Algoritmo ACO aplicado a la obtención aproximada de Triangulaciones de Peso Mínimo*

Maria Gisela Dorzán, Edilma Olinda Gagliardi, Mario Guillermo Leguizamón[1]
Gregorio Hernández Peñalver[2]

[1] Facultad de Ciencias Físico Matemáticas y Naturales, Universidad Nacional de San Luis, Argentina
{mgdorzan,oli,legui}@unsl.edu.ar
[2] Facultad de Informática, Universidad Politécnica de Madrid, España
gregorio@fi.upm.es

Resumen In Computacional Geometry there are many problems that either are NP-hard or no polynomial algorithms are known. Therefore, we can find approximate solutions using metaheuristics. In this work we propose an algorithm based in Ant Colony Optimization (ACO) Metaheuristic to approximate the Minimum Weight Triangulation (MWT) problem. We assess through the experimental evaluation the applicability of the ACO metaheuristic for MWT.

Key words: Metaheurísticas, Geometría Computacional, Triangulaciones, Triangulación de Peso Mínimo, Optimización basada en Colonias de Hormigas.

1. Introducción

En Geometría Computacional hay numerosos problemas que, o bien son de naturaleza NP-dura, o bien son problemas para los cuales no se conocen soluciones eficientes. De todos modos, resulta de interés encontrar soluciones a tales problemas, aunque las mismas sean aproximadas a las óptimas, por medio de métodos de naturaleza heurística. En particular, es interesante el estudio de problemas de optimización geométrica relacionados con ciertas configuraciones geométricas obtenidas a partir de un conjunto de puntos como son las triangulaciones. En este problema se busca optimizar ciertas propiedades que miden la calidad de las configuraciones: peso, número de apuñalamiento, dilación, factor de carga, etc. Dada la dificultad inherente de dichos problemas, los algoritmos aproximados surgen como candidatos alternativos para su aplicación. Estos, pueden dar soluciones cercanas a las óptimas y pueden ser específicos para un problema tratado o formar parte de una estrategia general aplicable en la resolución de distintos problemas, como lo son las técnicas metaheurísticas.

* Parcialmente subvencionado por Proyecto UPM AL09-PAC-12 y por Proyecto Tecnologías Avanzadas de Bases de Datos N 22/F614, UNSL.

Una metaheurística es un proceso de generación iterativo que guía la búsqueda de soluciones combinando inteligentemente diferentes conceptos de diversos campos como: inteligencia artificial [16], evolución biológica [1], inteligencia colectiva [7], entre otros. Una metaheurística da un marco algorítmico general que puede ser aplicado en problemas de optimización con pocas modificaciones que lo adapten a un problema específico. Estos métodos son simples de implementar y han demostrado ser exitosos en encontrar de forma eficiente buenas soluciones para problemas de optimización NP-duros [13].

El algoritmo propuesto en este trabajo deriva de una metaheurística basada en inteligencia colectiva (*Swarm Intelligence*), más específicamente, en el comportamiento de ciertas colonias de hormigas. Este marco presupone un sistema de agentes que obedecen a un conjunto de reglas muy simples, pero que actuando cooperativamente, componen un sistema complejo. En particular, tratamos con la técnica Optimización basada en Colonias de Hormigas (Ant Colony Optimization - ACO), la cual es un proceso distribuido, en el que un conjunto de agentes (reactivos) actúan en forma independiente y cooperan esporádicamente en forma indirecta, para llevar a cabo un objetivo común.

En este artículo presentamos el diseño del algoritmo ACO para la obtención de soluciones aproximadas al problema: Triangulación de Peso Mínimo (Minimum Weight Triangulation - MWT). Presentamos los resultados obtenidos de la evaluación experimental realizada utilizando combinaciones de parámetros adecuadas. El trabajo se ha organizado de la siguiente forma. En la Sección 2 presentamos definiciones y antecedentes del problema MWT. En la Sección 3 describimos los aspectos teóricos y generales de ACO y presentamos el algoritmo ACO propuesto para la resolución del problema estudiado. En la Sección 4 presentamos aspectos de la experimentación y mostramos los resultados obtenidos para conjuntos de cuarenta y ochenta puntos, con diversas configuraciones experimentales para el algoritmo ACO-MWT propuesto. Terminamos en la Sección 5 con las conclusiones y trabajos futuros.

2. Triangulación de Peso Mínimo

Sea S un conjunto de puntos en el plano. Una triangulación T de S es un conjunto maximal de aristas cuyos extremos son puntos de S, y para cualquier par de aristas, ellas no se cortan entre sí. El peso de una triangulación T es la suma de las longitudes euclídeas de todas las aristas de T. La triangulación que minimiza esta suma se denomina triangulación de peso mínimo de S. Esta triangulación, que optimiza un criterio bastante simple y natural, ha motivado en los últimos cuarenta años gran número de trabajos, primero en el intento de su obtención de forma exacta y después en la búsqueda de triangulaciones que se aproximaran a ella. Uno de los hitos fundamentales de estos trabajos fue la demostración en 2006 por Mulzer y Rote [14] de que el problema de obtener una triangulación con peso inferior a un valor k es un problema NP-duro. Con esto concluía la búsqueda de un algoritmo polinómico para obtener

la triangulación de peso mínimo, pero impulsaba la de algoritmos que obtengan buenas aproximaciones con peso cercano al mínimo.

2.1. Antecedentes

El estudio de la triangulación de peso mínimo comenzó antes de 1970. Lo iniciaron Düppe y Gottschalk [5] quienes propusieron que la estrategia *ávida* (*greedy*), en la que en cada paso se añade la arista más corta posible a la triangulación, llevaría al peso mínimo en la triangulación final obtenida. En 1975, Shamos y Hoey [20] indicaron que la triangulación de Delaunay sería la triangulación de peso mínimo. Sin embargo, en 1977 Lloyd [9] presentó ejemplos en donde ni la triangulación obtenida con la estrategia ávida ni la triangulación de Delaunay son la triangulación de peso mínimo. Además ninguna de estas triangulaciones constituye una buena aproximación de la triangulación de peso mínimo. La triangulación de Delaunay puede diferir en un factor $\Omega(n)$ de la de peso mínimo y la triangulación ávida sólo aproxima en un factor $\Theta(\sqrt{n})$ [15]. Describimos a continuación algunos de los resultados obtenidos en cuanto a la búsqueda de buenas aproximaciones.

En cuanto a la búsqueda de buenas aproximaciones, Plaisted y Hong [17] propusieron una heurística que aproxima la triangulación de peso mínimo dentro de un factor de $O(\log n)$. Su algoritmo requiere un tiempo de $O(n^2 \log n)$ para el peor caso. Lingas [10], Levcopoulos et al. [12] y Levcopoulos y Krznaric [11] realizaron un profundo estudio de la relación entre la triangulación de peso mínimo y GT (triangulación ávida) incluyendo una modificación de GT, la denominada triangulación "quasi-greedy", QGT, que aproxima la triangulación de peso mínimo con un factor constante. Para polígonos convexos se obtiene en tiempo lineal un factor de aproximación $(1 + \varepsilon)$, para un ε fijo. Finalmente, Remy y Steger [19] presentaron en 2006 un QPTAS, esquema de aproximación casi-polinómico para obtener la triangulación de peso mínimo. Desde el punto de vista de metaheurísticas, la mayoría de los trabajos encontrados presentan soluciones a problemas dentro de la Computación Gráfica. En 1992, Sen y Zheng [21] proponen un algoritmo para aproximar la triangulación de peso mínimo utilizando *Recocido Simulado* (*Simulated Annealing*) pero en la mayoría de las pruebas realizadas obtienen soluciones "cercanas" a las óptimas. Cada configuración se representa utilizando una estructura de datos y el operador de vecindad simplemente realiza un *flip* (intercambio) en una arista aleatoria elegida aleatoriamente de la triangulación actual. En 1993, Wu y Wainwright [22] aproximan la triangulación de peso mínimo utilizando un algoritmo genético donde los operadores de recombinación y de mutación son iguales, es decir, ambos realizan un flip para obtener los individuos de la próxima población. Qin, Wang y Gong [18] también utilizan un algoritmo genético para lo que proponen nuevos operadores de recombinación y mutación adaptados al problema. Capp y Julstrom [2] presentan una nueva codificación ponderada de las triangulaciones para aplicar en un algoritmo genético. En los trabajos anteriores la evaluación experimental es muy limitada y no se extraen conclusiones acerca de la calidad de las soluciones encontradas. En 2001, Kolingerova y Ferko [8] presentan una optimización genética

cuyo operador de recombinación, denominado DeWall, utiliza las ideas de [4] y el operador de mutación realiza un flip en el individuo seleccionado. Realizan una extensa evaluación experimental variando los parámetros del algoritmo genético utilizando conjuntos de hasta 500 puntos. La debilidad principal del método es la demanda de tiempo.

En 2006 Mulzer y Rote [14] demostraron que la triangulación de peso mínimo es un problema NP-duro, cerrando así la búsqueda de un algoritmo polinómico para la construcción exacta de la triangulación de peso mínimo.

3. Optimización basada en Colonias de Hormigas

La Optimización basada en Colonias de Hormigas es una metaheurística inspirada en el comportamiento que siguen las hormigas para encontrar los caminos más cortos entre las fuentes de comida y el hormiguero, surgida a partir del trabajo inicial de Dorigo, Maniezzo y Colorni sobre Sistema de Hormigas (*Ant System*). Los algoritmos ACO son aptos para resolver problemas de optimización discretos y también continuos. Se basan en una colonia de hormigas artificiales, representadas por agentes computacionales simples, que trabajan de manera cooperativa y se comunican entre sí mediante rastros de feromona artificiales [6]. Son esencialmente algoritmos constructivos: en cada iteración del algoritmo, cada hormiga construye una solución al problema recorriendo un grafo de construcción. Cada arista (i, j) del grafo representa los posibles pasos que la hormiga puede dar y tiene asociadas dos fuentes de información que guían el movimiento de la hormiga: i) Información de los rastros de feromona artificiales, que miden la "deseabilidad aprendida" del movimiento, τ_{ij}, de i a j, imitando a la feromona real depositada por las hormigas reales. Esta información se modifica durante la ejecución del algoritmo dependiendo de las soluciones encontradas por las hormigas (retroalimentación); ii) Información heurística, que mide la preferencia heurística, η_{ij}, de moverse desde el nodo i hasta el nodo j, o sea, de recorrer la arista. Las hormigas no modifican esta información durante la ejecución del algoritmo. Una vez que cada hormiga ha generado una solución se evalúa la misma y se puede depositar una cantidad de feromona en función de la calidad de su solución. Esta información representa el sesgo de búsqueda que influye al resto de las hormigas de la colonia en el futuro. Además, el modo de operación genérico de un algoritmo ACO incluye dos procedimientos adicionales, la evaporación de los rastros de feromona y las acciones del demonio. La evaporación de feromona la lleva a cabo el entorno y se usa como un mecanismo que evita el estancamiento en la búsqueda y permite que las hormigas busquen y exploren nuevas regiones del espacio. Las acciones del demonio son acciones opcionales, que no tienen una correspondencia con el comportamiento de las hormigas reales, para implementar tareas desde una perspectiva global. Algunos ejemplos de acciones del demonio son: observar la calidad de todas las soluciones generadas y depositar una nueva cantidad de feromona adicional sólo en las aristas asociadas a algunas soluciones, o aplicar un procedimiento de búsqueda local

a las soluciones generadas por las hormigas antes de actualizar los rastros de feromona.

3.1. Algoritmo ACO General

A continuación presentamos los pasos fundamentales de la estrategia *ACO*.

Algoritmo 1 ACO-General

```
Inicializar
for c ∈ {1, ..., C} do
   for k ∈ {1, ..., K} do
      ConstSolucionk
   end for
   EvaluarSolucion
   ActualizarRastro
end for
RetornarMejorSolucion
```

Inicializar: Incluye la inicialización de los valores de los parámetros que se consideran en el algoritmo. Entre otros, se deben fijar: el rastro inicial de feromona asociado a cada arista, τ_0, que es un valor positivo pequeño, normalmente el mismo para todas las aristas, el número de hormigas en la colonia, K, y los pesos que definen la proporción en la que afectarán la información heurística y de los rastros de feromonas en la regla de transición probabilística, denominados β y α respectivamente. C es la cantidad de ciclos.

ConstSolucionk: construcción de una solución por la hormiga k. Se inicia con una solución parcial vacía, que se extiende a cada paso añadiéndole un componente de solución factible elegido entre los vecinos de la solución actual. Esto equivale a encontrar una ruta en el grafo de construcción guiada por el mecanismo que define el conjunto de vecinos factibles con respecto a la solución parcial. La elección de un vecino factible se realiza de manera probabilística en cada paso de la construcción, dependiendo de la variante ACO utilizada. En este trabajo, el modelo de probabilidad con retroalimentación utilizado en los algoritmos de construcción es:

$$P_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha}.\eta_{ij}^{\beta}}{\sum_{h \in F(i)} \tau_{ih}^{\alpha}.\eta_{ih}^{\beta}}, j \in F(i); \\ 0, \qquad\qquad \text{en otro caso.} \end{cases} \tag{1}$$

- $F(i)$ es el conjunto de puntos factibles para el punto i.
- τ_{ij} es el valor de feromona asociado a la arista (i, j).
- η_{ij} es el valor heurístico asociado a la arista (i, j).
- α y β son parámetros positivos que determinan la importancia relativa de la feromona con respecto a la información heurística.

ActualizarRastro: actualización del rastro. En este paso se aumenta el nivel de feromona de los caminos prometedores y disminuye el de los caminos no tan buenos. Primero, se reducen todos los valores de feromona por medio del proceso de evaporación. Luego, se incrementa el nivel de feromona al conjunto de soluciones buenas. Se utiliza la siguiente fórmula:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \Delta\tau_{ij} \tag{2}$$

- $\rho \in (0, 1]$ es el factor de persistencia del rastro.
- $\Delta\tau_{ij} = \sum_{k=1}^{K} \Delta^k\tau_{ij}$ es la acumulación de rastro, proporcional a la calidad de las soluciones.
- $\Delta^k\tau_{ij} = \begin{cases} Q/L_k, \text{ si la hormiga } k \text{ utilizó la arista } (i,j); \\ 0, \qquad \text{en otro caso.} \end{cases}$
- Q es una constante que depende del problema; usualmente es igual a 1.
- L_k representa el valor objetivo de la solución k (aparece en el denominador para problemas de minimización).

La evaporación de feromona evita una convergencia demasiado rápida del algoritmo. Además, esta forma de olvidar permite la exploración de nuevas áreas del espacio de búsqueda. La actualización del rastro de feromona se puede realizar al menos de dos formas: *elitista* o no *elitista*. En el caso elitista, se utiliza la mejor solución encontrada para dar un refuerzo adicional a los niveles de feromona. El no elitista utiliza las soluciones encontradas por todas las hormigas para dar un refuerzo adicional a los niveles de feromona. Nuestra propuesta incluye la aplicación de estas dos formas de actualización del rastro de feromona.

3.2. Algoritmo ACO-MWT

A continuación presentamos la adaptación del algoritmo ACO-General al problema MWT. En particular, mostramos el procedimiento *ConstSolucionk* ya que el resto de los procedimientos funcionan de manera similar que en el algoritmo ACO-General. En el algoritmo *ACO*, cada hormiga construye una triangulación, partiendo de un punto inicial cualquiera, agregando aristas a la triangulación mientras no se corten. En caso de no haber aristas factibles desde el punto actual, la elección del próximo punto de referencia se realiza mediante alguno de los siguientes criterios: i) en forma aleatoria; ii) que tenga mayor cantidad de aristas factibles; iii) tenga menor cantidad de aristas factibles. Cabe aclarar que el punto de referencia seleccionado puede pertenecer o no al conjunto de puntos de la solución parcial.

$SeleccionarPuntoInicial(S)$: retorna un punto $p \in S$, elegido aleatoriamente.

$CantFactibles(S_k)$: retorna la cantidad de puntos factibles calculada de acuerdo a las aristas que forman parte de la solución S_k hasta el momento.

$Factibles(i, S_k)$: retorna un conjunto de puntos $p \in S$, tal que la arista (i, p) no corta a ninguna arista perteneciente a la solución S_k.

Algoritmo 2 ConstSolucionk

```
S_k ← ∅                                  /* S_k Solución construida por la hormiga k */
i ← SeleccionarPuntoInicial(S)
while CantFactibles(S_k) > 0 do
   F_i ← Factibles(i, S_k)
   if F_i = ∅ then
      i ← SeleccionarPunto(S, S_k)
      F_i ← Factibles(i, S_k)
   end if
   j ← SeleccionarPuntoProbabilisticamente(F_i)
   S_k ← S_k ∪ {(i, j)}
   ActualizarFactibles(i, j)
   i ← j
end while
```

$SeleccionarPunto(S, S_k)$: retorna un punto $p \in S$, tal que $Factibles(p, S_k)$ debe contener al menos un punto. Esta selección se realiza teniendo en cuenta las aristas que forman parte de la solución S_k y alguno de los siguientes criterios: 1) se elige el punto p en forma aleatoria; 2) se elige el punto p que tenga mayor cantidad de puntos factibles; 3) se elige el punto p que tenga menor cantidad de puntos factibles.

$SeleccionarPuntoProbabilisticamente(F_i)$: retorna un punto $j \in F_i$ elegido de acuerdo al modelo de probabilidad dado en la ecuación 1 donde η_{ij} es igual a $1/d_{ij}$, siendo d_{ij} la distancia euclídea entre los puntos i y j.

$ActualizarFactibles(i, j)$: actualiza los conjuntos de puntos factibles para punto de S considerando la nueva arista (i, j) agregada a la solución.

Para el problema MWT, el procedimiento *EvaluarSolución*, evalúa si la solución S_k obtenida por la hormiga k es mejor que la mejor solución S_{Mejor} obtenida hasta el momento. Es decir, si el peso de la solución S_k es menor que el de la solución S_{Mejor}, S_k pasa a ser la mejor solución.

4. Experimentación

La evaluación de heurísticas aplicadas en la resolución de problemas geométricos necesita generar aleatoriamente instancias de prueba, para contar con conjuntos de tamaño y complejidad variados. Se implementó un generador de conjuntos de puntos, utilizando diferentes funciones de generación aleatoria de la librería CGAL [3]. Dichos conjuntos varían en tamaño y distribución en el plano, como por ejemplo puntos generados en un disco, en un cuadrado, en un círculo, entre otros. Se implementó el algoritmo ACO para MWT en lenguaje C, encontrándose actualmente en etapa de evaluación experimental. Se están realizando pruebas con conjuntos de 40, 80, 120, 160 y 200 puntos, con 50 hormigas y 1000 iteraciones por ejecución. Los valores utilizados en los parámetros del algoritmo ACO son ρ: 0.1, 0.25 y 0.5; α: 1; y β: 1 y 5. Se realizan 30 ejecuciones para cada combinación de valores de parámetros, dado que se consideran 30 semillas

para la funcin aleatoria, y se obtiene la mediana, varianza y promedio de los resultados obtenidos. Este artículo muestra la primera etapa experimental, en donde se han utilizado combinaciones estándares para los parámetros de ejecución, a modo de tener resultados preliminares que guíen la experimentación a futuro, y permitan tomar decisiones fundamentadas acerca de cuáles son los valores realmente adecuados.

	Combinación	Mejor	Media	Varianza	Mediana
1	1-5-5-1-3	4755,22998	10174,34766	6992,94873	9070,584961
2	1-1-5-1-3	4768,859863	7326,850586	1248,872925	7437,22998
3	1-1-5-0-3	4897,290039	7528,376953	1102,507202	7679,165039
4	1-5-1-1-1	5175,560059	6796,282715	710,010559	6749,379883

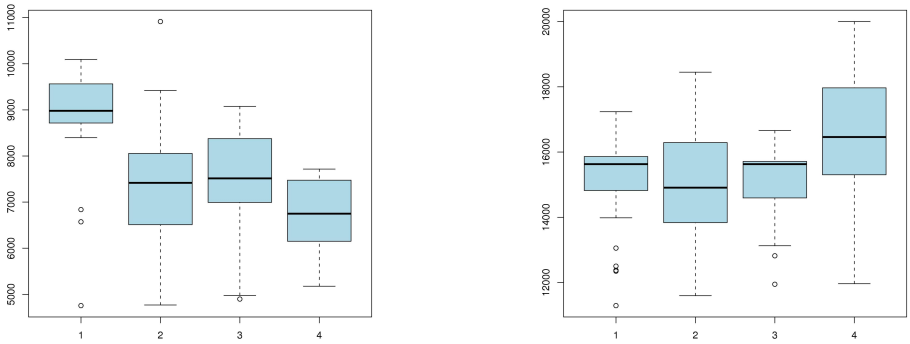
Figura 1. Resultados obtenidos considerando los cuatro mejores valores, para las otras medidas en las respectivas configuraciones para conjunto de 40 puntos.

	Combinación	Mejor	Media	Varianza	Mediana
1	1-5-5-0-2	11291,88965	15156,28418	1493,786621	15629,75
2	1-5-1-1-2	11595,00977	14907,50293	1868,981323	15191,66504
3	1-5-1-0-2	11946,41992	15022,59473	1178,1698	15629,75
4	1-5-1-1-1	11963,69043	16480,18164	1733,206421	16875,96094

Figura 2. Resultados obtenidos considerando los cuatro mejores valores, para las otras medidas en las respectivas configuraciones para conjunto de 80 puntos.

Presentamos los resultados experimentales, para lotes de 40 y 80 puntos, mostrando los valores de las medidas estadísticas habituales (media, varianza, mediana), considerando los valores obtenidos para la función objetivo (peso). En las figuras 1 y 2, considerando los cuatro mejores valores, mostramos las tablas con los resultados obtenidos para las otras medidas en las respectivas configuraciones. Los boxplots corresponden a los 30 valores obtenidos en las cuatro combinaciones correspondientes a las tablas mostradas. Para cada medida se muestra la combinación de parámetros y el valor de la función objetivo obtenida. La combinación de parámetros se debe interpretar de la siguiente manera: α-β-ρ-*elit*-*criterio*, donde α y β corresponden a los valores utilizados en la ecuación 1 y ρ es el valor utilizado en la ecuación 2. Si *elit* es igual a 1 significa que el rastro se actualiza de manera elitista y si es igual a 0 se actualiza de manera no elitista. *criterio* se refiere a la manera de seleccionar un punto en el procedimiento $SeleccionarPunto(S, S_k)$. Si es igual a 1 se elige el punto en forma aleatoria; si es igual a 2 se elige el punto que tenga mayor cantidad de puntos factibles; y si es igual a 3 se elige el punto que tenga menor cantidad de puntos factibles.

Observando los boxplots, en el caso a), la combinación número cuatro es la que muestra mejor comportamiento, dado que su mediana está mejor distribuida respecto de sus valores máximo y mínimo, no hay valores atípicos (outliers), y si bien, entre los bigotes no hay similitud, son bastantes cercanos a la mediana; además, mantiene el mejor valor medio, pero no es la mejor combinación para el mejor valor. La combinación número dos, se ve más uniforme, pero su mediana no se aproxima a tanto al peso mínimo como en el caso anterior, y sus bigotes



(a) (b)

Figura 3. Boxplots para a) 40 y b) 80 puntos de las cuatro mejores combinaciones.

están más alejados. Respecto de las demás combinaciones, observamos que la combinación número uno es la que obtiene el mejor valor, más aproximado al peso mínimo, pero como un caso atípico. Para el caso b), las combinaciones dos y cuatro son las más regulares, pero quien más se aproxima al peso mínimo es la segunda, e incluso los bigotes son más regulares. Pero, la combinación uno es la que obtiene como casos atípicos, los mejores valores aproximados al peso mínimo.

5. Conclusiones y trabajos futuros

En este artículo hemos presentado el diseño de un algoritmo *ACO* para la obtención de soluciones aproximadas al problema de triangulación de peso mínimo. Además mostramos los resultados obtenidos en la primera etapa experimental que nos permiten mostrar cuáles son las combinaciones de parámetros del algoritmo $ACO-MWT$ más propicias para el problema de optimización presentado. En función de los resultados obtenidos en esta primera etapa, actualmente la experimentación se realiza sobre diversos lotes de pruebas, los cuales son obtenidos por diversos generadores de puntos, manteniéndose lotes de 40, 80, 120, 160 y 200 puntos. De esta manera, pretendemos realizar un análisis estadístico en profundidad, a fin de fundamentar las configuraciones que conjeturan ser las más adecuadas para el problema en cuestión. También, a futuro, pretendemos constatar el rendimiento (*performance*) del algoritmo *ACO-MWT* comparándolo con otras técnicas metaheurísticas, a fin de poder realizar un análisis elaborado acerca del comportamiento de estas técnicas en la obtención de triangulaciones de mínimo peso.

Referencias

1. Bäck T., Fogel D., Michalewicz Z.: Handbook of Evolutionary Computation. IOP Publishing Ltd and Oxford University Press (1997)
2. Capp K., Julstrom B.: A weight-coded genetic algorithm for the minimum weight triangulation problem. Proc.of ACM symposium on Applied Computing (1998)
3. Computational Geometry Algorithms Library (CGAL). http://www.cgal.org/
4. Cignoni P., Montani C., Scopigno R.: A Merge-first divide & conquer algorithm for Ed Delaunay triangulations. Intern Rep C92/16, CNUCE/CNR, Pisa (1992)
5. Düppe R., Gottschalk H.: Automatische Interpolation von Isolinien beiwillkürlichen Stützpunkten. Allgemeine Vermessungsnachrichten 77, 423-426 (1970)
6. Dorigo M., Stützle T.: Ant Colony Optimization. Massachusetts Institute of Technology (2004)
7. Kennedy J., Eberhart R.: Swarm Intelligence. Morgan Kaufmann Publishers (2001)
8. Kolingerova I., Ferko A.: Multicriteria-optimized Triangulations. The Visual Computer, Springer Verlag, Vol. 17, No. 6, 2001, s.380-395 (2001)
9. Lloyd E.: On triangulations of a set of points in the plane. In 18th Annual Symposium on Foundations of Computer Science (Providence, R.I., 1977). IEEE Computer Society, Long Beach, Calif., 228-240 (1977)
10. Lingas A.: A linear-time heuristic for minimum weight triangulation of convex polygons. In Proc. 23rd Allerton Conf. Commun. Control Comput. (1985)
11. Levcopoulos C., Krznaric D.: A nearoptimal heuristic for the minimum weight triangulation of convex polygons. Unpublished (1997)
12. Levcopoulos C., Lingas A., Sack J.: Heuristics for optimum binary search trees and minimum weight triangulation problems. Theoret. Comput. Sci, 66(2):181-203 (1989)
13. Michalewicz Z., Fogel D.: How to Solve It: Modern Heuristics. 2nd Edition, Springer (2004)
14. Mulzer W. y Rote G., Minimum weight triangulation is NP-hard. Proceedings of the 22nd Annual ACM Symp. on Computational Geometry. pp. 1-10 (2006)
15. Manacher G., Zobrist, A.: Neither the greedy nor the Delaunay triangulation of a planar point set approximates the optimal triangulation. Inf. Process. Lett. 9, 1, 31-34 (1979)
16. I. Osman y J. Kelly, MetaHeuristics: Theory & Application. Kluwer Academic Publishers (1996)
17. Plaisted D., Hong J.: A heuristic triangulation algorithm. J. Algorithms, 8:405-437 (1987)
18. Qin K., Wang W., Gong M.: A genetic algorithm for the minimum weight triangulation. In Proceedings of 1997 IEEE International Conference on Evolutionary Computation, pages 541-546 (1997)
19. Remy J., Steger A.: A quasi-polynomial time approximation scheme for minimum weight triangulation. Proc. of the 38th Annual ACM. ACM Press, New York, 316-325 (2006)
20. Shamos M., Hoey D.: Closest-point problems. In 16th Annual Symp. on Found.of Computer Science. IEEE Computer Society, Long Beach, Calif., 151-162 (1975)
21. Sen S., Zheng S.: Near-optimal triangulation of a point set by Simulated Annealing. Proceedings of the 1992 CM/SIGAPP Symposium on Applied Computing, pp. 1000-1008 (1992)
22. Wu Y., Wainwright R.: Near-optimal triangulation of a point set using Genetic Algorithms. Proceedings of the Seventh Oklahoma Conference on Artificial Intelligence, pp. 122-131, November (1993)