Partition into perfect matchings

GT[16] - partition into perfect matchings

*Luis Daniel Fuentes Licero*

*licerol@uninorte.edu.co*

*Universidad del norte*

*200149098*

*3226018344*

***RESUMEN***— **En este artículo se ve a desarrollar la explicación del problema I+D, asignado, el cual es el GT[16], partition into perfect matchings, se desarrolló dicho problema usando el lenguaje de programación de JAVA, se construyó el tiempo de ejecución del algoritmo y se halló su complejidad y se realizó una comparación con el algoritmo común con un algoritmo mejorado.**

**ABSTRACT- In this article we are going to develop the explanation of the R&D problem, assigned, which is the GT[16], partition into perfect matchings, this problem was developed using the JAVA programming language, the execution time of the algorithm was constructed and its complexity was found and a comparison was made with the common algorithm with an improved algorithm.**

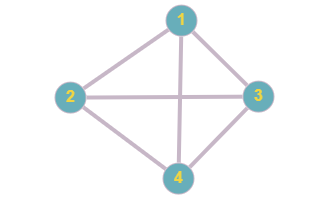
I. INTRODUCCIÓN

El problema de particionar un grafo en unos emparejamientos “perfectos” es un problema bastante recurrente en teoría de grafos este concepto en la disciplina matemática de la teoría de grafos, un emparejamiento o conjunto de aristas independientes en un grafo no dirigido es un conjunto de aristas sin vértices comunes. Un problema que tiene sus aplicaciones en la vida real y que su solución y optimización es de vital importancia.

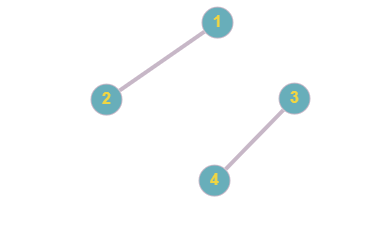
II. PLANTEAMIENTO DEL PROBLEMA

Dado un grafo , el problema es el de particionar el grafo G, en k particiones, donde , estas particiones se deben hacer mediante un emparejamiento perfecto Vi, donde Estos emparejamientos tienen que cumplir la propiedad de que no hay dos aristas en M que compartan el mismo nodo. Y M debe tener la mitad de los vértices del grafo, es decir V/2, esto se puede visualizar mejor de la siguiente manera.

Dado el siguiente grafo:



Al momento de particionarlo en emparejamientos perfectos quedarias asi:



Son 2 parejas que no comparten aristas, y el número de aristas es V/2, es decir 4/2 = 2, por lo que se confirma que efectivamente es una partición en parejas perfectas

III. ANTECEDENTES DEL PROBLEMA

Este problema ha sido estudiado y se dado diversas resoluciones por diversos autores a lo largo de la historia, el primer algoritmo conocido que resolvía este problema fue creado por Jack Edmonds publicado en 1965, y desde entonces muchos investigadores han estudiado su algoritmo con el fin de optimizarlo, en la siguiente tabla se muestra la comparación de los algoritmos hechos por diversos autores y tiempo de ejecución.

TABLA I

|  | **Historia de los algoritmos** | |
| --- | --- | --- |
| **Año** | **Autor/es** | **Complejidad** |
| 1965 | Edmonds |  |
| 1973 | Lawler |  |
| 1974 | Gabow |  |
| 1985 | Gabow |  |
| 1986 | Galil, Micali y Gabow |  |
| 1989 | Gabow, Galil, Spencer |  |
| 1990 | Gabow |  |
| 1991 | Gabow y Tarjan |  |

IV. ALGORITMO

Inicio

F ← lista vacía

Para cada vértice v

añadir v a f

Fin Para

Mientras que hayan vértices o aristas no marcadas del grafo en f

Si w no está F

x ← vértice emparejado de w en m

añadir nodos y aristas a f

Si no

Si distancia es entre el nodo y raiz sea par

//no hacer nada

Si no

Si raiz(v) != raiz(w)

P ← añadir arista entre la raiz y v

retornar P

Si no

B ← subgrafo formada por e y las aristas del camino v → w en T

G’, M’ ← union G y M por B

P ← añadir arista entre P y G’

retornar P

Fin Si

Fin Si

Fin si

añadir arista e

Fin Mientras

añadir vértice v

Fin mientras

Fin

Como el algoritmo es probabilístico, crear una función de tiempo es algo bastante complejo, ya que su resultado puede variar.

La complejidad del algoritmo en el peor de los casos es de donde E es el número de aristas y V el número de nodos, y en el mejor de los casos su complejidad sería de

V. ALGORITMO MEJORADO

Inicio

Si existe un emparejamiento perfecto

M = Emparejamiento perfecto

P =( P(n, M))

Mientras que P !=

P = (P(n’, m’))

Borrar P(n’, m’) de P

M = P(n’, m’)

Si M !=

m’’ = P(n’, m’)

e = m’ - m’’

añadir P(n’/e, M’)

Fin SI

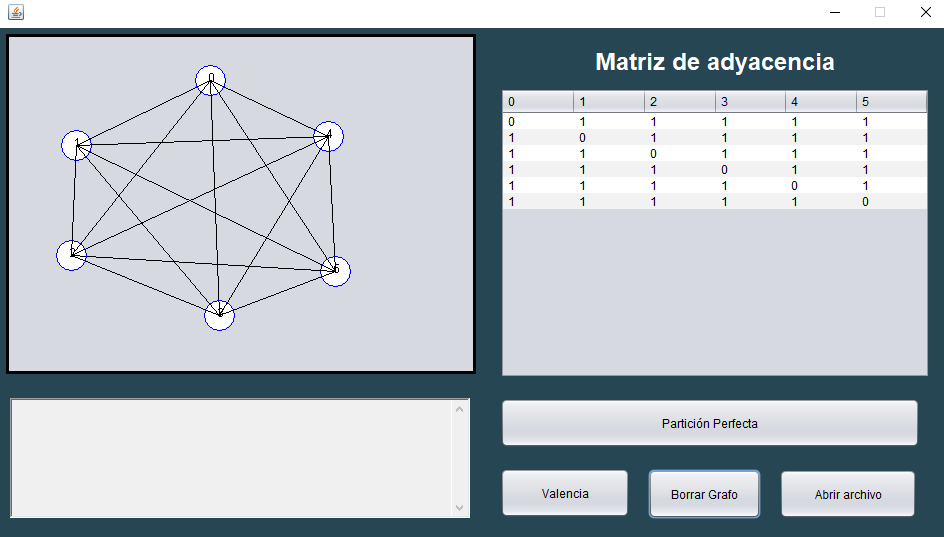
Fin Mientras

Fin Si

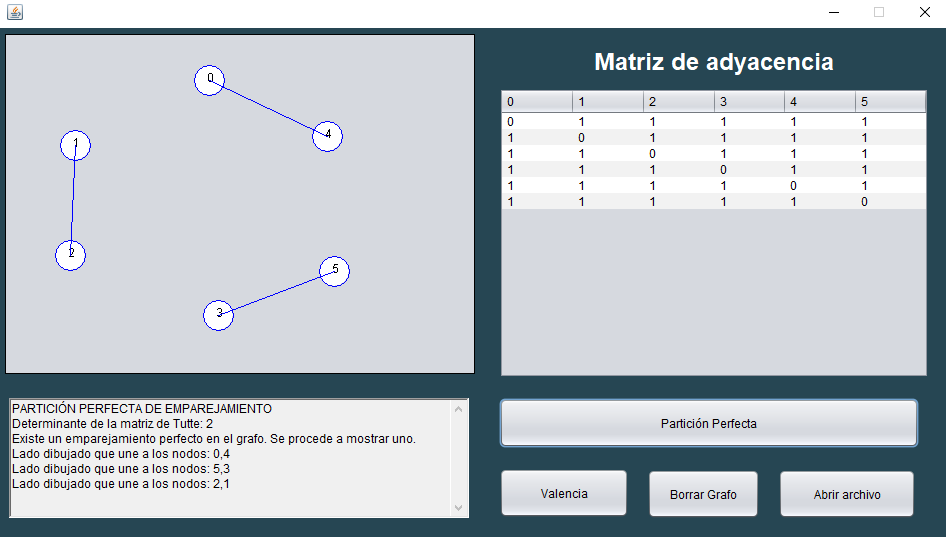
Fin

VI. PROGRAMA EN JAVA

El programa en java se ve de la siguiente manera:



Donde el usuario puede clickear en la pantalla para añadir nodos y unirlos, a la vez que el programa va generando la matriz de adyacencia asociada al grafo, después al aplicar el algoritmo de partición perfecta, queda entonces así:



Esto demuestra inmediatamente que todo grafo k-regular puede ser particionado en emparejamientos perfectos y que todo grafo bipartito k- regular tiene un emparejamiento perfecto.

Otra manera de dibujar el grafo sin tener que hacerlo clickeando la pantalla es teniendo un archivo con la matriz de adyacencia, este archivo debe tener la siguiente forma:

0 1 1 1 1 1

1 0 1 1 1 1

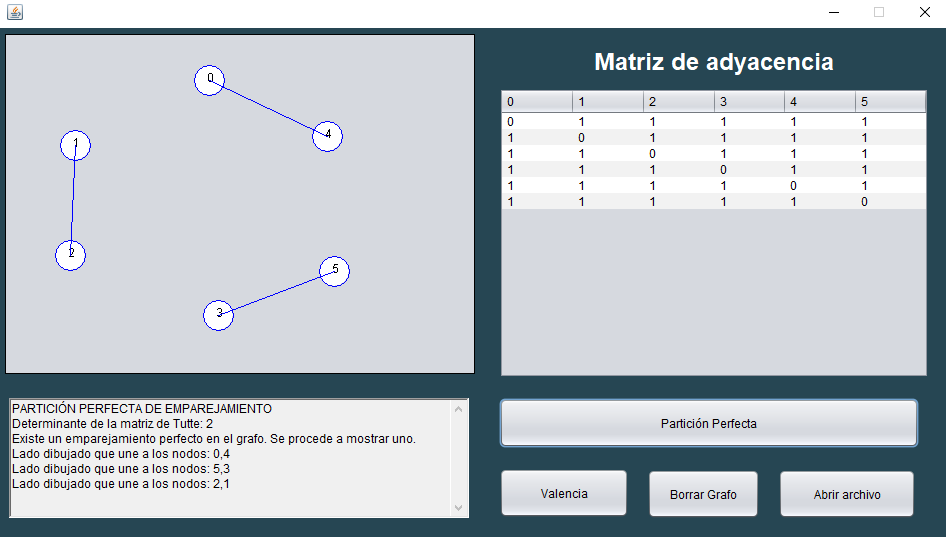
1 1 0 1 1 1

1 1 1 0 1 1

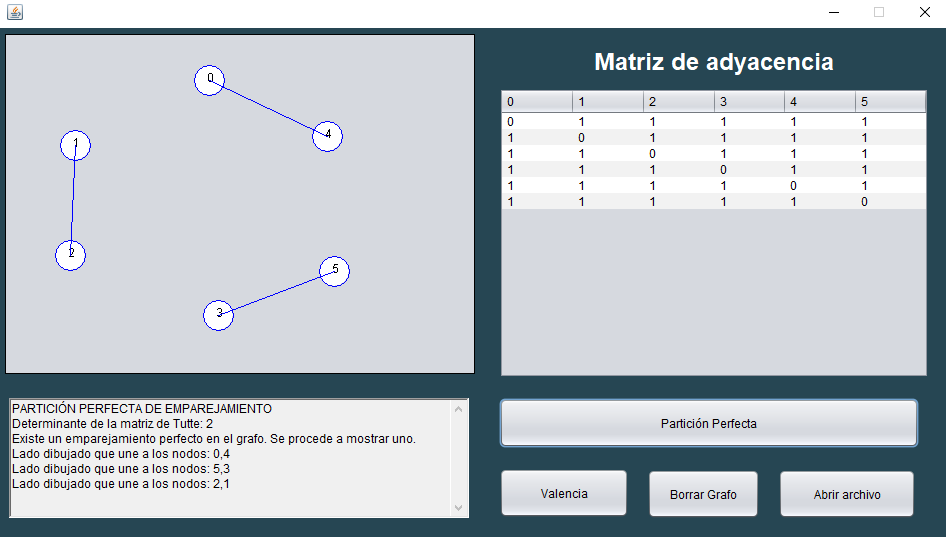
1 1 1 1 0 1

1 1 1 1 1 0

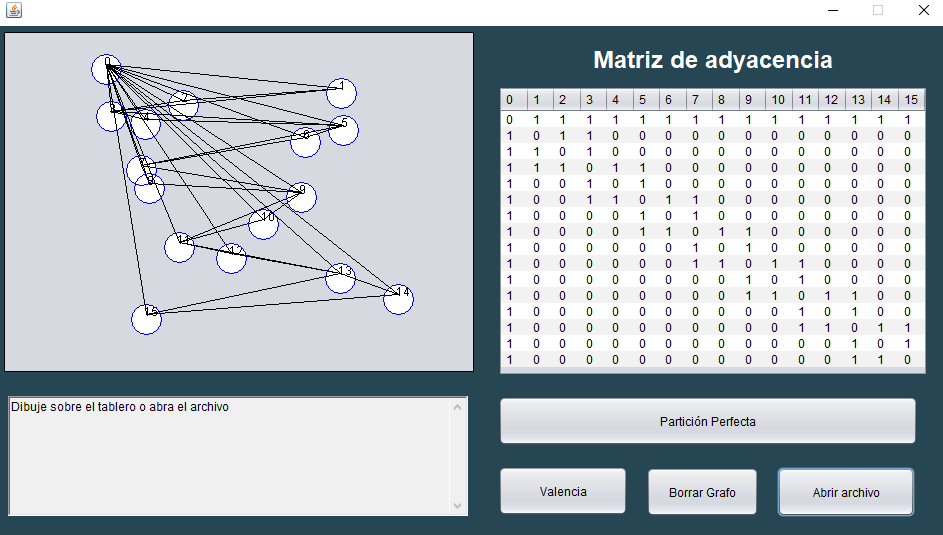
Para añadir dicha matriz de adyacencia del archivo al programa se debe de pulsar el botón de abrir archivo, buscas el archivo en las carpetas del pc.



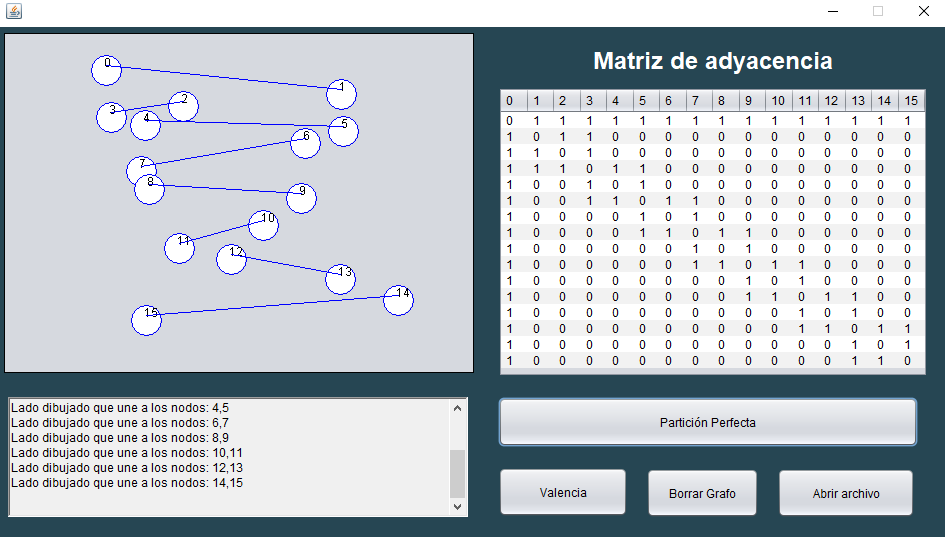
Esto dibujara los nodos de la matriz de adyacencia, y luego ya se pulsa partición perfecta.



El programa para grandes valores de entrada tambien funcionan, por ejemplo para un grafo 15-completo, el programa muestra lo siguiente.

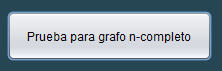


Y al aplicar el algoritmo queda asi:



El algoritmo puede acepatar mucho mas nodos, pero ciertamente el aspecto visual del grafo generado va a empeorar.

Para hacer una prueba para n grafos completos hay que pulsar en el programa.



Después digita n, que es el número de pruebas, el programa hará un grafo completo desde 2 hasta n y para cada grafo aplica el algoritmo de partición perfecta, y luego creará dos archivos en excel con los tiempos de el algoritmo mejorado y el algoritmo no mejorada, dicha tabla es la usada para la prueba de hipótesis.

VII. FUNCIÓN DEL TIEMPO

Igualmente que el algoritmo no mejorado, este es un algoritmo probabilístico, por lo que crear una función de tiempo es algo bastante complejo, ya que su resultado puede variar.

La complejidad del algoritmo es entonces en el peor de los casos es de , y en el mejor de los casos su complejidad sería de

Pero por mejora en la complejidad este funciona para mayores datos de entrada, que algoritmo no mejorado, por ejemplo para el algoritmo no mejorada cuando llegaba a los 12 nodos ya que comenzaba a tardar minutos en realizar las particiones, pero para la version mejorada este seguida siendo funcional y tardaba unos cuantos segundos.

VIII. COMPARACION DE FUNCIONES DE TIEMPO

El algoritmo no mejorado para valores a partir de n=12, el tiempo de ejecución se eleva demasiado y el programa deja de funcionar. por lo que la prueba se realizara para para n=2 hasta 12, debido a que sera un poco sin sentido hacer la prueba para un solo nodo.

Tabla de tiempos de ejecucion en nanosegundos para el algoritmo NO mejorado.

| N0 experimento | n | Particion posible | Tiempos (nanoseg) |
| --- | --- | --- | --- |
| 1 | 2 | Si | 12246000 |
| 2 | 3 | No | 7499400 |
| 3 | 4 | Si | 17465300 |
| 4 | 5 | No | 8474600 |
| 5 | 6 | Si | 26395100 |
| 6 | 7 | No | 13734400 |
| 7 | 8 | Si | 33714500 |
| 8 | 9 | No | 77251700 |
| 9 | 10 | Si | 513885100 |
| 10 | 11 | No | 4885687700 |
| 11 | 12 | Si | 5,8677E+10 |

La media es

Desviación estándar

Varianza

Y para el algoritmo mejorado la tabla de sus tiempos es la siguiente:

| N0 experimento | n | Particion posible | Tiempos (nanoseg) |
| --- | --- | --- | --- |
| 1 | 2 | Si | 12859900 |
| 2 | 3 | No | 7896400 |
| 3 | 4 | Si | 20111200 |
| 4 | 5 | No | 10201400 |
| 5 | 6 | Si | 29968200 |
| 6 | 7 | No | 14086500 |
| 7 | 8 | Si | 34031100 |
| 8 | 9 | No | 12634000 |
| 9 | 10 | Si | 34299600 |
| 10 | 11 | No | 15434600 |
| 11 | 12 | Si | 48595800 |

La media es

Desviación estándar

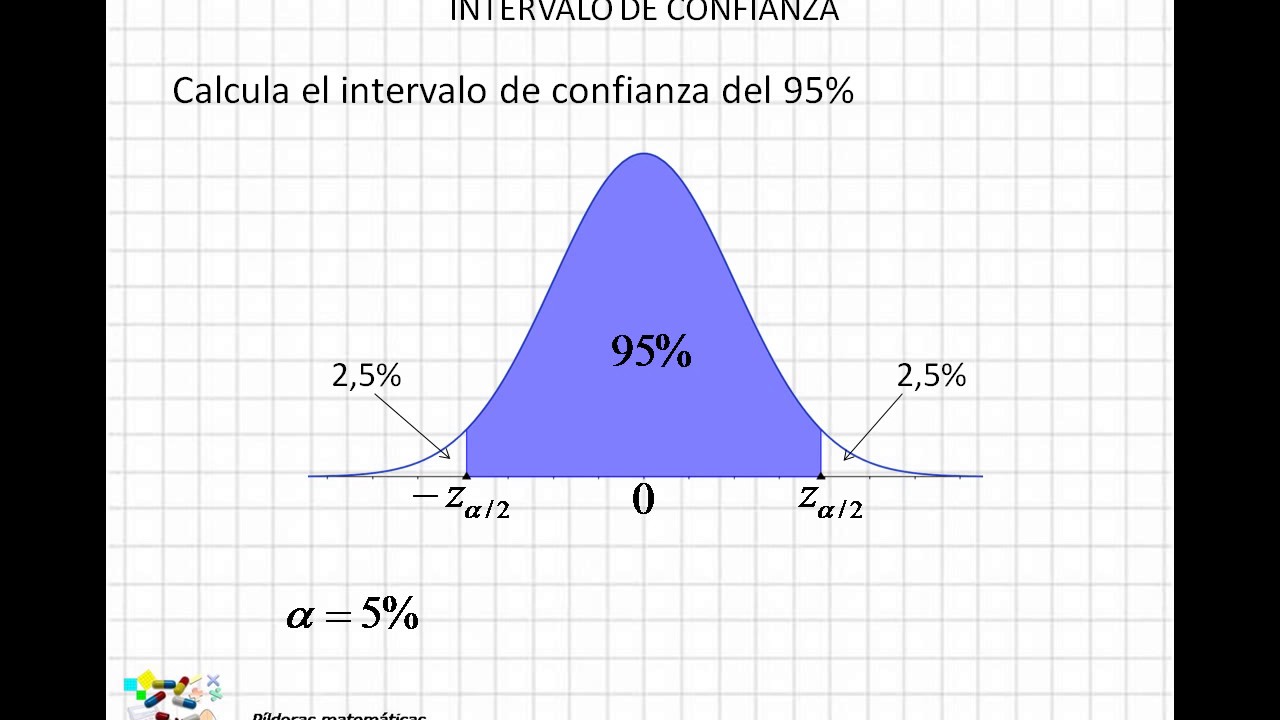
Varianza

Plan experimental:

Hipótesis nula:

Hipótesis alterna:

Nivel de significancia de 95% por lo que la grafica queda así:

****

Cálculo del estadístico de Z

Se rechaza la hipótesis nula, se queda con la hipótesis alterna.

IX. ALGORITMO

X.CONCLUSIONES

Para finalizar este ensayo, podemos concluir que a partir de el problema GT[16] del libro de David S. Johnson y Michael Garey llamado Computers and Intractability, encontramos y diseñamos dos algoritmos que permiten la resolución del problema, particionando un grafo en sus k subgrafos perfectos, del primer algoritmo tiene un tiempo de ejecución un poco mas largo que el algoritmo mejorado, esto lo comprobamos con la prueba estadística para ambas muestras, donde se pudo apreciar mejor, que a parte que la media del tiempo de ejecución del algoritmo mejorado es menor que la media del algoritmo no mejorado. Esta optimización puede tener un impacto en la resolución del problema en los contextos global, ambiental, económico y social, ya que este problema tiene sus aplicaciones en el área de la química ya que este se usa para el área de la química, específicamente en la estructura Kekulé, Kekulé es el término que se utiliza habitualmente para las correspondencias perfectas en las estructuras químicas, como por ejemplo la estructura del Benceno. Por lo que la optimización de los algoritmos usados en la simulación de estos pueden ser bastante útiles y ahorrar a químicos un tiempo bastante considerable para grandes valores de entrada. El impacto ambiental puede ser también bastante grande ya que el poder optimizar y hacer el trabajo de procesamiento y análisis de estructuras químicas como por ejemplo la del benceno, permitirá en menor tiempo analizar algo que normalmente tomaría mucho tiempo, por lo que tendría un efecto positivo para el medio ambiente.

También el algoritmo sirve para casos prácticos, por ejemplo en las aerolíneas donde se empareja automáticamente a los pasajeros con rutas que se solapan, lo que nos permite ofrecer un viaje más barato a todos los pasajeros. Ciertas aerolíneas usan este algoritmo para ofrecer vuelos más baratos ya que cuanto más rutas coincidan, mayor será el descuento que se le ofrece al usuario. En este caso teniendo en cuenta el contexto de un sistema de interconexión usados por aeropuertos, se puede decir que la optimización de este algoritmo permitiendo reducir considerablemente los tiempos de ejecución, es algo muy bueno y mas que todo en sistemas usados en aerolíneas que usualmente tienen un gran tráfico de datos, por lo que cualquier optimización o mejora de rendimiento aunque sea muy pequeña viene de maravilla para estos sistemas, Y esto tendrá claramente un impacto para las economías de estas aerolíneas ya que aunque el sistema funciona para para ofrecer vuelos más baratos cuantas más rutas coincida, el que sea más eficiente significa que más personas podrán acceder a esta oportunidad y en gran escala, en un punto va a ser sostenible o rentable para la aerolínea.

Y por último, aunque estas aplicaciones de el problema de partición de un grafo en sus parejas perfectas son algo rebuscadas, no obstante, son aplicaciones reales, por lo que su solución estaría de verdad ayudando al grupo o nicho que tenga que lidiar con estos problemas dia a dia, por lo que aunque no parezca mucho la optimización o mejora del rendimiento, para gente que trabaja constantemente en esta área, esta pequeña mejora podría ser suficiente como para hacer más eficiente su trabajo.

XI. REFERENCIAS

1. Aho, A. V., et al. «On Finding Lowest Common Ancestors in Trees». SIAM Journal on Computing, vol. 5, n.o 1, marzo de 1976, pp. 115-32. DOI.org (Crossref), https://doi.org/10.1137/0205011.
2. Allen, P. B. «New Method for Solving Boltzmann’s Equation for Electrons in Metals». Physical Review B, vol. 17, n.o 10, mayo de 1978, pp. 3725-34. DOI.org (Crossref), https://doi.org/10.1103/PhysRevB.17.3725.
3. Andrews, George E. «Plane Partitions (III): The Weak Macdonald Conjecture». Inventiones Mathematicae, vol. 53, n.o 3, octubre de 1979, pp. 193-225. DOI.org (Crossref), https://doi.org/10.1007/BF01389763.
4. Avis, David. «A Survey of Heuristics for the Weighted Matching Problem». Networks, vol. 13, n.o 4, 1983, pp. 475-93. DOI.org (Crossref), https://doi.org/10.1002/net.3230130404.
5. Ball, Michael O., y Ulrich Derigs. «An Analysis of Alternative Strategies for Implementing Matching Algorithms». Networks, vol. 13, n.o 4, 1983, pp. 517-49. DOI.org (Crossref), https://doi.org/10.1002/net.3230130406.
6. Bazaraa, M. S., y R. W. Langley. «A Dual Shortest Path Algorithm». SIAM Journal on Applied Mathematics, vol. 26, n.o 3, mayo de 1974, pp. 496-501. DOI.org (Crossref), https://doi.org/10.1137/0126047.
7. ---. «A Dual Shortest Path Algorithm». SIAM Journal on Applied Mathematics, vol. 26, n.o 3, mayo de 1974, pp. 496-501. DOI.org (Crossref), https://doi.org/10.1137/0126047.
8. Bentley, Jon Jouis. «Fast Algorithms for Geometric Traveling Salesman Problems». ORSA Journal on Computing, vol. 4, n.o 4, noviembre de 1992, pp. 387-411. DOI.org (Crossref), https://doi.org/10.1287/ijoc.4.4.387.
9. Brown, J. R. «Shortest Alternating Path Algorithms». Networks, vol. 4, n.o 4, 1974, pp. 311-34. DOI.org (Crossref), https://doi.org/10.1002/net.3230040404.
10. Chang, Alfred E., et al. «Adoptive Immunotherapy of Cancer with Activated Lymph Node Cells Primed In Vivo with Autologous Tumor Cells Transduced with the GM-CSF Gene University of Michigan, Ann Arbor, Michigan». Human Gene Therapy, vol. 7, n.o 6, abril de 1996, pp. 773-92. DOI.org (Crossref), https://doi.org/10.1089/hum.1996.7.6-773.
11. Chegireddy, Chandra R., y Horst W. Hamacher. «Algorithms for Finding K-Best Perfect Matchings». Discrete Applied Mathematics, vol. 18, n.o 2, noviembre de 1987, pp. 155-65. DOI.org (Crossref), https://doi.org/10.1016/0166-218X(87)90017-5.
12. ---. «Algorithms for Finding K-Best Perfect Matchings». Discrete Applied Mathematics, vol. 18, n.o 2, noviembre de 1987, pp. 155-65. DOI.org (Crossref), https://doi.org/10.1016/0166-218X(87)90017-5.
13. Cook, William, y André Rohe. «Computing Minimum-Weight Perfect Matchings». INFORMS Journal on Computing, vol. 11, n.o 2, mayo de 1999, pp. 138-48. DOI.org (Crossref), https://doi.org/10.1287/ijoc.11.2.138.
14. David, Guy, y Carlos Tomei. «The Problem of the Calissons». The American Mathematical Monthly, vol. 96, n.o 5, mayo de 1989, p. 429. DOI.org (Crossref), https://doi.org/10.2307/2325150.
15. Derigs, Ulrich. «A Shortest Augmenting Path Method for Solving Minimal Perfect Matching Problems». Networks, vol. 11, n.o 4, 1981, pp. 379-90. DOI.org (Crossref), https://doi.org/10.1002/net.3230110407.
16. ---. «A Shortest Augmenting Path Method for Solving Minimal Perfect Matching Problems». Networks, vol. 11, n.o 4, 1981, pp. 379-90. DOI.org (Crossref), https://doi.org/10.1002/net.3230110407.
17. Dijkstra, E. W. «A Note on Two Problems in Connexion with Graphs». Numerische Mathematik, vol. 1, n.o 1, diciembre de 1959, pp. 269-71. DOI.org (Crossref), https://doi.org/10.1007/BF01386390.
18. Edmonds, Jack. «Paths, Trees, and Flowers». Canadian Journal of Mathematics, vol. 17, 1965, pp. 449-67. DOI.org (Crossref), https://doi.org/10.4153/CJM-1965-045-4.
19. Ehrhardt, Wolfgang. Das Akademische Kunstmuseum der Universität Bonn unter der Direktion von Friedrich Gottlieb Welcker und Otto Jahn. Westdeutscher Verlag, 1982.
20. Fukuda, K., y T. Matsui. «Finding All the Perfect Matchings in Bipartite Graphs». Applied Mathematics Letters, vol. 7, n.o 1, enero de 1994, pp. 15-18. DOI.org (Crossref), https://doi.org/10.1016/0893-9659(94)90045-0.
21. Hassin, Refael. «The Minimum Cost Flow Problem: A Unifying Approach to Dual Algorithms and a New Tree-Search Algorithm». Mathematical Programming, vol. 25, n.o 2, junio de 1983, pp. 228-39. DOI.org (Crossref), https://doi.org/10.1007/BF02591772.
22. Jockusch, William. «Perfect Matchings and Perfect Squares». Journal of Combinatorial Theory, Series A, vol. 67, n.o 1, julio de 1994, pp. 100-15. DOI.org (Crossref), https://doi.org/10.1016/0097-3165(94)90006-X.
23. Lai, Tri. «Enumeration of Antisymmetric Monotone Triangles and Domino Tilings of Quartered Aztec Rectangles». Discrete Mathematics, vol. 339, n.o 5, mayo de 2016, pp. 1512-18. DOI.org (Crossref), https://doi.org/10.1016/j.disc.2015.12.027.
24. Murty, Katta G. «Letter to the Editor—An Algorithm for Ranking All the Assignments in Order of Increasing Cost». Operations Research, vol. 16, n.o 3, junio de 1968, pp. 682-87. DOI.org (Crossref), https://doi.org/10.1287/opre.16.3.682.
25. «Publications of the National Bureau of Standards. Published by NBS July 1960 through June 1966; Published by Others 1960 through 1965». International Journal of Radiation Biology and Related Studies in Physics, Chemistry and Medicine, vol. 12, n.o 4, enero de 1967, pp. 402-402. DOI.org (Crossref), https://doi.org/10.1080/09553006714550971.
26. Sanders, David E., et al. «SCT89: A Computer Code for Atomic and Molecular Scattering from Clean and Adsorbate Covered Surfaces». Computer Physics Communications, vol. 70, n.o 3, julio de 1992, pp. 579-608. DOI.org (Crossref), https://doi.org/10.1016/0010-4655(92)90118-I.
27. Stanley, Richard P. «Symmetries of Plane Partitions». Journal of Combinatorial Theory, Series A, vol. 43, n.o 1, septiembre de 1986, pp. 103-13. DOI.org (Crossref), https://doi.org/10.1016/0097-3165(86)90028-2.
28. Stembridge, J. R. «The Enumeration of Totally Symmetric Plane Partitions». Advances in Mathematics, vol. 111, n.o 2, marzo de 1995, pp. 227-43. DOI.org (Crossref), https://doi.org/10.1006/aima.1995.1023.
29. Womack, James P., et al. Die zweite Revolution in der Autoindustrie: Konsequenzen aus der weltweiten Studie aus dem Massachusetts Institute of Technology. 6. Aufl, Campus-Verl, 1992.
30. Zhang, Heping. «The Connectivity of Z-Transformation Graphs of Perfect Matchings of Polyominoes». Discrete Mathematics, vol. 158, n.o 1-3, octubre de 1996, pp. 257-72. DOI.org (Crossref), https://doi.org/10.1016/0012-365X(95)00048-2.