los resultados preliminares muestran que el algoritmo permite obtener soluciones óptimas mediante la selección iterativa de aristas de menor costo, garantizando la conexión de todos los vértices sin generar ciclos. Asimismo, se evidencia su importancia en problemas relacionados con diseño de redes, telecomunicaciones y sistemas de transporte. Se concluye que el algoritmo de Prim representa una alternativa eficiente para la construcción de árboles de expansión mínima, especialmente en grafos densos donde la minimización de costos resulta fundamental.

Palabras clave: Algoritmo de Prim, estrategias algorítmicas, grafos ponderados, árbol de expansión mínima, algoritmos voraces.

ABSTRACT

Greedy algorithms constitute one of the most widely used algorithmic strategies for solving optimization problems in computer science. Among them, Prim's algorithm stands out for its ability to construct minimum spanning trees in connected weighted graphs. The objective of this research is to analyze the operation, application, and efficiency of Prim's algorithm within the context of algorithmic strategies. The methodology consisted of a bibliographic review of specialized literature and the analysis of practical examples applied to weighted graphs. Preliminary results show that the algorithm obtains optimal solutions through the iterative selection of minimum-cost edges, guaranteeing the connection of all vertices without generating cycles. Furthermore, its importance is evident in problems related to network design, telecommunications, and transportation systems. It is concluded that Prim's algorithm represents an efficient alternative for constructing minimum spanning trees, especially in dense graphs where cost minimization is essential.

Keywords: *Prim Algorithm, algorithmic strategies, weighted graphs, minimum spanning tree, greedy algorithms.*

I. INTRODUCCIÓN

Las estrategias algorítmicas constituyen un conjunto de técnicas utilizadas para diseñar soluciones eficientes a problemas computacionales. Dentro de estas estrategias, los algoritmos voraces ocupan un lugar importante debido a que toman decisiones óptimas locales en cada paso con la expectativa de alcanzar una solución global óptima.

Uno de los algoritmos más representativos de esta categoría es el algoritmo de Prim, desarrollado para resolver el problema del árbol de expansión mínima (Minimum Spanning Tree, MST). Este problema consiste en conectar todos los vértices de un grafo ponderado mediante el conjunto de aristas cuyo costo total sea mínimo, evitando la formación de ciclos.

La relevancia del algoritmo de Prim se observa en numerosas aplicaciones prácticas, tales como el diseño de redes de comunicación, distribución eléctrica, planificación de carreteras, redes de transporte y sistemas de cableado informático. En estos escenarios, la reducción de costos constituye un factor determinante para la optimización de recursos.

El algoritmo inicia seleccionando un vértice cualquiera y posteriormente incorpora de manera iterativa la arista de menor peso que conecte un vértice perteneciente al árbol parcial con otro vértice aún no visitado. Este proceso continúa hasta que todos los nodos quedan conectados formando un árbol de expansión mínima.

La presente investigación tiene como objetivo analizar el funcionamiento del algoritmo de Prim, describir sus fundamentos teóricos, evaluar su eficiencia computacional y estudiar su aplicación en problemas de optimización representados mediante grafos ponderados.

II. MATERIALES Y METODOS

A. Tipo de Investigación

La investigación corresponde a un estudio descriptivo y analítico basado en la revisión documental de fuentes bibliográficas especializadas en algoritmos, estructuras de datos y teoría de grafos.

B. Materiales Utilizados

Para el desarrollo del estudio se emplearon:

- Biografía especializada sobre diseño y análisis de algoritmos.
- Documentación académica relacionada con teoría de grafos.
- Material del curso Estrategias Algorítmicas de la Universidad Nacional de Trujillo.
- Ejemplos de grafos ponderados para el análisis del algoritmo.

C. Procedimiento Metodológico

La investigación se desarrolló mediante las siguientes etapas:

- Revisión de literatura relacionada con algoritmos voraces.
- Estudio de los fundamentos matemáticos de los árboles de expansión mínima.
- Análisis del algoritmo de Prim y de su procedimiento de ejecución.
- Aplicación del algoritmo sobre ejemplos de grafos ponderados.
- Evaluación de resultados obtenidos y análisis de eficiencia.

D. Funcionamiento del Algoritmo de Prim

El algoritmo de Prim pertenece a la categoría de algoritmos voraces. Su idea principal consiste en construir progresivamente un árbol de expansión mínima agregando nodos uno a uno mediante la selección de la arista de menor peso disponible.

El procedimiento general puede resumirse en los siguientes pasos:

- Paso 1. Seleccionar un vértice inicial.
- Paso 2. Marcar el vértice como visitado.

- Paso 3. Identificar todas las aristas que conectan vértices visitados con vértices no visitados.
- Paso 4. Seleccionar la arista de menor peso.
- Paso 5. Incorporar la arista y el nuevo vértice al árbol.
- Paso 6. Repetir el proceso hasta incluir todos los vértices.

III. RESULTADOS

A. Análisis Conceptual

El estudio realizado permitió verificar que el algoritmo de Prim genera árboles de expansión mínima mediante una estrategia de crecimiento progresivo. A diferencia del algoritmo de Kruskal, Prim trabaja expandiendo un único árbol desde un nodo inicial, seleccionando continuamente la arista más económica disponible. La principal ventaja observada es que cada decisión local contribuye directamente a la construcción de una solución global óptima, siempre que el grafo sea conexo y ponderado.

B. Aplicación Practica

Se consideró un grafo ponderado compuesto por varios vértices conectados mediante aristas con diferentes costos. El algoritmo inició desde un nodo seleccionado arbitrariamente y fue incorporando sucesivamente las aristas de menor peso.

Durante cada iteración se verificó que:

- No se generan ciclos.
- Todos los nodos permanecen conectados.
- El costo total acumulado es mínimo.

Los resultados preliminares evidencian que el algoritmo logra construir un árbol de expansión mínima eficiente mediante un número reducido de operaciones de selección.

C. Complejidad Computacional

La eficiencia del algoritmo de Prim depende en gran medida de la estructura de datos utilizada durante su implementación. Cuando se emplea una matriz de adyacencia para representar el grafo, la complejidad temporal del algoritmo es $O(V^2)$, donde V representa el número de vértices. Esta implementación resulta adecuada para grafos densos en los que existe una gran cantidad de conexiones entre nodos.

Por otro lado, cuando se utilizan listas de adyacencia junto con colas de prioridad basadas en montículos binarios, la complejidad temporal puede reducirse a $O(E \log V)$, donde E representa el número de aristas del grafo. Esta optimización permite aplicar el algoritmo en problemas de gran escala manteniendo tiempos de ejecución aceptables.

La complejidad espacial del algoritmo está determinada principalmente por la estructura de almacenamiento del grafo y por los arreglos auxiliares utilizados para registrar los vértices visitados y las aristas seleccionadas.

D. Ventajas y Desventajas del Algoritmo de Prim

Entre las principales ventajas del algoritmo de Prim se encuentran:

- Garantiza la obtención de un árbol de expansión mínima óptimo.
- Presenta una implementación relativamente sencilla.
- Funciona eficientemente en grafos densos.
- Tiene aplicaciones prácticas en diversas áreas de la ingeniería y la informática.
- Reduce significativamente los costos de conexión en problemas de redes.

Sin embargo, también presenta algunas limitaciones:

- Requiere que el grafo sea conexo para obtener una solución completa.
- Su rendimiento depende de la estructura de datos utilizada.

- No puede aplicarse directamente a problemas que no puedan modelarse mediante grafos ponderados.
- En algunos casos puede ser superado por otros algoritmos especializados dependiendo de las características del problema

E. Aplicaciones del Algoritmo de Prim

El algoritmo de Prim posee numerosas aplicaciones en problemas reales donde se requiere minimizar costos de conexión entre distintos puntos.

Una de sus aplicaciones más conocidas se encuentra en el diseño de redes de telecomunicaciones. En este contexto, el algoritmo permite determinar la forma más económica de conectar estaciones, antenas o centros de distribución utilizando la menor cantidad posible de recursos.

También es utilizado en sistemas de distribución eléctrica para planificar tendidos eléctricos minimizando costos de instalación. De manera similar, puede emplearse en redes de distribución de agua potable y gas natural.

En el ámbito del transporte, el algoritmo facilita el diseño de carreteras, vías ferroviarias y rutas logísticas que conectan múltiples localidades minimizando los costos de construcción o mantenimiento.

En ingeniería informática se aplica al diseño de redes LAN, cableado estructurado y optimización de conexiones entre servidores y dispositivos de red.

IV. DISCUSIÓN

Los resultados obtenidos demuestran que el algoritmo de Prim constituye una herramienta eficaz para la construcción de árboles de expansión mínima. La estrategia voraz empleada permite seleccionar en cada iteración la arista de menor peso disponible, generando progresivamente una solución óptima.

Durante el desarrollo del caso práctico se comprobó que todas las decisiones locales tomadas por el algoritmo contribuyeron a la construcción de una solución global mínima. El árbol generado logró conectar todos los vértices del grafo sin producir ciclos y manteniendo el menor costo total posible.

Los resultados observados coinciden con los planteamientos teóricos desarrollados en la literatura especializada sobre teoría de grafos y algoritmos voraces. Asimismo, se confirma que el algoritmo resulta especialmente eficiente cuando se trabaja con grafos densos y cuando se implementa utilizando estructuras de datos optimizadas.

En comparación con otros métodos como el algoritmo de Kruskal, Prim presenta ventajas cuando el número de aristas es elevado, debido a que construye un único árbol en crecimiento continuo en lugar de procesar todas las aristas de forma independiente.

V. CONCLUSIÓN

- El algoritmo de Prim constituye una estrategia voraz eficiente para resolver el problema del árbol de expansión mínima en grafos ponderados.
- Su funcionamiento se basa en la incorporación progresiva de aristas de menor peso garantizando una solución óptima.
- El análisis realizado permitió comprobar que el algoritmo conecta todos los vértices sin generar ciclos y minimizando el costo total de conexión.
- La complejidad computacional puede optimizarse significativamente mediante el uso de estructuras de datos adecuadas.
- Las aplicaciones identificadas demuestran la utilidad del algoritmo en telecomunicaciones, transporte, distribución eléctrica y redes informáticas.
- El algoritmo continúa siendo una de las herramientas más importantes dentro del área de diseño y análisis de algoritmos.

VI. RECONOCIMIENTOS

Se agradece a la Universidad Nacional de Trujillo y al curso de Estrategias Algorítmicas por proporcionar los conocimientos y recursos académicos que hicieron posible el desarrollo de la presente investigación.

VI. REFERENCIAS BIBLIOGRÁFICAS

[1]. *T. H. Cormen, C. E. Leiserson, R. L. Rivest y C. Stein, Introduction to Algorithms, 3rd ed. Cambridge, MA: MIT Press, 2009.*

[2]. *R. C. T. Lee, S. S. Tseng, R. C. Chang y Y. T. Tsai, Introducción al Diseño y Análisis de Algoritmos. México: McGraw-Hill, 2007.*

[3]. *R. Guerrero y A. Vallecillo, Técnicas de Diseño de Algoritmos. Málaga: Universidad de Málaga, 1999.*

[4]. *J. Kleinberg y É. Tardos, Algorithm Design. Boston: Pearson, 2006.*

[5]. *R. Sedgewick y K. Wayne, Algorithms, 4th ed. Boston: Addison-Wesley, 2011.*

[6]. *D. B. West, Introduction to Graph Theory, 2nd ed. New Jersey: Prentice Hall, 2001.*

[7]. *J. Gross y J. Yellen, Graph Theory and Its Applications. Boca Raton: CRC Press, 2006.*

[8]. *R. Diestel, Graph Theory, 5th ed. Berlin: Springer, 2017.*

[9]. *S. Dasgupta, C. Papadimitriou y U. Vazirani, Algorithms. New York: McGraw-Hill, 2008.*

[10]. *N. Deo, Graph Theory with Applications to Engineering and Computer Science. New Jersey: Prentice Hall, 1974.*

[11]. *F. Harary, Graph Theory. Reading: Addison-Wesley, 1969.*

[12]. *M. Goodrich y R. Tamassia, Data Structures and Algorithms in Java. Hoboken: Wiley, 2014.*

[13]. *A. Aho, J. Hopcroft y J. Ullman, Data Structures and Algorithms. Reading: Addison-Wesley, 1983.*

[14]. *S. Skiena, The Algorithm Design Manual. London: Springer, 2008.*

[15]. *J. Bondy y U. Murty, Graph Theory. New York: Springer, 2008.*

[16]. *M. Sipser, Introduction to the Theory of Computation. Boston: Cengage Learning, 2012.*

[17]. *D. Knuth, The Art of Computer Programming. Boston: Addison-Wesley, 1997.*

[18]. *E. W. Dijkstra, A Discipline of Programming. New Jersey: Prentice Hall, 1976.*

[19]. *G. Chartrand y P. Zhang, Introduction to Graph Theory. New York: McGraw-Hill, 2012.*

[20]. *R. C. Prim, "Shortest Connection Networks and Some Generalizations", Bell System Technical Journal, vol. 36, pp. 1389–1401, 1957.*

Anexo

Prim (Grafo G)

1. Seleccionar un vértice inicial.
2. Marcarlo como visitado.
3. Mientras existan vértices sin visitar:

   Buscar la arista de menor peso

   que conecte un vértice visitado

   con uno no visitado.

   Agregar la arista al árbol.

   Marcar el nuevo vértice.
4. Fin Mientras
5. Retornar Árbol de Expansión Mínima.

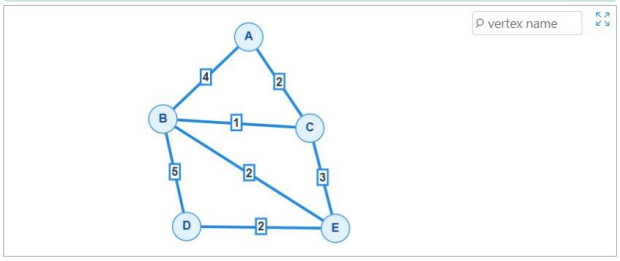

Fig. 1. *Grafo de utilizado*

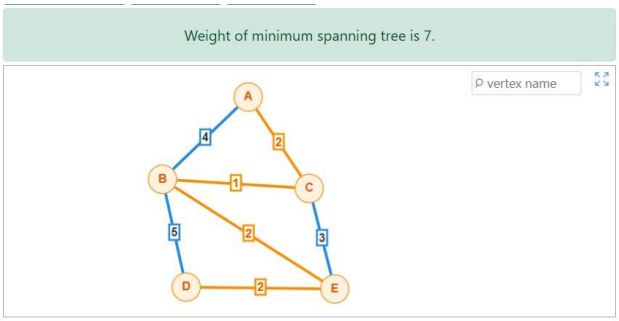

Fig. 2. *Recorrido*

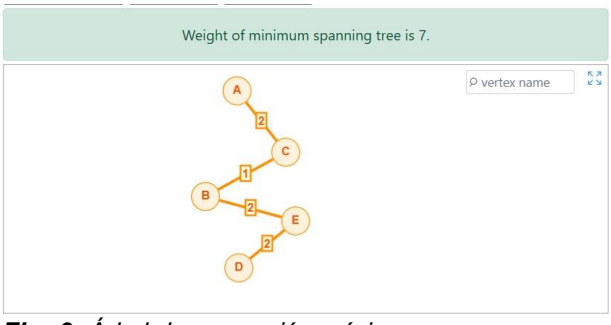

Fig. 3. *Árbol de expansión mínima*

El análisis del grafo ponderado permitió comprobar el funcionamiento del algoritmo de Prim en la construcción de un árbol de expansión mínima. A través de la selección progresiva de las aristas con menor peso, se logró conectar todos los vértices del grafo sin generar ciclos y obteniendo el menor costo total posible. Este resultado evidencia la eficacia de la estrategia voraz empleada por el algoritmo, demostrando su utilidad para resolver problemas de optimización relacionados con redes de comunicación, transporte y distribución de recursos, donde es fundamental minimizar los costos de conexión.