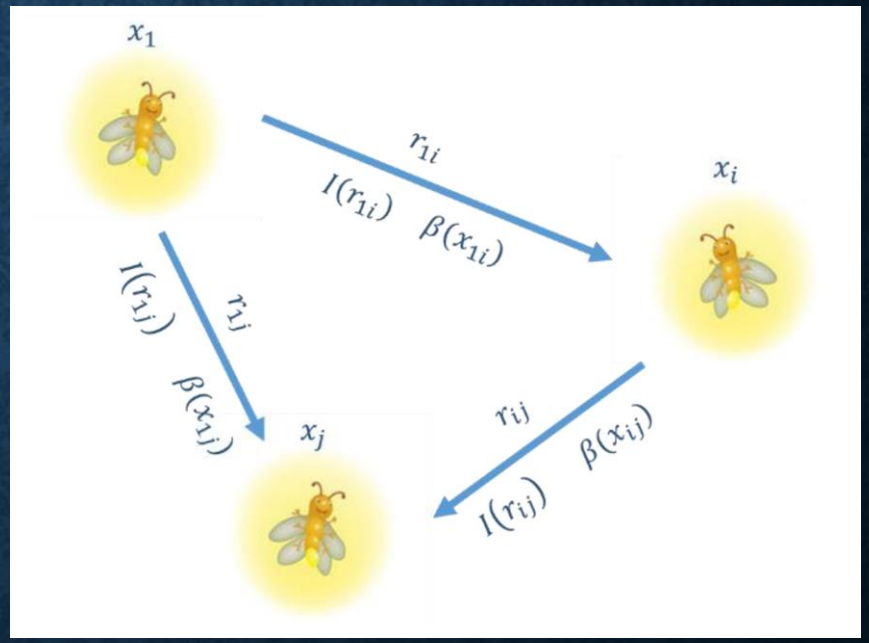
Firefly Algorithm

Práctica Alternativa

Javier Ramírez Pulido

DESCRIPCION DE FIREFLY ALGORITHM

- Autor: Yang, 2008
- Se basa en el parpadeo de las luciérnagas
  - Luciérnagas unisex. Todas pueden a traer a todas
  - Atracción proporcional a su brillo y se acercan a las mas brillantes. La mas atractiva se mueve aleatoria
  - La intensidad se obtiene de la función objetivo
- El atractivo depende desde donde sea mirada
- r: Distancia / I: Intensidad / β = Atractivo

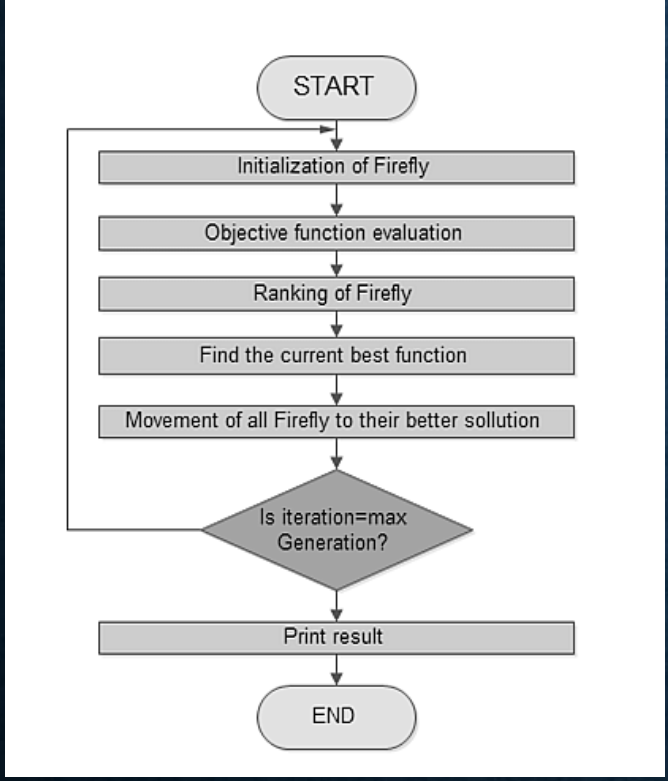


FORMULAS

- Intensidad: $I = Io * e^{-\gamma r}$
- Atractivo: $\beta = \beta o * e^{-\gamma r^2}$
- Distancia: $ri,j = ||xi - xj|| = \sqrt{\sum_{k=1}^{d} \left(x(i,k) - x(j,k)\right)^2}$
- Movimiento: $xi = xi + \beta o * e^{-\gamma r^2} (xi - xj) + \alpha(\varepsilon i)$

- γ -> Coeficiente de absorción lumínica
- d -> dim (10, 30 o 50)
- α -> factor de aleatorización

PSEUDOCODIGO



Initialize all the parameters $(\alpha, \beta, \gamma, n)$;
Initialize randomly a population of n fireflies;
Evaluate the fitness of the initial population at $\mathbf{x}_i$ by $f(\mathbf{x}_i)$ for $i = 1, \ldots, n$;
while *(t < MaxGeneration)* **do**
  for *All fireflies* $(i = 1 : n)$ **do**
    for *All other fireflies* $(j = 1 : n)$ *(inner loop)* **do**
      if *Firefly j is better/brighter than i* **then**
        Move firefly i towards j according to Eq. (1);
      end
    end
    Evaluate the new solution and accept the new solution if better;
  end
  Rank and update the best solution found so far;
  Update iteration counter $t \leftarrow t + 1$;
  Reduce α (randomness strength) by a factor;
end

Algorithm 1: Firefly algorithm.

HIBRIDACION

- **algoritmo FFA_hibrido**
- **variables como parámetros**
- tipo entero dim
- **variables locales**
- tipo entero constante poblacion_inicial <- 50, cantidad_excesiva_elementos <- 100, evaluaciones_bl_maximas <- 100
- tipo double constante delta <- 0.4
- vector tipo Luciernaga población
- tipo entero evaluaciones <- cantidad_excesiva_elementos
- **inicio**

1. poblacion <- crea_poblacion(poblacion_inicial, cantidad_excesiva_elementos, dim, generador aleat.)
2. Mientras evaluaciones sea menor que 10000 x dim
    a. Para cada luciérnaga Lu de la población
        i. Aplicar BL(Lu.solucion, Lu.fitness, delta, evaluaciones_bl_maximas, -100, 100, generador)
    b. Mover_luciernaga(población, dim, evaluaciones)

Solis Wets (BL)

MEJORA DE LA METAHEURISTICA

- Aplicada cada 3 iteraciones (recorridos completos a la población acercando luciérnagas)
- Comprobamos la distancia entre los 10 mejores elementos de la población
- Cada pareja de luciérnagas "muy cercana" tiene un descendente

Genera menos de 10 hijos

Entran todos sustituyendo

a los peores de la población

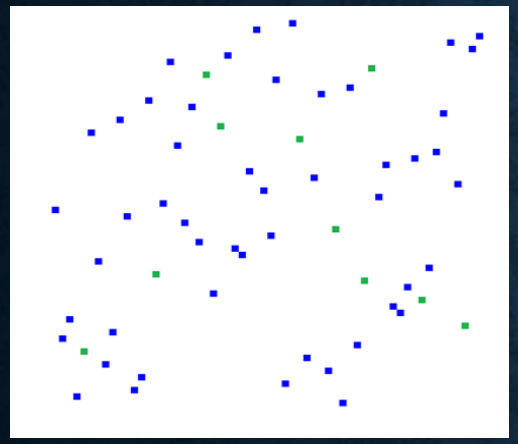
Genera mas de 10 hijos

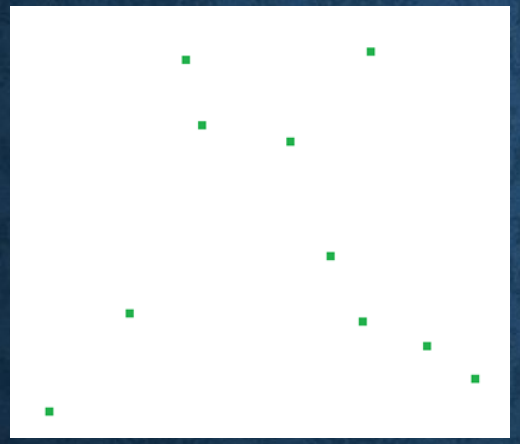
Entran los 10 mejores hijos

sustituyendo a los 10 peores

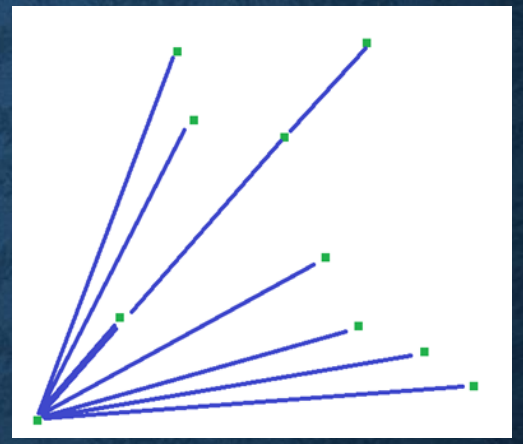
MEJORA DE LA METAHEURISTICA

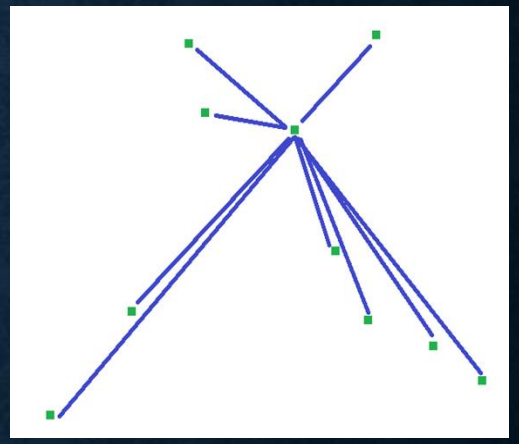
Seleccionamos 10 mejores



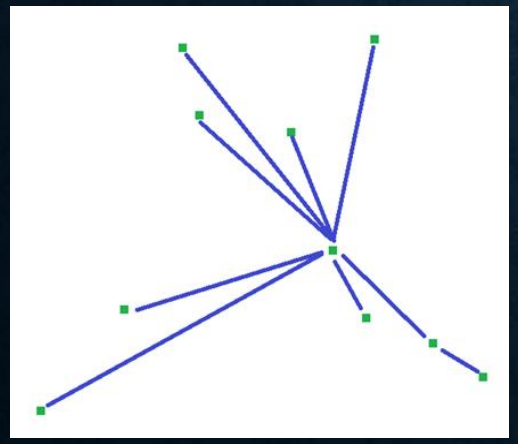


Calculamos la distancia de todos con todos…

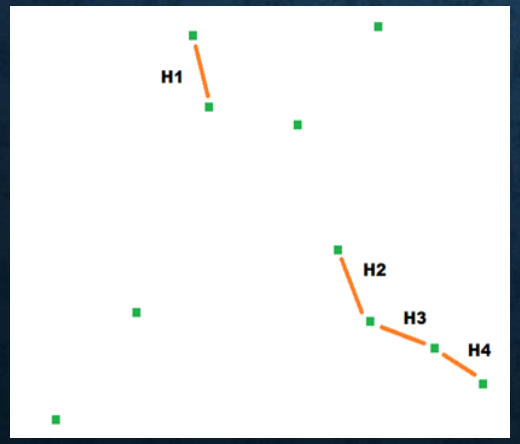




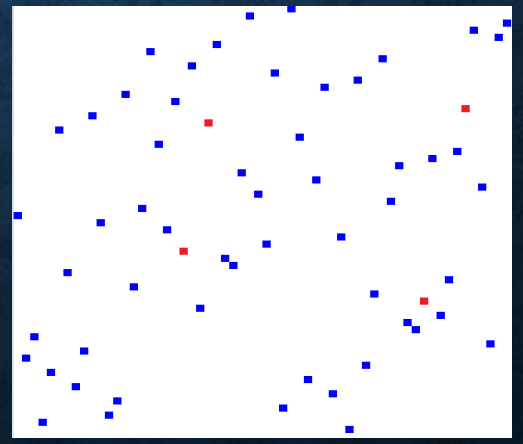
(Se muestran 3 ejemplos)



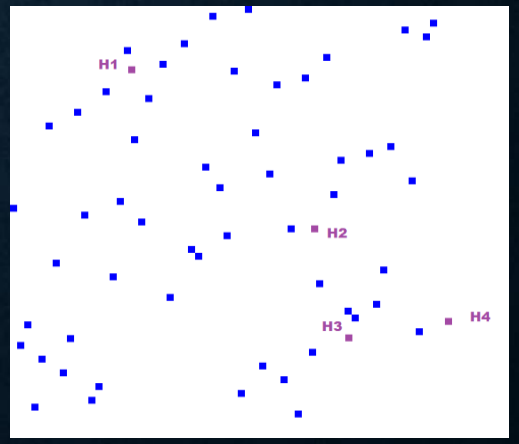
Los que están muy cerca



Ubicamos los peores



Sustituimos



MEJORA DE LA METAHEURISTICA

- **algoritmo FFA_mejorado**
- **variables como parámetros**
- tipo entero dim
- **variables locales**
- tipo entero constante poblacion_inicial <- 50, cantidad_excesiva_elementos <- 100, evaluaciones_bl_maximas <- 100
- tipo double constante delta <- 0.4
- vector tipo Luciernaga población, población_auxiliar
- tipo entero evaluaciones <- cantidad_excesiva_elementos, contador <- 0

inicio

1. poblacion <- crea_poblacion(poblacion_inicial, cantidad_excesiva_elementos, dim, generador aleat.)
2. Mientras evaluaciones sea menor que 10000 x dim
    a. Si contador es multiplo de 3
        i. Ordenamos poblacion
        ii. Para cada i desde 0 hasta 9
            1. Para cada j desde i+1 hasta 10
                a. (double) distancia <- 0
                b. Para cada el3 desde 0 hasta dim
                    i. distancia <- distancia + (el1.solucion(el3) - el2.solucion(el3))^2
                c. distancia <- raíz cuadrada(distancia)
                d. Si distancia es menor que 10xdim
                    i. Reproducción(población(i), población(j), población_auxiliar)
        iii. Para cada elemento de la poblacion_auxiliar o los 10 primeros si hay mas
            1. Sustituir elemento del final de poblacion por uno de poblacion_auxiliar
    b. Para cada luciérnaga Lu de la población
        i. Aplicar BL(Lu.solucion, Lu.fitness, delta, evaluaciones_bl_maximas, -100, 100, generador)
    c. Mover_luciernaga(población, dim, evaluaciones)

Bibliografía

Firefly algorithm – Wikipedia

(PDF) Firefly Algorithm, Stochastic Test Functions and Design Optimisation (researchgate.net)

Algoritmo determinista - Wikipedia, la enciclopedia libre

Firefly algorithm (slideshare.net)

Performance evaluation of firefly algorithm with variation in sorting for non-linear benchmark problems (scitation.org)

GII - METAHEURÍSTICAS

Práctica alternativa

Javier Ramírez Pulido
javierramirezp@correo.ugr.es

Curso 2020/21