Análisis del algoritmo de optimización por enjambre de partículas por medio de una aplicación gráfica 3D

Charles F. Velázquez Dodge, M. Mejía Lavalle

Centro Nacional de Investigación y Desarrollo Tecnológico,
Cuernavaca, Morelos,
México

{charlesdodge, mlavalle}@cenidet.edu.mx

Resumen. El método de optimización por enjambre de partículas, se ha vuelto popular en los últimos años, dado a su eficiencia y bajo costo computacional. En este documento, se realiza un breve análisis del algoritmo estándar por medio de una aplicación gráfica 3D que se desarrolló. El código fuente de la aplicación es libre y es posible descargarla en GitHub.

Palabras clave: Métodos de optimización, optimización de enjambre de partículas, algoritmos.

Analysis of Particle Swarm Optimization Algorithm Using a 3D Application

Abstract. Particle swarm optimization method has become popular in recent years, because of its efficiency and low computational cost. In this document, a brief analysis of the standard algorithm is performed by a 3D application. The source code of the application is free software and you can download at GitHub.

Keywords: Optimization methods, particle swarm optimization, algorithms.

1. Introducción

El algoritmo de optimización por enjambre de partículas (PSO), fue originalmente desarrollado por James Kennedy y Russell Eberhart en 1995, es un algoritmo del área de la inteligencia artificial de la rama de inteligencia de enjambres. Está inspirada en el comportamiento social de los seres vivos [1]. Comparado con los algoritmos genéticos (GA) que se basa en el mecanismo de evolución biológica, el algoritmo de optimización por enjambre de partículas se inspira en la evolución en el comportamiento colectivo, principalmente trata de imitar el comportamiento social de varios grupos de animales como lo son los cardúmenes, parvadas, manadas, etc. [2]. Los dos algoritmos se basan en población de individuos, solo que con diferentes enfoques y han demostrado ser eficientes para la solución de problemas complejos.

Ambos son algoritmos de optimización metaheurísticos, ya que utilizan analogías con otros procesos para resolver el problema, los métodos metaheurísticos no se especializan a resolver un problema en particular, por lo que puede usarse para cualquier problema. Estos métodos no garantizan dar el mejor resultado, pero sí un resultado aceptable [3].

Otra característica de estos algoritmos, es que son no deterministas (estocásticos), esto quiere decir que los resultados obtenidos no siempre serán los mismos aunque se trate de una misma función.

En muchas de sus aplicaciones, tanto los algoritmos genéticos, como los algoritmos de enjambre de partículas, ofrecen resultados de calidad del 99%, sin embargo, se ha demostrado que los algoritmos de enjambre de partículas son superiores en cuanto a eficiencia, debido a su bajo costo computacional[4].

El algoritmo de optimización de enjambre de partículas original ha tenido varios cambios y han surgido variaciones del mismo según el problema que se quiera resolver. Sin embargo, en 2006, se trató de establecer un estándar (SPSO), y que posteriormente se le ha contribuido con algunas sugerencias y cambio en el 2007 y 2011 [5-6]. Comparado con el algoritmo clásico, el estándar, agregó el factor social, cognitivo, de inercia y constricción. También se cambió el modo en que las partículas se comunicaban por medio de topologías definidas.

2. Descripción del estándar

En esta parte se describirán los factores utilizados en el estándar, así también como las fórmulas utilizadas según los factores a utilizar.

2.1. Cognitiva y social

La cognitiva contribuye para que la partícula tenga una especie de memoria y pueda saber si anteriormente había adoptado una mejor posición [3].

El factor social, es el responsable de que la partícula sea influenciado por otras partículas en una mejor posición, provocando ser atraída al óptimo encontrado [3].

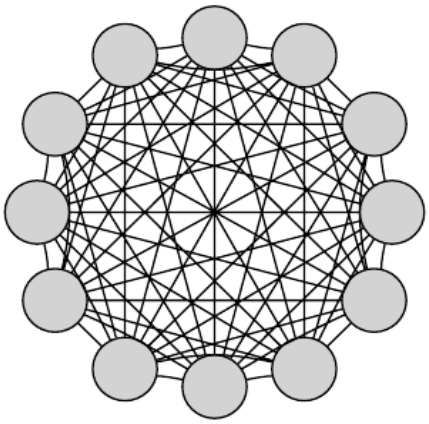

Fig. 1. Modelo gbest [6]

2.2. Topología gbest y lbest

El concepto del modelo gbest es que todas las partículas se comunican entre sí y el mejor punto encontrado, es comunicado a las demás partículas. Tiene la ventaja de ser muy rápido para encontrar un valor óptimo, pero esto genera un problema, ya que puede provocar una convergencia prematura en un óptimo local, que queremos evitar en la mayoría de los casos [7].

En contraste, el modelo lbest es más lento pero hay más posibilidad de evitar una convergencia prematura, en la práctica, el modelo mantiene varios puntos de atracción que hace escapar de óptimos locales [7].

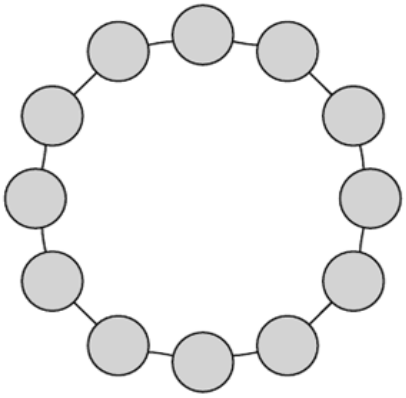

Fig. 2. Modelo lbest [6]

2.3. Inercia y constricción

La inercia es un factor que se añadió para dar mayor estabilidad a la fórmula, permite que la partícula siga una trayectoria constante ignorando la influencia de otros [3].

El factor de constricción, a diferencia de la inercia, es variable, permite un equilibrio entre búsquedas locales y globales. Cuando este factor es menor a 4, el enjambre se mueve lentamente y tiene una convergencia lenta, mientras que si es mayor a 4, logra una convergencia rápida [7].

El factor de inercia modulada, brinda un comportamiento similar al de la constricción. Consiste en comenzar con una inercia alta, e ir disminuyendo con el paso del tiempo [7]. Está comprobado que este factor da mejores resultados que una inercia constante [3].

2.4. Fórmulas

A continuación plantearemos las ecuaciones que conforman el algoritmo. Principalmente existen dos ecuaciones según el factor que se ha elegido, ya sea el factor de inercia o el de constricción. La notación utilizada es la siguiente:

v= velocidad,

i= índice de la partícula,
k= estado en el tiempo,
C_1 = cognitiva,
rand = valor aleatorio,
p^i = mejor posición de la partícula,
C_2 = social,
p^g = mejor posición global,
x = posición de la partícula,
ω= inercia.

La fórmula para el cálculo de velocidad de cada partícula utilizando inercia es:

$$v_{k+1}^i = \omega v_k^i + c_1 rand\left(p_k^i - x_k^i\right) + c_2 rand\left(p_k^g - x_k^i\right). \quad (1)$$

Por otro lado para usar el factor de constricción, la fórmula es:

$$v_{k+1}^i = \chi\left[v_k^i + c_1 rand\left(p_k^i - x_k^i\right) + c_2 rand\left(p_k^g - x_k^i\right)\right], \quad (2)$$

donde χ es el valor de constricción determinado por:

$$\chi = \frac{2}{\left|2 - \zeta - \sqrt{\zeta^2 - 4\zeta}\right|}, \quad (3)$$

$$\zeta = c_1 + c_2. \quad (4)$$

Por último la fórmula para actualizar su posición utilizando inercia es:

$$x_{k+1}^i = x_k^i + v_k^1. \quad (5)$$

Algunos autores proponen otras mejoras como es el de dividir entre un incremento del tiempo, [4] utilizar valores aleatorios basados en alguna distribución [5], sin embargo, en estos momentos nos basaremos en el estándar del 2007.

3. Contribución

Se contribuye con una aplicación que ayudará al entendimiento del algoritmo, fue desarrollada en el lenguaje de programación C++ con librerías multiplataforma Qt y QwtPlot3D. La aplicación se ha probado en varios equipos y funciona perfectamente en un sistema GNU/Linux, sin embargo se han encontrado bugs al momento de ejecutar la aplicación de Windows. Todo el software utilizado es software libre y la aplicación desarrollada se ha subido a GitHub para que sea accesible para todo el público. El código se encuentra en:

https://github.com/CharlesVD/Optimizacion_por_Enjambre_de_Particulas

Las librerías de Qt permiten el desarrollo de aplicaciones gráficas en 3D gracias a que tiene incorporado OpenGL, sin embargo se optó por usar QwtPlot3D para agilizar el desarrollo y no generar primitivas 3D desde cero.

4. Funciones de prueba en la aplicación

Para la aplicación tomamos varias funciones de la tesis doctoral de Helbert Eduardo Espitia Cuchango [7].

Tabla 1. Funciones

Nombre	Función	Mínimo
Parabolic	$f(x,y) = x^2 + y^2$	0
Ackley	$f(x,y) = e + 20 - 20 \exp(-0.2\sqrt{0.5x^2 + y^2}) - \exp(0.5(\cos(2\pi x) + \cos(2\pi y)))$	0
Circles	$f(x,y) = (x^2 + y^2)^{0.25} ((\sin(50(x^2 + y^2)^{0.1}))^2 + 1)$	0
Rastrigin	$f(x,y) = 20 + (x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi y))$	0
Equal Peaks	$f(x,y) = \cos(x)^2 + \sin(y)^2$	0
Himmelblaus	$f(x,y) = -0.01 (200 - (x^2 + y^2 - 11)^2 - (x + y^2 - 7)^2)$	-2
Peaks	$f(x,y) = 3(1-x)^2 e^{-(x^2+(y+1)^2)} - 10(\frac{x}{5} - x^3 - y^5)e^{-(x^2+y^2)} - \frac{1}{3}e^{-((x+q)^2+y^2)}$	-6.4169
Schaffer	$f(x,y) = 0.5 + \frac{\sin(\sqrt{x^2 + y^2})^2 - 0.5}{(1 + 0.1(x^2 + y^2))^2}$	0
Rosenbrock (banana)	$f(x,y) = 100(y - x^2)^2 + (1 - x)^2$	0

5. Conclusión

Gracias a la aplicación se logró entender y analizar el algoritmo, se desarrolló una aplicación que podrá ser útil en generaciones futuras que quieran experimentar o analizar de una manera visual e intuitiva, el comportamiento del algoritmo de optimización por enjambre de partículas. Al ser una aplicación libre, es posible analizarla y modificarla para incorporar otras variaciones del algoritmo, la interfaz gráfica y el algoritmo están separados en clases por lo que no es muy difícil hacer los cambios.

6. Trabajo a futuro

Se pretende incorporar más variaciones del algoritmo a la aplicación, mejorar la compatibilidad con otros sistemas operativos, añadir más parámetros de configuración y durante el proceso, ir encontrado posibles mejoras al algoritmo.

Referencias

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. Purdue School of Engineering and Technology Indianapolis (1995)
2. Kennedy, J., Eberhart, R.: Tutorial on Particle Swarm Optimization. In: IEEE Swarm Intelligence Symposium (2005)
3. Benítez, R., Escudero, G., Kanaan, S.: Inteligencia artificial avanzada. Universitat Oberta de Catalunya, pp. 164–199 (2013)
4. Hassan, R., Cohanim, B., de Weck, O., Venter, G.: A copmarison of Particle Swarm Optimization and The Genetic Algorithm. Massachusetts Institute of Technology, Vanderplaats Research and Development (2004)
5. Clerc, M.: Standard Particle Swarm Optimization (2012)
6. Bratton, D., Kennedy, J.: Defining a Standard for Particle Swarm Optimization. In: IEEE Swarm Intelligence Symposium (2007)
7. Espitia Cuchango, H.E.: Algoritmo de optimización basado en enjambres de partículas con comportamiento de vorticidad. Universidad Nacional de Colombia, Facultad de Ingeniería, pp. 11–22 y pp. 92–96 (2014)