Algoritmos Bioinspirados

Unidad V : Inteligencia de enjambre

Programa: Ingeniería en Inteligencia Artificial

Colonia de abejas artificiales

Colonia de abejas artificiales (ABC de sus siglas en inglés) es un algoritmo basado en enjambres que fue introducido originalmente por Karaboga [1], inspirado en el comportamiento inteligente de las abejas en el forrajeo.

Se basa en 3 componentes principales:

Abejas de forrajeo ocupadas

Abejas de forrajeo libres

Fuentes de comida

1. Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Vol. 200, pp. 1-10). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

Abejas en la naturaleza

Fuentes de comida

- Cercanía al nido.
- Abundancia de alimento.

Abejas de forrajeo ocupadas (abeja obrera)

- Se encuentran asociadas a una fuente de comida.
- Llevan y comparten información sobre la fuente de comida.

Abejas de forrajeo libres (abeja observadora)

- Buscan una fuente de comida que explotar.
- Pueden ser: exploradoras o espectadoras.

Abeja exploradora (scouts)

Buscan comida alrededor del nido

Abeja observadora (onlookers)

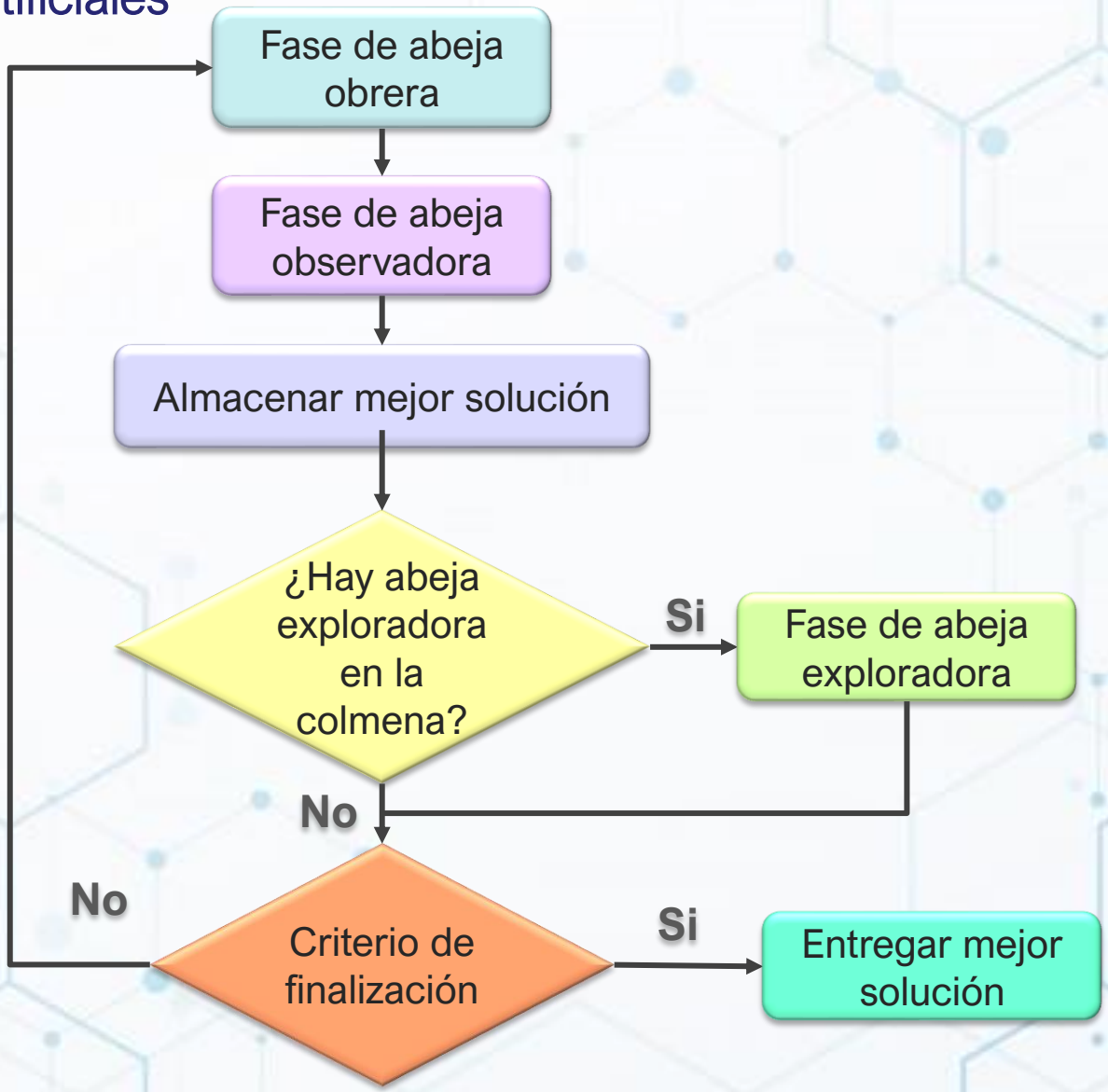
Esperan en el nido por la información por las abejas obreras

- El intercambio de información entre las abejas se realiza a través de una danza (*waggle dance*).Simple en concepto, pero visualmente complejo.
- La información compartida tiene una probabilidad proporcional a la rentabilidad de la fuente de comida (duración del baile)
- Las observadoras ven varias danzas y se deciden por la más rentable.

Colonia de abejas artificiales

- Se basa en el comportamiento de forrajeo de las abejas.
- El enjambre esta conformado por:
  - Abejas obreras
  - Abejas observadoras (*Onlooker*)
  - Abejas exploradoras (*Scout*)
- El rol de cada abeja puede cambiar a lo largo de la ejecución del programa.

Colonia de abejas artificiales



Metodología

1. En la fase inicial solo hay ***exploradoras*** y ***observadoras.*** Las ***exploradoras*** son enviadas a buscar fuentes de comida, mientras que las ***observadoras*** esperan en el nido a ser reclutadas***.*** Si una ***exploradora*** encuentra una fuente de comida se transforma en ***abeja obrera***.
2. La ***abeja obrera*** colecta el néctar y regresa a la colmena. Danza para compartir la información de su fuente de comida.
3. Cada ***observadora*** estima la calidad de las fuentes de comida encontrada por las abejas empleadas y decide seguir a alguna de las ***abejas obreras***. Las mejores fuentes de comida tienen mayor probabilidad de ser seleccionadas.
4. Una vez que una fuente de comida es agotada, la ***abeja obrera*** en esa fuente la abandona y se transforma en ***exploradora***.

Fase de inicialización

- Cada fuente de comida representa una solución al problema a optimizar.
- La cantidad de néctar de una fuente de comida corresponde a la calidad (*fitness*) de la solución asociada a esa fuente.
- El número de abejas obreras es igual al número de soluciones.

Cada solución x_i $(i = 1,2, \dots, \boldsymbol{D})$ es un vector $\boldsymbol{D}$ dimensional donde cada posición es una de las variables/parámetros a optimizar.

Generar una población inicial aleatoria de soluciones de tamaño ***SN*** (fuente de comida). ***SN*** denota el tamaño de la población de soluciones.

$$x_{ij} = l_j + r_1 * (u_j - l_j)$$

$i \rightarrow$ índice del número de soluciones

$j \rightarrow$ índice del número de variables a optimizar

$r_1 \rightarrow$ valor aleatorio [0,1] (distribución uniforme)

$u_j \rightarrow$ límite superior de la j-ésima variable

$l_j \rightarrow$ límite inferior de la j-ésima variable

Fase de inicialización – Parámetros de control [1]

1. *Abejas obreras*: 50% del enjambre. Es asignada a una fuente de comida y provee información del vecindario en la fuente almacenada en su memoria.
2. *Abejas observadoras:* 50% del enjambre. Obtienen información de las abejas empleadas y seleccionan una fuente de comida para recolectar néctar.
3. *Abejas exploradoras:* las abejas cuya fuente de comida se agotó se convierten en exploradoras.
4. Tamaño del enjambre.
5. Límite: indica la cantidad de veces permitida en que no se logra mejorar una solución. Generalmente se define como $\frac{SN}{2} \times D$ donde SN es el tamaño del enjambre.
6. Número de ciclos.

- El número de abejas obreras es igual al número de fuentes de comida.
- Cada fuente de comida es una posible solución al problema
- La cantidad de néctar de una fuente de comida es igual a la calidad de la solución (*fitness*)

Fase de abejas obrera

- Una abeja artificial obrera produce probabilísticamente una modificación en la posición (solución) en su memoria para encontrar una nueva fuente de alimento y prueba la cantidad de néctar (valor de aptitud) de la nueva fuente (nueva solución).

Las abejas obreras seleccionan aleatoriamente una posición de fuente de alimento y producen una modificación en la existente en su memoria como se describe en:

$$v_{ij} = x_{ij} + r_2 * (x_{ij} - x_{kj})$$

Donde $k \in \{1,2,\dots,\boldsymbol{SN}\}$ y $j \in \{1,2,\dots,\boldsymbol{D}\}$ son índices seleccionados aleatoriamente y $k \neq i$. Si la nueva solución excede el valor permitido de alguna variable, se asigna el valor mayor/menor permitido.

v_{ij} → la j-ésima posición de la solución i actualizada

x_{ij} → la j-ésima posición de la solución i

x_{kj} → la j-ésima posición de otra solución k seleccionada al azar

r_2 → número aleatorio entre [-1,1] (distribución uniforme)

Fase de abejas obrera

- Si el parámetro de **límite** se alcanza, la fuente de comida se abandona.
- Si la cantidad de néctar (fitness) de la nueva solución es mayor que la anterior, la abeja memoriza la nueva solución y olvida la anterior. Inicializar el parámetro **limite** a cero.
- Después que todas las abejas obreras completan el proceso de búsqueda, comparten la información de las fuentes de comida con las abejas espectadoras.

Fase de abejas observadora

- Una abeja observadora evalúa la información de néctar tomada de todas las abejas empleadas y elige una fuente de alimento con una probabilidad relacionada con su cantidad de néctar, calculada con la siguiente expresión:

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n}$$

fit_i puede ser el valor de aptitud de la función que se está optimizando.

Para caso de minimización usar:

$$fit_i = \begin{cases} \frac{1}{1+f(i)} & si\ f(i) \geq 0 \\ 1+|f(i)| & si\ f(i) < 0 \end{cases}$$

Una vez que se tiene calculada la probabilidad de selección de cada fuente de comida, se realiza la selección usando cualquier método de selección basado en la aptitud. La forma más sencilla es:

Fase de abejas observadora

- Luego que la abeja observadora elige una fuente de comida, se crea una fuente de comida cercana usando la misma expresión que la abeja obrera. Se evalúa la nueva fuente de comida y la abeja observadora se queda con la mejor fuente de comida.

$$v_{ij} = x_{ij} + r_2 * (x_{ij} - x_{kj})$$

Exploración contra explotación

Las abejas obreras y observadoras realizan el proceso de explotar una fuente de comida.

Las abejas exploradoras llevan el proceso de explorar nuevas fuentes de comida.

Fase de abeja exploradora

- Las abejas libres que eligen sus fuentes de alimento al azar se llaman exploradoras.
- Las abejas obreras cuyas soluciones no mejoran luego de un número predeterminado (parámetro límite), se convierten en exploradoras y sus soluciones son abandonadas.

La nueva fuente de comida (solución) se genera con la misma expresión que se uso durante la fase de inicialización.

$$x_{ij} = l_j + r_1 * (u_j - l_j)$$

Pseudocodigo [2]

1: Inicializar la población de soluciones $x_{ij}, i = 1,2,\dots,SN, j = 1,2,\dots,D$.

2: Evaluar la población (fit_i).

3: ciclo=1

4: Repetir

5: Producir nuevas soluciones v_{ij} para las abejas obreras y evaluarlas.

6: Seleccionar mejores soluciones.

7: Calcular los valores de probabilidad p_i para cada solución x_{ij}

8: Producir nuevas soluciones v_{ij} para las observadoras a partir de las soluciones x_{ij} seleccionadas en función de p_i y evaluarlas.

9: Seleccionar mejores soluciones.

10: Determinar soluciones abandonadas las exploradoras, si existe, y reemplazarla con una nueva solución aleatoria x_{ij}.

11: Memoriza la mejor solución lograda hasta el momento.

12: ciclo=ciclo+1

13: hasta que ciclo = *max_ciclo*

Fase inicial (Exploradora)

Generar SN soluciones usando la fórmula:

$$x_{ij} = l_j + r_1 * (u_j - l_j)$$

Fase de Obrera

- Generar nueva solución $v_{ij} = x_{ij} + r_2 * (x_{ij} - x_{kj})$
- Calcular aptitud de la nueva solución
- Selección de nueva solución

Fase de Exploradora

- Encontrar fuentes de comida abandonadas.
- Generar nuevas fuentes al azar usando

$$x_{ij} = l_j + r_1 * (u_j - l_j)$$

Fase de Observadora

- Calcular probabilidades (selección de obrera a seguir)
- Producir nueva solución
- Calcular aptitud de la nueva solución
- Selección de nueva solución

Calcular probabilidades

Fase de **Obrera**

Por cada fuente de comida (obrera):

- Seleccionar variable a actualizar y socio.
- Actualizar solución usando:

$$v_{ij} = x_{ij} + r_2 * (x_{ij} - x_{kj})$$

- Seleccionar nueva solución

Fase **Observadora**

Usar probabilidad para seleccionar obrera a seguir

Actualiza solución usando

$$v_{ij} = x_{ij} + r_2 * (x_{ij} - x_{kj})$$

Si contador > límite

Aplicar fase de exploradora

(Generar solución al azar)

$$x_{ij} = l_j + r_1 * (u_j - l_j)$$

iter = iter +1

Minimizar $f(x,y) = x^2 + 2xy - y^2$ donde $-5 \leq x, y \leq 5$

Tamaño del enjambre = 10 Iteraciones = 20

Dimensión del problema = 2 Límite = 2

obreras = # observadoras = # soluciones = 5

Primer paso: generar soluciones iniciales al azar usando $x_{ij} = l_j + r_1 * (u_j - l_j)$

$$\begin{bmatrix} 2.5 & 1.5 \\ -1.2 & 4.3 \\ 1.8 & 2.9 \\ -2.3 & -3.1 \\ 4.1 & 2.8 \end{bmatrix}$$

Desarrollar ejemplo en la pizarra

Bibliografía

1. Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Vol. 200, pp. 1-10). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

2. Karaboga, D., Basturk, B. (2007). Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems. In: Melin, P., Castillo, O., Aguilar, L.T., Kacprzyk, J., Pedrycz, W. (eds) Foundations of Fuzzy Logic and Soft Computing. IFSA 2007. Lecture Notes in Computer Science(), vol 4529. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-72950-1_77.

3. S. Olariu, A. Y. Zomaya. 2006. Handbook of Bioinspired Algorithms and Applications. Chapman and Hall/CRC, ISBN 978-1-58488-475-0.

4. Slowik, A. (Ed.). (2020). Swarm Intelligence Algorithms: A Tutorial.

¡ Gracias !

Thanks !

Obrigado

Xie xie ni

Domo arigatou

Спасибо

Merci

Grazie