

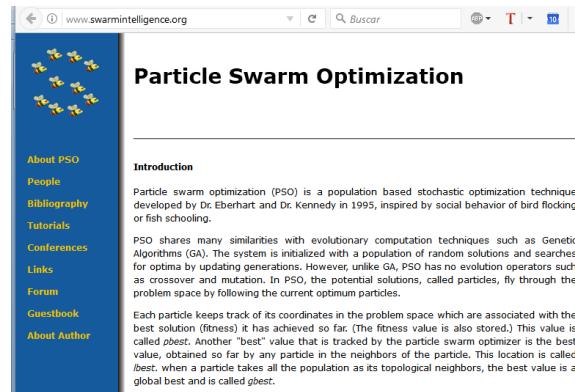
# Tema- 5 Inteligencia de Enjambre (Swarm intelligence)

(Inteligencia Social, Inteligencia Colectiva, Metaheurísticas Bio-Inspiradas)

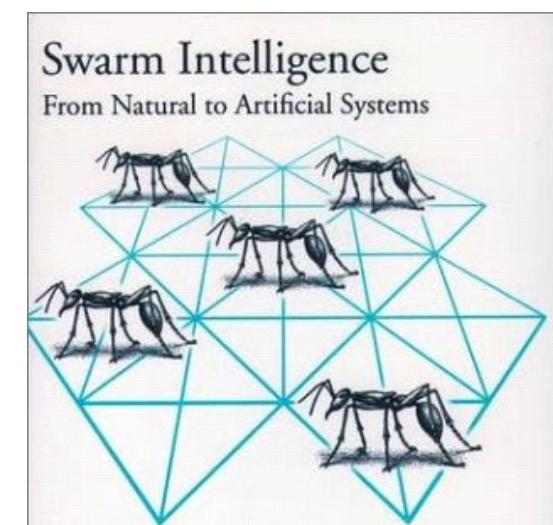
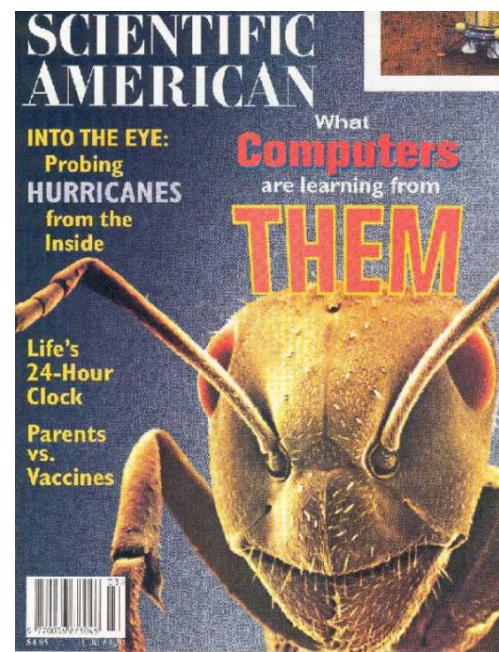
- **Algoritmo de las Hormigas** (*Ant Colony Optimization - ACO*),
- **Enjambre de partículas** (*Particle Swarm Optimization - PSO*)
- **Otras variantes**

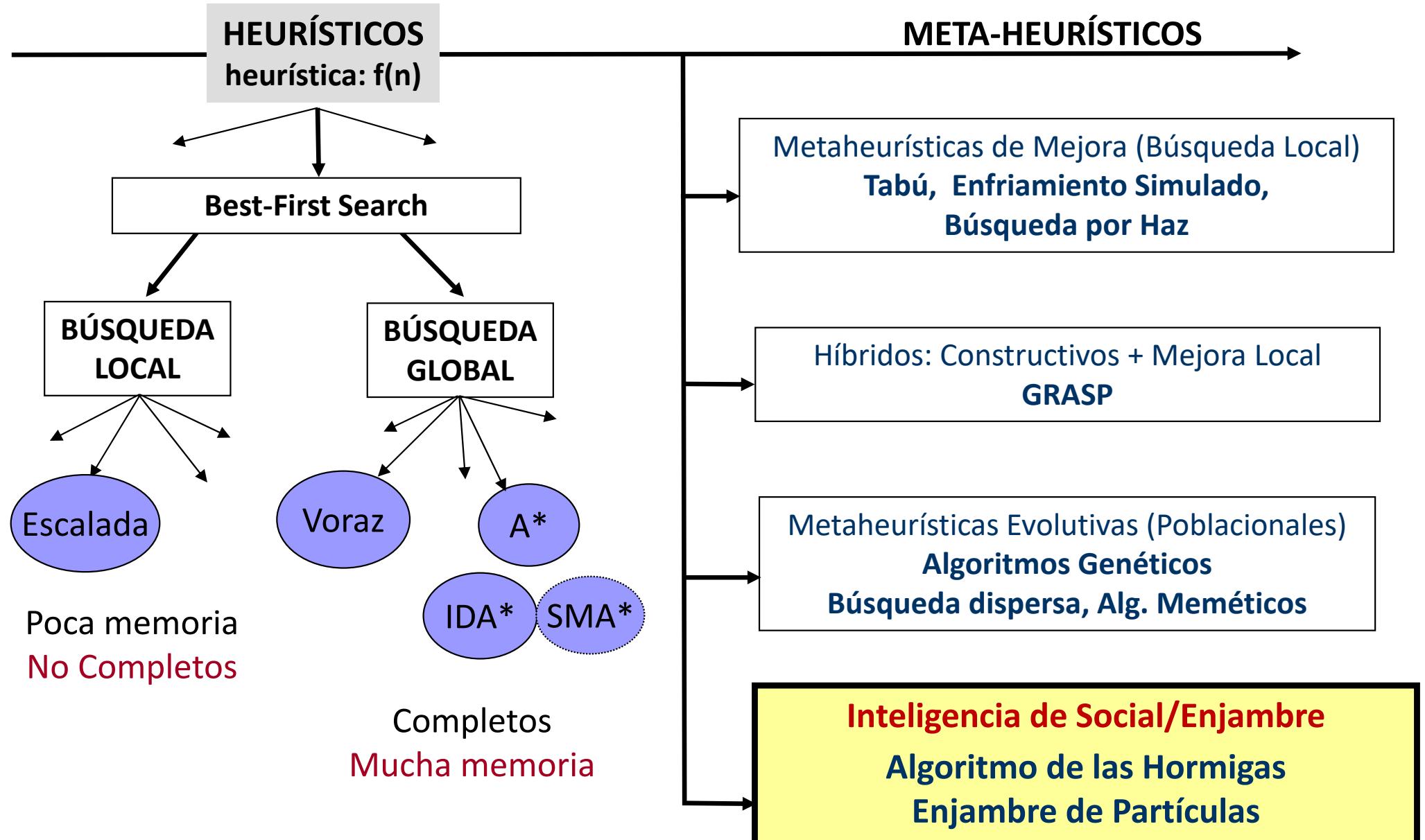


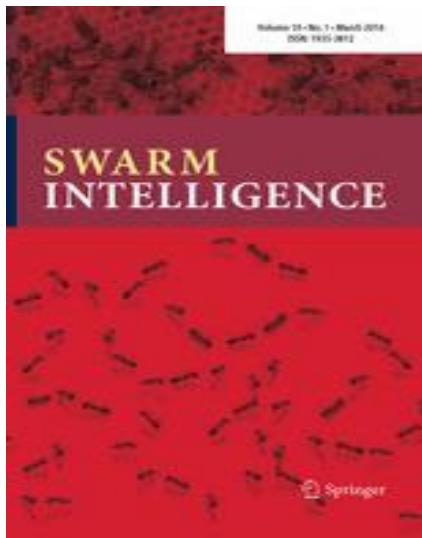
<http://www.swarmintelligence.org/>



The screenshot shows the homepage of the Swarm Intelligence website. On the left, there's a sidebar with links: About PSO, People, Bibliography, Tutorials, Conferences, Links, Forum, Guestbook, and About Author. The main content area has a title "Particle Swarm Optimization". Below it is an "Introduction" section with text about PSO being a population-based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling. It also describes the algorithm's similarity to Genetic Algorithms (GA) and how it differs from GA.







Artificial Intelligence Review (2020) 53:5589–5635  
<https://doi.org/10.1007/s10462-020-09829-2>

**Performance comparison of five metaheuristic nature-inspired algorithms to find near-OGRs for WDM systems**

Artif Intell Rev (2019) 52:2191–2233  
<https://doi.org/10.1007/s10462-017-9605-z>

## Metaheuristic research: a comprehensive survey

Artificial Intelligence Review (2020) 53:753–810  
<https://doi.org/10.1007/s10462-018-09676-2>

## From ants to whales: metaheuristics for all tastes

[ACO Timetabling Tool.pdf](#)

[Ant Colony Optimization Book.pdf](#)

[Aplicaciones Algoritmo Hormigas.pdf](#)

[Aplicaciones Metaheuristica POO.pdf](#)

[APPLICATION IN POWER SYSTEM PROBLEMS.pdf](#)

[Comparison of 5 novel nature inspired metaheuristic.pdf](#)

[From ants to whalesmetaheuristics for all tastes \(2020\).pdf](#)

[Metaheuristic Review 2019.pdf](#)

[multi-objective particle swarm optimization.pdf](#)

[PARTICLE SWARM OPTIMIZATION.pdf](#)

[Particle Swarm Optimization Algorithms.pdf](#)

[PSO for engineering.problems.pdf](#)

[PSO-Survey.pdf](#)

[PSO Viajante Multiobjetivo.pdf](#)

[State of the art review of intelligent scheduling.pdf](#)

[Survey PSO para timetabling.pdf](#)

Principios de la Inteligencia de Enjambre: Descentralización, Auto-organización, Estigmería

Basada en el comportamiento colectivo de sistemas descentralizados y auto-organizados.

- Un sistema auto-organizado mantiene una estructura (funcionalidad) *independiente del exterior*, o de un *control global*, y que proviene de las interacciones en los *niveles más bajos*.

*Múltiples interacciones      Realimentación positiva /negativa*

*Amplificación en la selección de opciones aleatorias*

- En un sistema descentralizado, el control esta completamente distribuido entre sus elementos componentes



Estigmería: stigma (estimulación) + ergon (trabajo)

#### *Interacción por el entorno:*

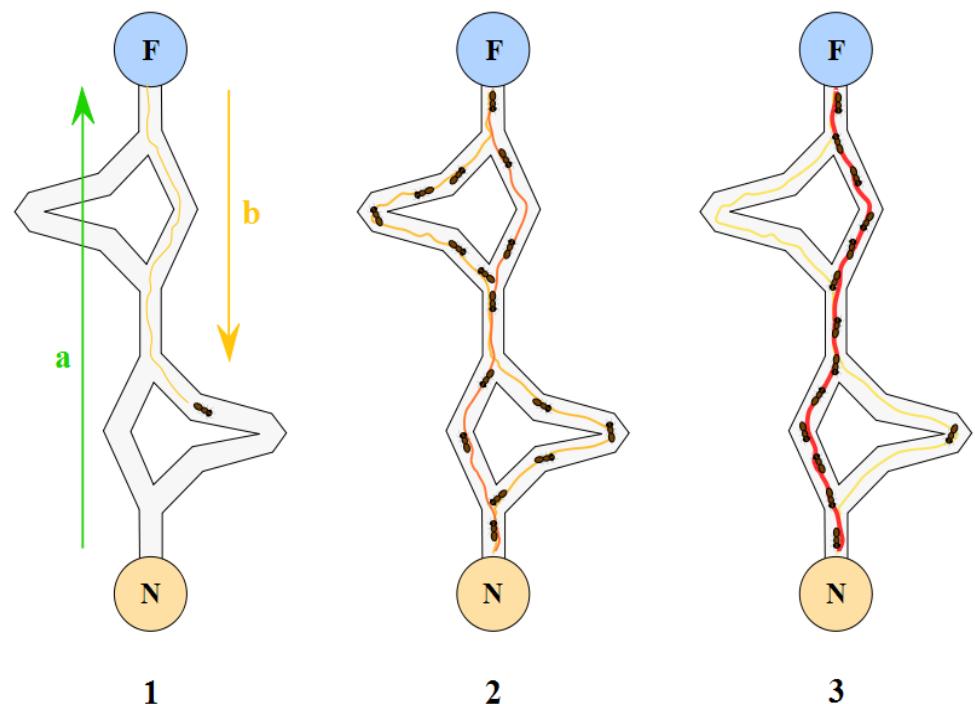
- Los agentes actúan como respuesta al entorno y a otros agentes, mediante reglas de conducta muy simples.
- Los agentes modifican (indirectamente) el entorno: Interaccionan (colaboran) entre ellos a través del entorno (medio físico).
- El entorno actúa como una memoria de trabajo.
- La interacción puede ser realizada por cualquier individuo.
- Las reglas simples de la respuesta al entorno pueden crear diferentes patrones de conducta.

## Inteligencia de enjambre: Basada en la observación de la naturaleza (**bio-inspirada**)

- Se basa en una población (sociedad) de agentes individuales, auto-organizados y sin control centralizado.
- Los individuos interactúan entre sí y con el entorno, de forma localizada y siguiendo reglas muy simples.
- Hay un cierto seguimiento al líder, a los buenos exploradores, o a la costumbre social.
- A diferencia de las metaheurísticas poblacionales, los individuos no son eliminados (no hay reemplazo), no hay control centralizado y los individuos suelen estar en constante movimiento.

### Resultado:

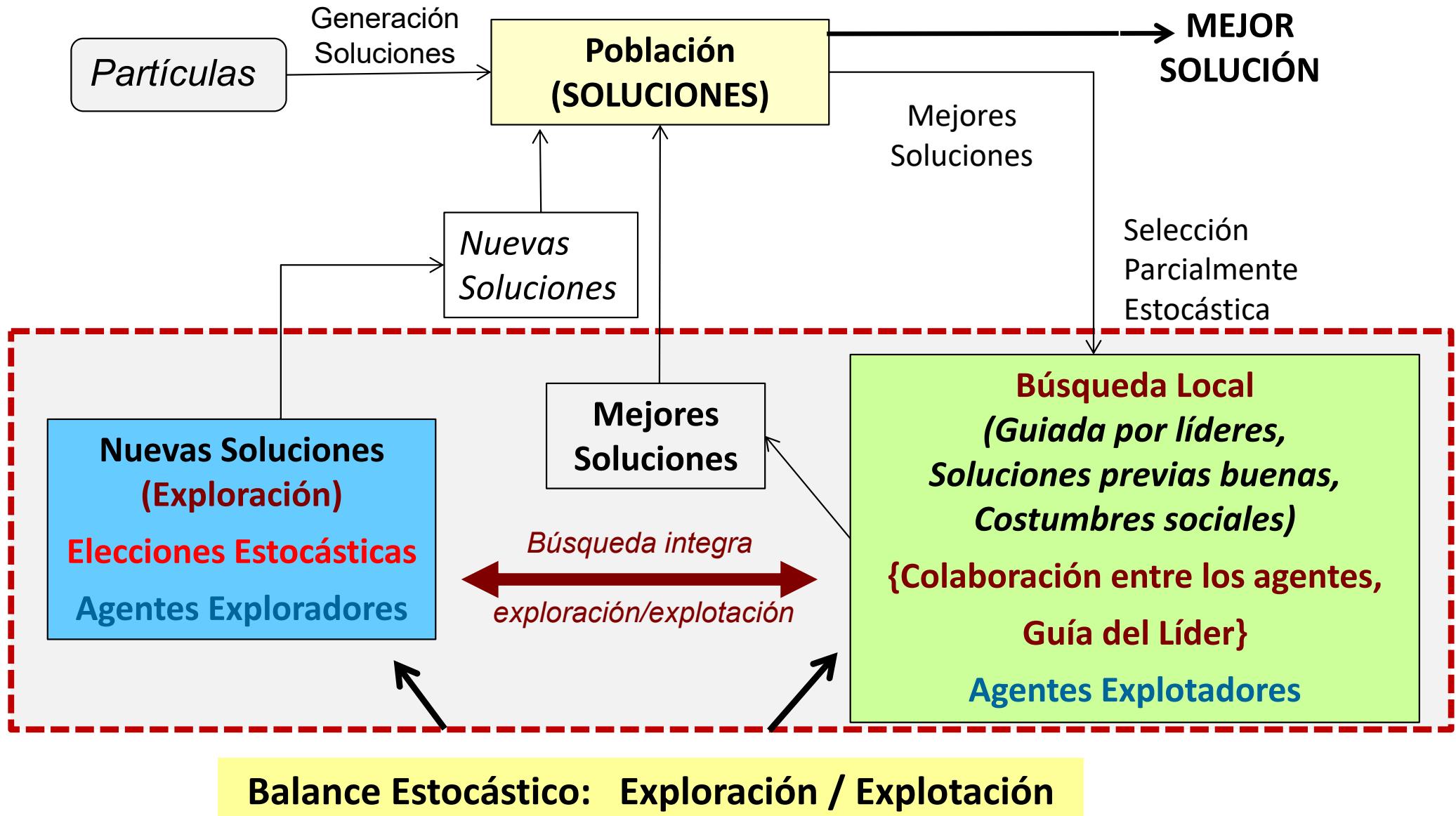
- La interacción entre los agentes induce una cierta inteligencia colectiva (desconocida incluso para los propios individuos).
- La funcionalidad del sistema global transciende el repertorio de las conductas individuales.
- La productividad del sistema (colonia) es mayor que el esfuerzo de sus miembros
- La respuesta conjunta del grupo es robusta, flexible y adaptativa con respecto a los cambios en el entorno (**sistemas auto-inmunes**)



# Esquema General

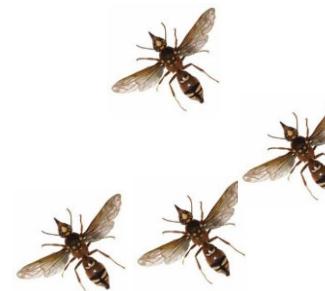
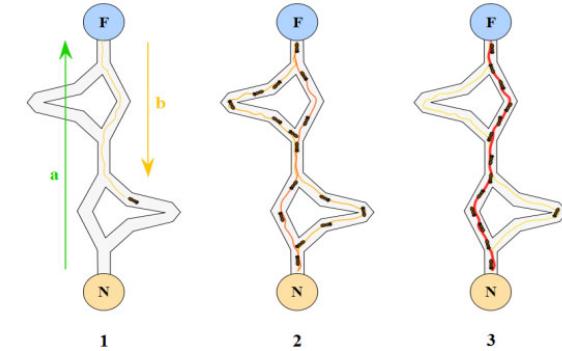
*Las partículas generan las soluciones (no son las soluciones)*

INICIO



## Principales Variantes:

- **Algoritmo de las Hormigas**  
(Ant Colony Optimization),



- **Enjambre de partículas**  
(Particle Swarm Optimization)  
Bandada de Pájaros, Bancos de Peces, etc.



*Otras metaheurísticas (enjambre/bioinspiradas):*

Algoritmo de las abejas,

Algoritmo de los murciélagos (Bat algorithm),

Algoritmo de las luciérnagas (glowworm swarm optimization, GSO),

Caída inteligente de gotas de agua (Intelligent Water Drops, IWD),

Etc., etc., etc., etc.....



# Algoritmo de las Hormigas

(*Ant Colony Optimization*) M. Dorigo, 1992



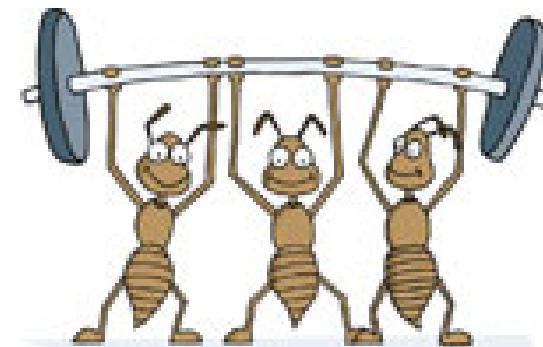
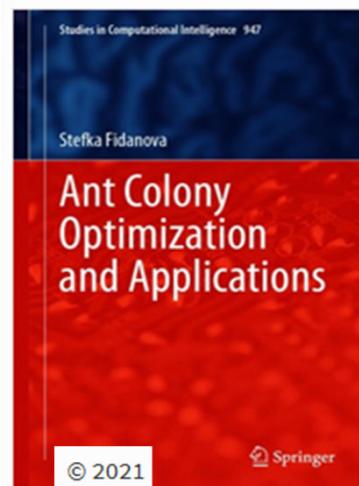
[http://www.scholarpedia.org/article/Ant\\_colony\\_optimization](http://www.scholarpedia.org/article/Ant_colony_optimization)

Google Académico

<http://www.midaco-solver.com/>

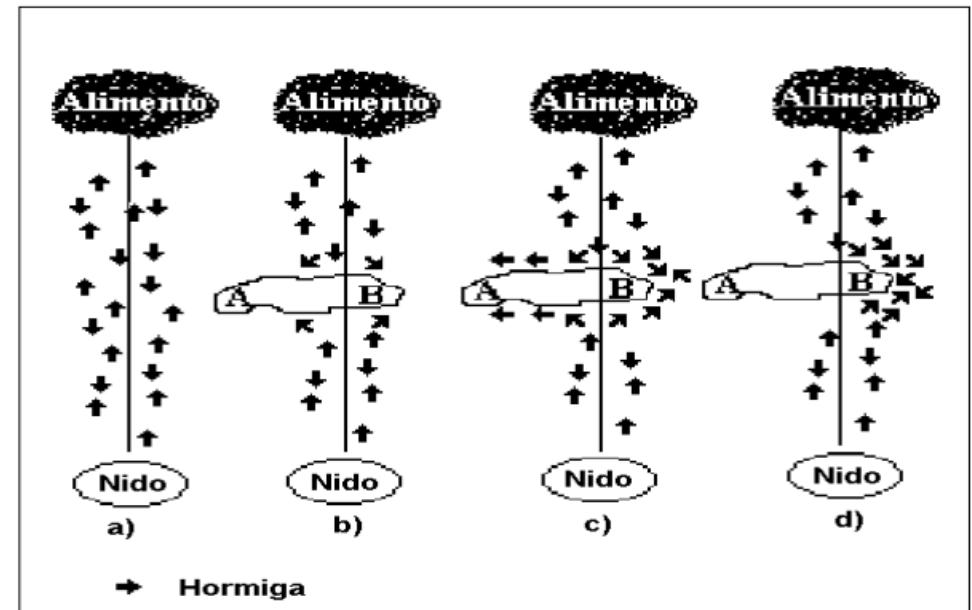
<http://www.aco-metaheuristic.org/>

Artículos en Poliformat



## Basado en la observación de la naturaleza: Hormigas en la búsqueda de comida.

- Las hormigas recorren diversos caminos buscando el alimento. Eventualmente, encuentran (si existen) varias sendas (soluciones).
- Las hormigas dejan una substancia (feromonas), marcando el rastro que recorren.
  - La feromona se evapora en el tiempo.
  - La feromona en cada punto de un camino dependerá del número (caudal) de hormigas que han dejado su rastro.
- Las hormigas, en la búsqueda del camino, siguen el rastro de las feromonas.
- El camino más corto tendrá el mayor número de paso de hormigas: más caudal  $\Rightarrow$  mayor feromona
- Al final, ***todas las hormigas siguen el camino óptimo.***
- Aplicable para obtener sendas optimizadas en grafos (y problemas equivalentes).



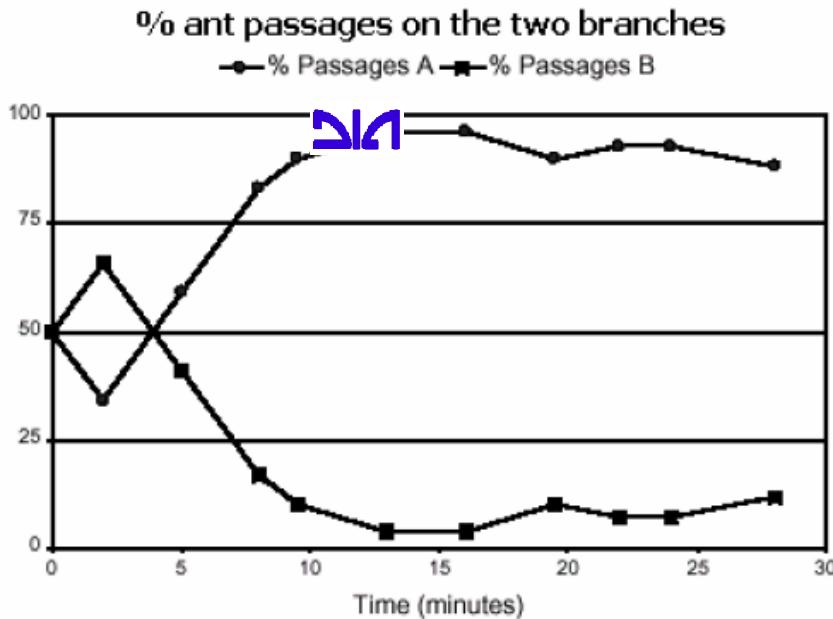
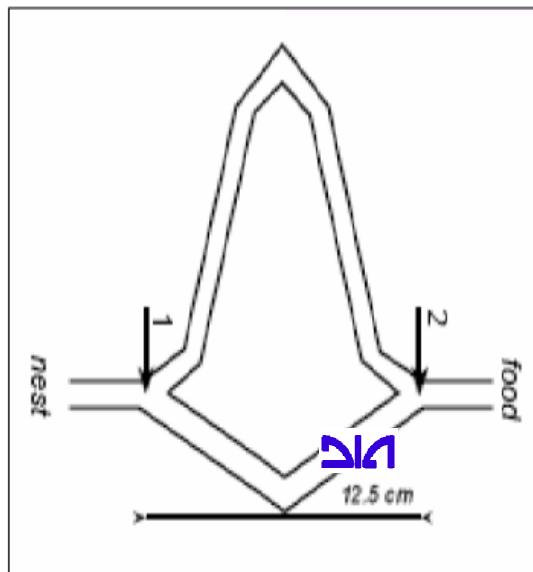
Al ser interrumpido el camino directo, la mitad de las hormigas se dirigen por A y la otra mitad por B.

- B es la menor distancia para rodear el obstáculo, luego mayor cantidad de hormigas y depositan más feromonas.

Las hormigas siguen el rastro de las feromonas y elegirán rodear el obstáculo por B, lo que a su vez depositará más feromonas en ese trayecto.

Por realimentación positiva, el camino B será seleccionado como el camino más corto.

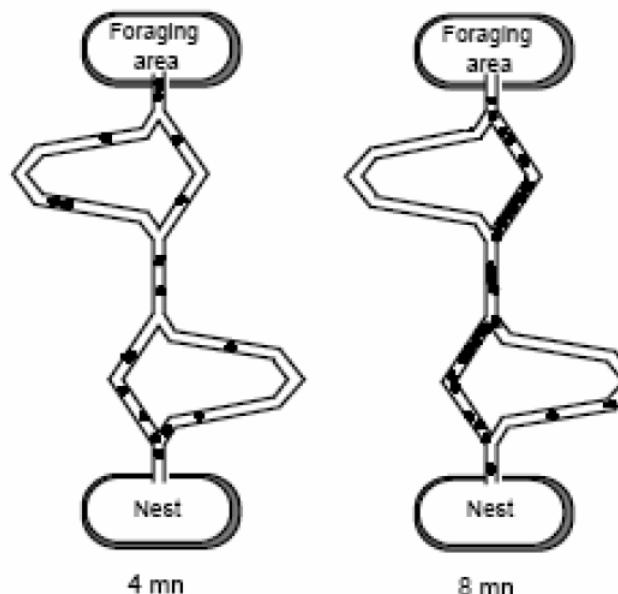
## Antecedentes..... Estudio sobre las Hormigas ‘biológicas’ (Goss et al., 1989, Deneubourg et al., 1990).



*Self-organized shortcuts in the Argentine ant Goss et al., 1989*

Estudiaron el porcentaje de hormigas que tomaban cada camino.

Inicialmente los caminos lo elegían de manera arbitraria, pero casi todas acababan siguiendo el mismo camino.



A partir de sus experimentos, construyeron un **modelo estocástico sencillo** para describir la dinámica de las hormigas:

**$P_{i,a}$** : La probabilidad de que una hormiga llegue a la intersección  $i$  y seleccione la rama  $a$  en un instante  $t$  es:

$$p_{i,a} = \frac{[k + \tau_{i,a}]^\alpha}{[k + \tau_{i,a}]^\alpha + [k + \tau_{i,a'}]^\alpha}$$

donde

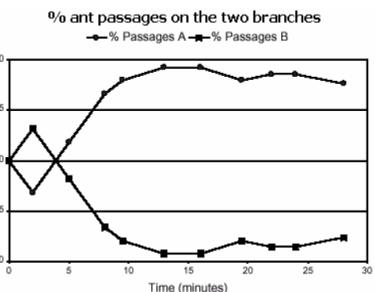
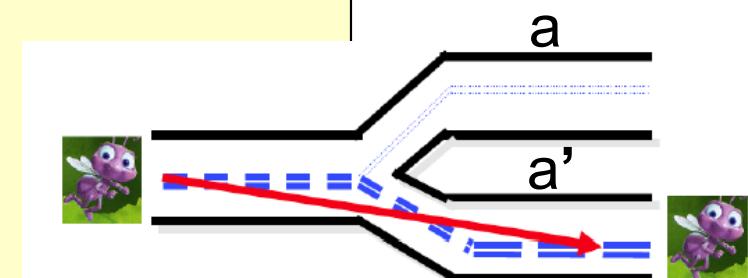
$\tau_{i,a}$  es la concentración de feromona en la rama  $a$  de la intersección  $i$ ,

$\tau_{i,a'}$  es la concentración de feromona en la otra rama de la intersección  $i$ ,

$\alpha = 2$  (obtenido de diversos experimentos), y

$k$  es el tiempo que se tarda en atravesar la rama  $a$ .

La concentración de feromona depende del caudal de hormigas



- La **metaheurística ACO** reproduce el comportamiento de las hormigas reales en una colonia artificial de hormigas para resolver problemas complejos de camino mínimo
- Cada **hormiga artificial** (agente) es un mecanismo probabilístico de **construcción** de soluciones al problema que combina:
  - Rastros de **feromona (artificiales)  $\tau$**  que cambian con el tiempo para reflejar la **experiencia** adquirida por los agentes previos en la resolución del problema (*exploración*).
  - Información **heurística  $\eta$**  sobre el problema.

## Aplicación al Problema del Viajante

- El objetivo es encontrar la senda más corta que recorre una serie de ciudades.
- Cada hormiga hace una senda alternativa de las posibles:  
Cada hormiga debe visitar todas las ciudades, y cada ciudad exactamente una vez,
- En cada iteración, **cada hormiga decide pasar de una ciudad a otra**, en base a:
  - a) Exploración Propia: Una ciudad lejana tiene menos posibilidades de ser elegida frente a una cercana (por **visibilidad: heurística  $\eta$** ),
  - b) Explotación/Colaboración: Cuanto más marcado esté el camino (más intensa es la **feromonía:  $T$** ) entre dos ciudades, mayor probabilidad de que sea elegido,
- Cada hormiga, tras completar su viaje, ha depositado más feromonas en sus trayectos si su viaje ha sido más corto (carga/longitud).
- Las feromonas se evaporan con el tiempo.

**Procedure Hormigas**

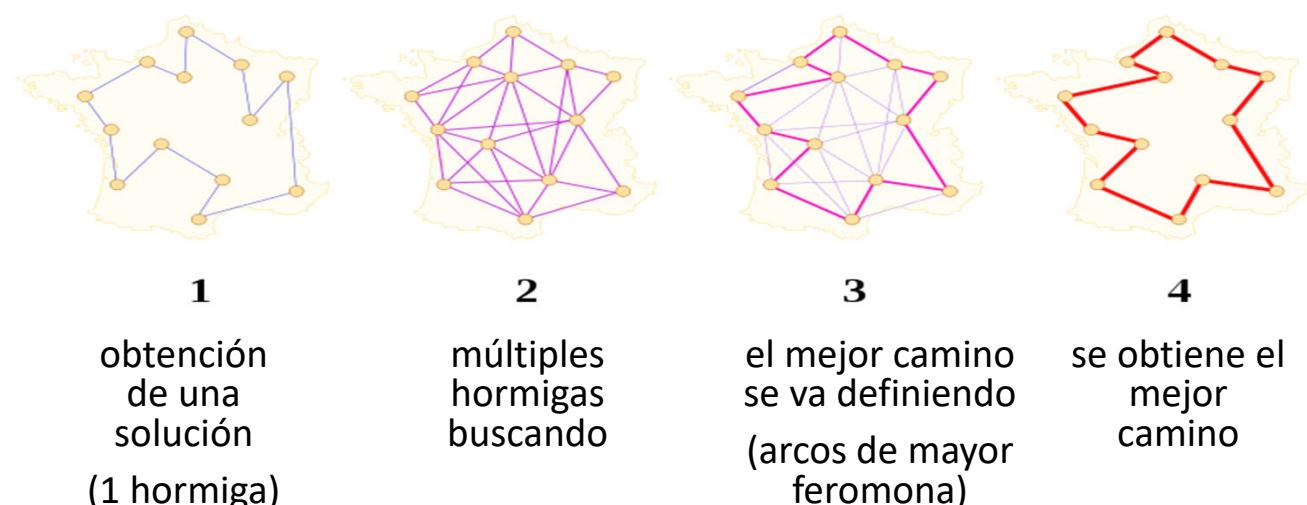
**while** (no-fin)

**GenerarSoluciones()**

**Actualización-Feromonas()**

**end while**

**end procedure**



# Algoritmo de las Hormigas-1

Procedure Hormigas

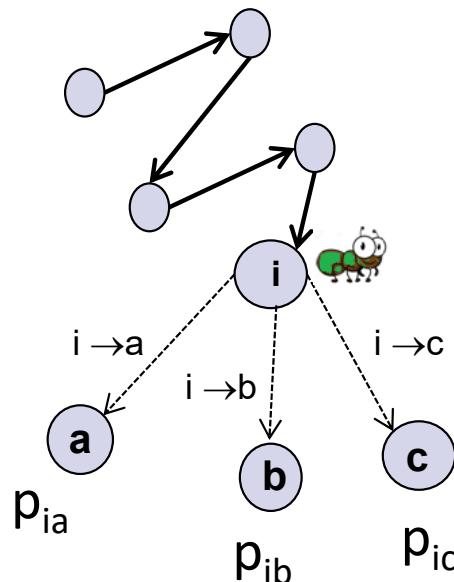
while (no-fin)

GenerarSoluciones()

Actualización-Feromonas()

end while

end procedure



**GenerarSoluciones ()**

;aplicación al viajante

- Cada hormiga obtendrá un camino: *Soluciones* ≡ *Caminos*
- En cada iteración, las hormigas eligen un trayecto  $i \rightarrow j$ .  
(si N ciudades, entonces N-1 iteraciones)

Para cada arco  $i$  ( $i=1, N-1$ ) ;  $N$ : ciudades

Para cada Hormiga  $h$  ( $h=1, H$ ) ;  $H$ : Hormigas

Hormiga  $h$  selecciona siguiente trayecto (ciudad) en su senda

- Criterio de Selección de Trayecto  
Una hormiga irá de  $i \rightarrow j$ , con probabilidad

$$p_{i,j} = \frac{(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}{\sum(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}$$

← Valores del arco  $i \rightarrow j$   
← Sumatorio para todos los nodos del grafo

Donde,

$\tau_{ij}$  es la feromona de arco  $i \rightarrow j$ , con una influencia  $\alpha$

$\eta_{ij}$  es la visibilidad del arco (típicamente,  $1/\text{distancia}$ ), con influencia  $\beta$

# Algoritmo de las Hormigas-2

## Procedure Hormigas

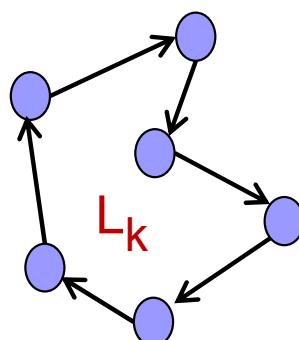
while (no-fin)

    GenerarSoluciones()

Actualización-Feromonas()

end while

end procedure



- Con un buen diseño, los caminos de todas las hormigas acaban convergiendo.
- El proceso acaba tras un número dado de iteraciones, o cuando se cumple un criterio de terminación

## Actualización-Feromonas ()

Las feromonas en los arcos se actualizan tras la obtención de un camino (solución) por todas las hormigas.

La feromona de cada arco  $i \rightarrow j$  ( $\tau_{ij}$ ) se actualiza de la forma:

$$\tau_{ij} = (1-\rho) \tau'_{ij} + \Delta\tau_{ij}$$

Donde,

$\rho$  Es la tasa de evaporación, que se aplica a la feromona previa  $\tau'_{ij}$

$\Delta\tau_{ij}$  Es la feromona que se deposita tras los caminos encontrados:

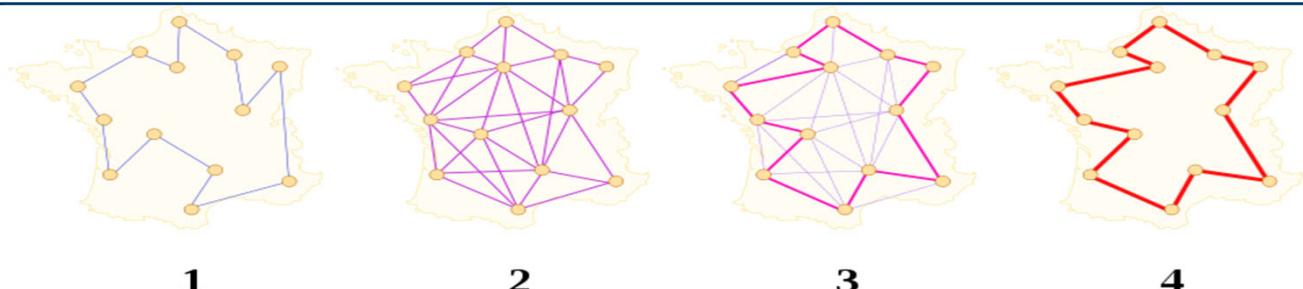
$$\Delta\tau_{ij} = \sum_{h=1,H} \Delta\tau^h_{ij}$$

siendo  $\Delta\tau^h_{ij}$  la feromona dejada por hormiga  $h$  en  $i \rightarrow j$ :

=  $1/L_h$  si la hormiga  $h$  atraviesa el arco

= 0 en otro caso

$L_h$  es el coste (distancia) de todo el camino obtenido por la hormiga- $h$



# Implementación Algoritmo de las Hormigas: 3 matrices

**Matriz de feromonas:** Feromonas del arco  $i \rightarrow j$

Las feromonas de cada celda  $(i,j)$   
se reactualizan tras cada iteración:

$$\tau_{ij} = (1-\rho) \tau'_{ij} + \Delta\tau_{ij}$$

**Matriz de Visibilidades:**

Distancia del arco  $i \rightarrow j$

	1	2	3	4	5	...	n-1	n
1	██████					.....		
2		██████				.....		
3			██████			.....		
4				██████		.....		
.....	.....	.....	.....	.....	.....	.....	.....	.....
n-1						.....	██████	
n						.....		██████

**Matriz de Probabilidades:**

Probabilidades del arco  $i \rightarrow j$

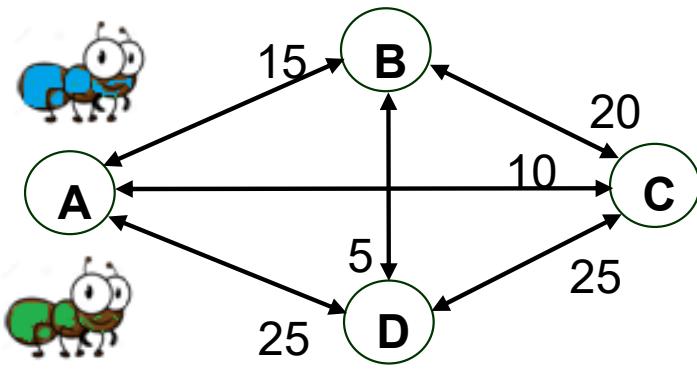
Las probabilidades de cada  
celda  $(i,j)$  se reactualizan tras  
cada iteración

$$p_{i,j} = \frac{(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}{\sum(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}$$

**Parámetros:** Hormigas,  
 $\rho$  (evaporación),  
 $\alpha, \beta$  (pesos),  $\tau_{\text{INICIAL}}$

$\uparrow \rho, \beta, \tau_{\text{INICIAL}} \Rightarrow$  Exploración ,  
 $\downarrow \rho, \uparrow \alpha \Rightarrow$  Explotación ( $\approx$  feromona colaborativa)

## Ejemplo: Problema del Viajante



$$(\eta_{ij} = 1/l_{ij})$$

	B	C	D
A	1/15	1/10	1/25
B		1/20	1/5
C			1/25

Matriz de Visibilidades

$$\tau_{ij} = (1-\rho) \tau'_{ij} + \Delta\tau_{ij}$$

	B	C	D
A	0,1	0,1	0,1
B		0,1	0,1
C			0,1

Matriz de Feromonas (inicial)

### Parámetros:

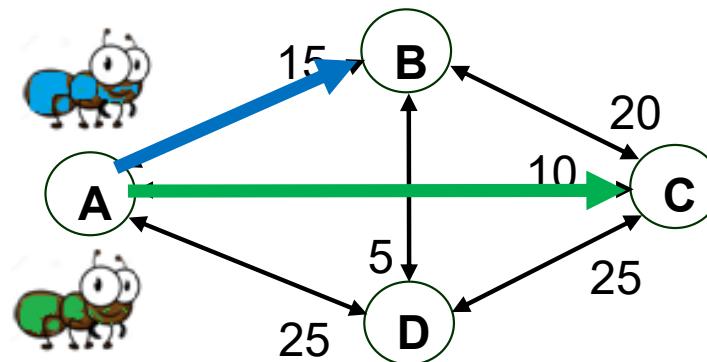
Hormigas  $\{h_i\}=2$ ,  $\rho=0,5$ ,  $\alpha=0,4$ ,  $\beta=0,6$ ,  $\tau_{INICIAL}=0,1$

$$p_{i,j} = \frac{(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}{\sum(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}$$

	B	C	D
A	0,332	0,423	0,244
B		0,222	0,51
C			0,255

Matriz de Probabilidades

- h1, en A elige C ( $p=0,423$ )  
h2, en A elige B ( $p=0,332$ ),



$$p_{i,j} = \frac{(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}{\sum(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}$$

	B	C	D
A	0,332	0,423	0,244
B		0,222	0,51
C			0,255

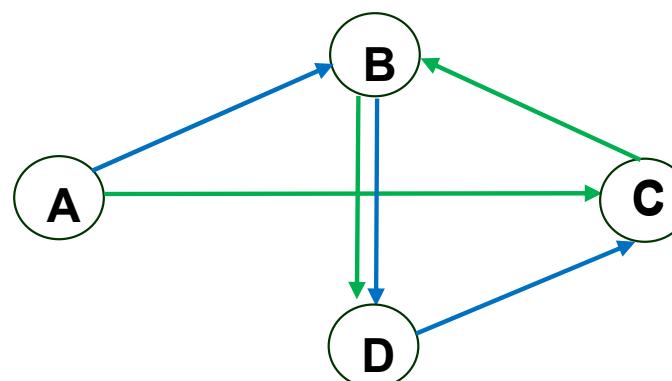
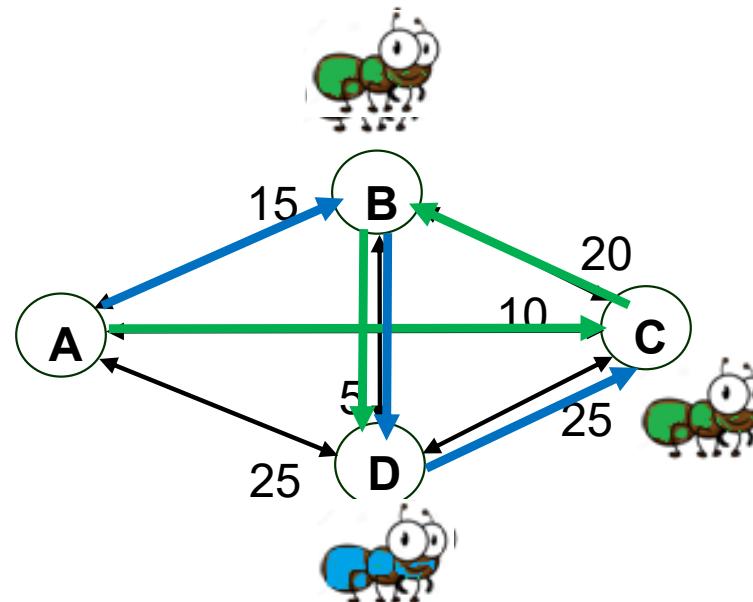
### Matriz de Probabilidades

h1 (A-C) elige B ( $P=0,222$ )  
 h2 (A-B) elige D ( $p=0,51$ )

h1 (A-C-B) elige D  
 h2 (A-B-D) elige C

### Parámetros:

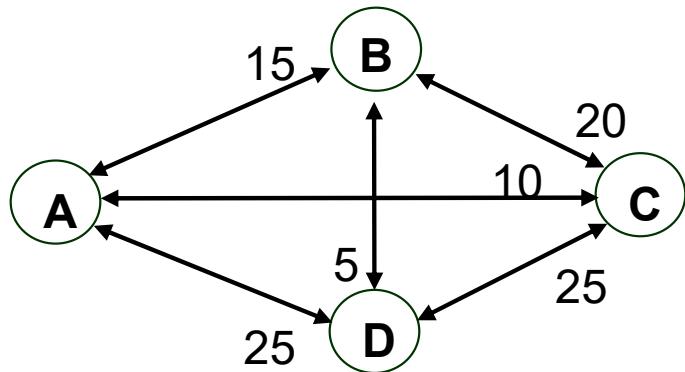
Hormigas  $\{h_i\}=2$ ,  $\rho=0,5$ ,  $\alpha=0,4$ ,  $\beta=0,6$ ,  $\tau_{\text{INICIAL}}=0,1$



$$L1: 35+25=60$$

$$L2: 45+10=55$$

## Actualización de feromonas



$$(\eta_{ij} = 1/l_{ij})$$

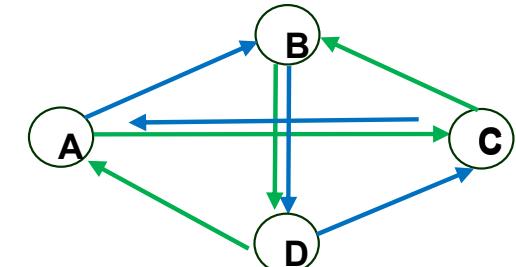
	B	C	D
A	1/15	1/10	1/25
B		1/20	1/5
C			1/25

Matriz de Visibilidades

Hormigas=2,  $\rho=0,5$ ,  $\alpha=0,4$ ,  $\beta=0,6$ ,  $\tau_{INICIAL}=0,1$

L1: 60. En cada arco que pasa deja  $1/60$

L2: 55. En cada arco que pasa deja  $1/55$



	B	C	D
A	0,332	0,423	0,244
B		0,222	0,51
C			0,255

	B	C	D
A	0,324	<b>0,436</b>	0,238
B		0,216	0,525
C			0,24

Matriz de Probabilidades (actualizada)

	B	C	D
A	0,07	0,08	0,07
B		0,07	0,08
C			0,06

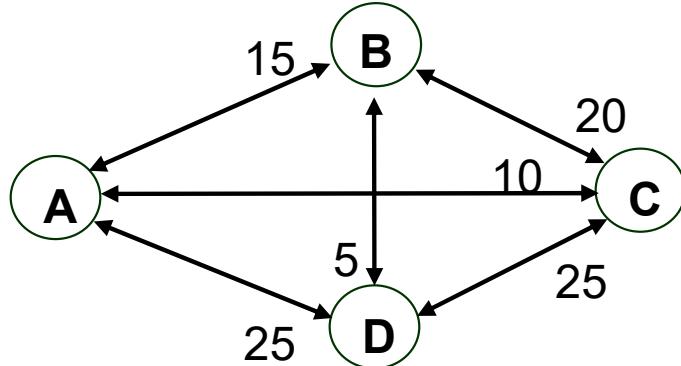
Matriz de Feromonas (actualizada)

$$p_{i,j} = \frac{(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}{\sum(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}$$

$$\nwarrow \tau_{ij} = (1-\rho) \tau'_{ij} + \Delta\tau_{ij}$$

	B	C	D
A	0,1	0,1	0,1
B		0,1	0,1
C			0,1

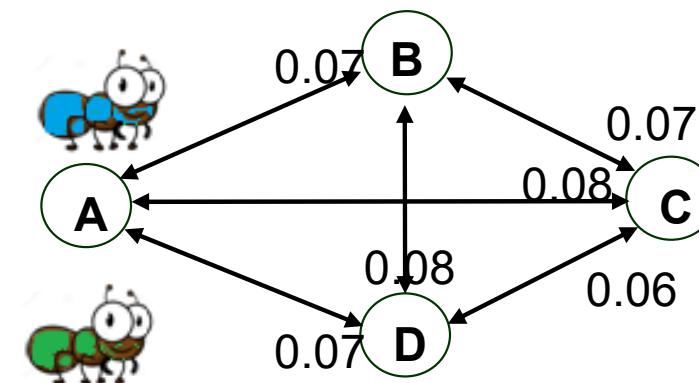
Matriz de Feromonas (inicial)



Hormigas=2,  $\rho=0,5$ ,  $\alpha=0,4$ ,  $\beta=0,6$ ,  $\tau_{INICIAL}=0,1$

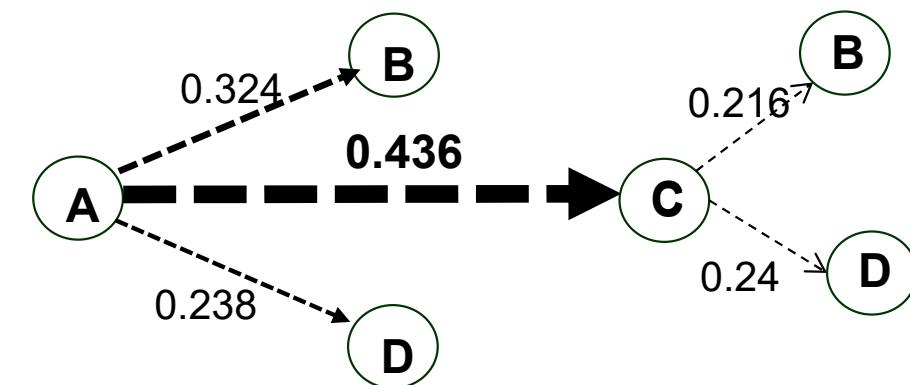
Segunda iteración:

*Hay una retro-alimentación entre las hormigas*



	B	C	D
A	0,324	<b>0,436</b>	0,238
B		0,216	0,525
C			0,24

Matriz de Probabilidades (actualizada)



**A-C-D-B-A es un camino óptimo**

## AlgoritmoHormigas ()

- Inicialización Matriz de Visibilidad, Matriz de Feromonas
- Repetir hasta fin

Construcción Soluciones:

Para cada paso i (i=1, N) {

    Para cada Hormiga-h (h=1, Hormigas)

        { Hormiga-h selecciona siguiente arco (ciudad) en su senda:

            Una hormiga irá de  $i \rightarrow j$ , con probabilidad  $p_{i,j} = \frac{(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}{\sum(\tau_{i,j}^\alpha)(\eta_{i,j}^\beta)}$

            Donde,

$\tau_{ij}$  es la feromona de arco  $i \rightarrow j$ , con una influencia  $\alpha$

$\eta_{ij}$  es la visibilidad del arco, con influencia  $\beta$  }

}

Actualización-Feromonas:

La feromona de cada arco  $i \rightarrow j$  ( $\tau_{ij}$ ) se actualiza de la forma:  $\tau_{ij} = (1-\rho) \tau'_{ij} + \Delta\tau_{ij}$

Donde,  $\rho$  Es la tasa de evaporación, que se aplica a la feromona previa  $\tau'_{ij}$

$\Delta\tau_{ij}$  Feromona depositada por todas las hormigas en los arcos que recorren:

$$\Delta\tau_{ij} = \sum_{h=1,H} \Delta\tau^h_{ij}$$

siendo  $\Delta\tau^h_{ij}$  la feromona dejada por hormiga h en  $i \rightarrow j$ :

=  $1/L_h$  si la hormiga h atraviesa el arco, y ha realizado una senda de coste  $L_h$

= 0 en otro caso

Actualización Matriz de Probabilidades

*Los problemas que se pueden resolver con los algoritmos ACO son los que se pueden representar como un grafo ponderado  $G = (N,A)$ , donde A es el conjunto de aristas que conectan los nodos N*

## Dependencia de los parámetros

### ↑ Feromona inicial $\Rightarrow$ ↑ Exploración

Si se inicializan las **feromonas a valores muy bajos** la búsqueda estará muy influenciada por los primeros recorridos: **menos explotación**.

Si se inicializan las **feromonas a valores muy altos** deberán pasar muchas iteraciones hasta que la evaporación reduzca los valores de feromona lo suficiente para que las feromonas introducidas por las hormigas empiecen a influir en la búsqueda: **más exploración**

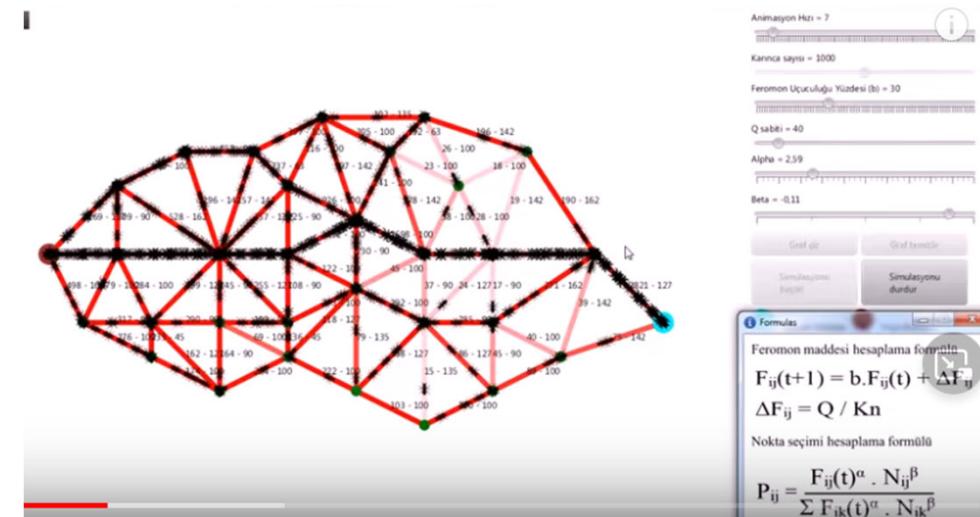
### ↑ Feromona depositable por hormigas $\Rightarrow$ ↑ Explotación

↑ H: nº de Hormigas

↑  $\rho$  (evaporación): menos realimentación

$\alpha$ : Influencia visibilidad ( $\eta$  : heurística del problema)

$\beta$  Influencia Feromona ( $\tau$ : retroalimentación)



Numan Karaaslan (Requiere JavaX)

# Variaciones Algoritmo de las Hormigas

## Sistema Básico

En cada iteración, las hormigas depositan feromona de acuerdo a la calidad de la solución que encuentran ( $1/L_k$ ).

## Sistema Elitista

En cada iteración, la **solución global (almacenada) mejor** también deposita feromonas, junto con el resto de las hormigas.

## Sistema basado en el rango

Las soluciones son ordenadas de acuerdo a su calidad (longitud).

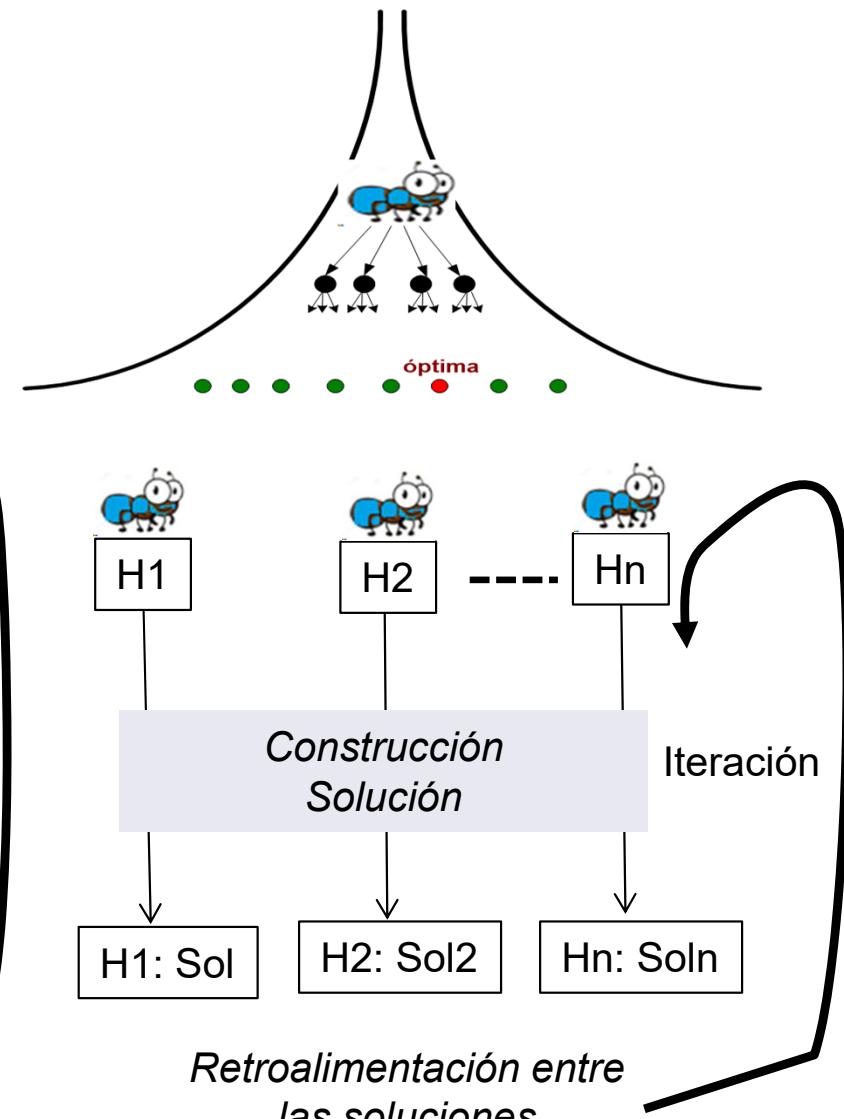
La cantidad de feromona depositada por cada hormiga **depende del rango de su solución, en vez de la longitud de su senda.**

## Sistema de Colonias:

Todas las hormigas dejan la misma feromona en los arcos que recorren, independientemente de la bondad de la solución.

Pero las hormigas no están sincronizadas y las que acaban antes (por haber encontrado mejor camino) vuelven a comenzar una nueva búsqueda. Así, los mejores caminos tendrán más probabilidad de ser repetidos. **Es la típica forma en la naturaleza.**

# Algoritmo Hormigas *versus* Metaheurísticas Mejora y Constructivas



## Búsqueda Local en la vecindad

- Enfriamiento Simulado, Tabú, Búsqueda en Haz.
- Función de Evaluación  $f(n)$
- No hay realimentación entre soluciones, salvo la propia solución actual

## Constructivo (GRASP)

- Construcción de Solución (Múltiples soluciones independientes)
- Mejora Local posterior
- Función heurística  $f(n)$

## Algoritmo de las Hormigas

- Construcción de las Soluciones
- Paso a paso, por múltiples hormigas
- Retro-alimentación entre las soluciones obtenidas

## Aplicaciones Típicas:

- Enrutamiento de vehículos. Determinación de rutas.
- Diseño de circuitos lógicos.
- Asignación de recursos. Asignación de Horarios.
- Problemas de Recubrimiento.
- Control Inteligente.
- Etc.

## Main Ant Colony Optimization

Routing	Traveling salesman	Vehicle routing (VRP)	VRP with time windows	VRPMTWMV	VRP with loading constraints	Team orienteering	Sequential ordering	TSP with time window	Single machine	Constraint satisfaction	Course timetabling	Ambulance location	MAX-SAT	Assembly line balancing	Simple assembly line balancing	Supply chain management
Scheduling																
	Flow shop	Machine learning														
	Industrial scheduling															
	Project scheduling															
	Group shop	Bioinformatics														
	Job shop															
	Open shop															
	Car sequencing															
Subset		Multiple knapsack														
		Maximum independent set														
		Redundancy allocation														
		Weight constraint graph tree partitioning														
			Bin packing													
			Set covering													
			Set packing													
			<i>l</i> -cardinality trees													
			Capacitated minimum spanning tree													
			Maximum clique													
			Multilevel lot-sizing													
				Edge-disjoint paths												
				Feature selection												
				Multicasting ad-hoc networks												
Assignment and layout		Quadratic assignment														
			Graph coloring													
			Generalized assignment													
			Frequency assignment													

## Bee colony optimization part i: The algorithm

overview. Tatjana Davidovic, Dusan Teodorovic, Milica Selmic (2015) DOI:10.2298/YJOR130515034A

## Monografía: Ant Colony Optimization - Techniques and Applications

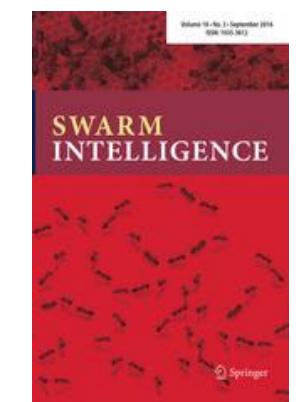
Edited by Helio J.C. Barbosa, InTech (Ed), 2013 .

## Monografía: Ant Colony Optimization – Methods and Applications

Edited by A. Ostfeld InTech (Ed), 2014

## A concise overview of applications of ant colony optimization.

Stützle, López, Dorigo. U. Libre de Bruselas. Tech. Rpt



Instance	GA		ACO		SA	
	Best Distance	Time (s)	Best Distance	Time (s)	Best Distance	Time (s)
bier127	419224	~3	124651.524	~27	265289	~0.3
berlin52	11551	~1	7721.432	~4	10586	~0.2
ali535	9466	~5	1510.637	~62	6471	~0.3

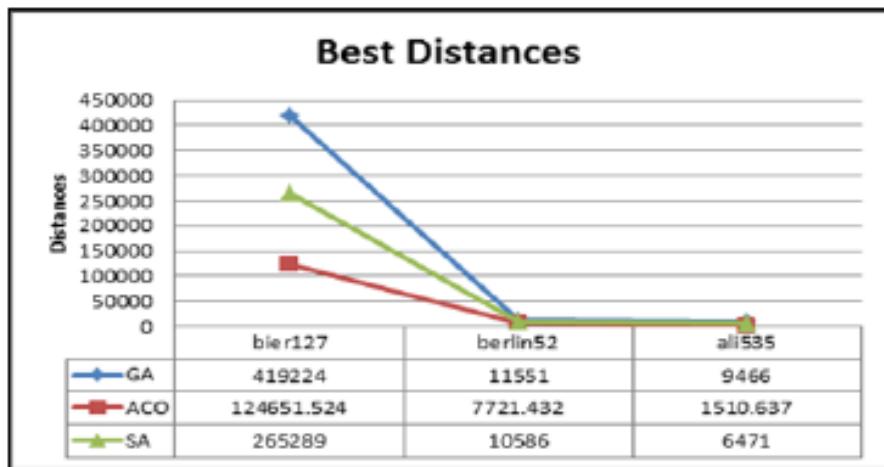


Figure 12. Evolution of GA, ACO and SA results over best distance for bier127, berlín52, and ali535

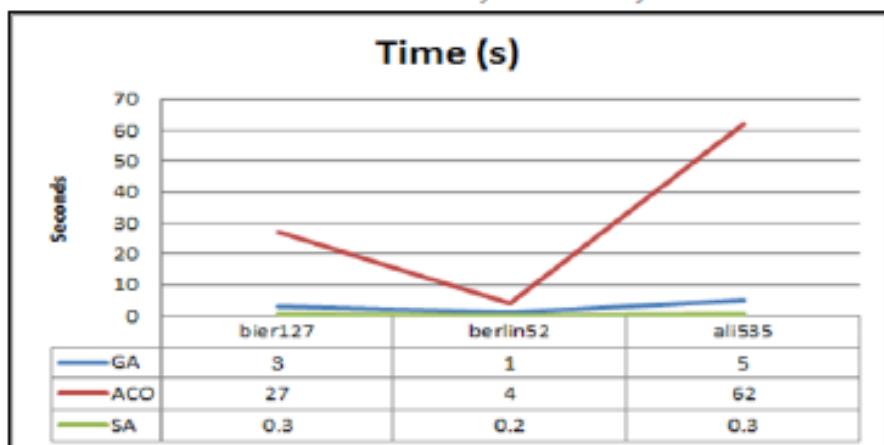


Figure 13. Evolution of GA, ACO and SA results over time(s) for bier127, berlín52 and ali535

El tipo de problemas que se pueden resolver por medio de ACO pertenece al **grupo de problemas de camino mínimo** que pueden representarse en forma de **grafo ponderado  $G = (N, A)$** .

- N: **nodos del grafo**,
- Las **soluciones se corresponden con caminos en el grafo** (secuencias de nodos o aristas),
- Existen **costes asociados a cada arista**,
- Las conexiones tendrán asociados rastros de feromona  $\tau$  y **valores heurísticos  $\eta$** , que representan la información heurística disponible en el problema.

In: Performance Comparison of Simulated Annealing, GA and ACO Applied to TSP. Int J of Intelligent Computing Research 6, 4, Dec2015

# Using Ant Colony Optimization to Solve Train Timetabling Problem of Mass Rapid Transit

Artículos en Poliformat

Jau-Ming Su<sup>1</sup> Jen-Yu Huang<sup>2</sup>

<sup>1</sup>Department of Transportation Technology and Logistic Management, Chung-Hua University

<sup>2</sup>Division of Traffic Engineering Institute of Civil Engineer, National Taiwan University



International Journal of Production Economics

Volume 149, March 2014, Pages 131-144

## Ant Colony System-based Applications to Electrical Distribution System Optimization

Gianfranco Chicco  
Politecnico di Torino  
Italy

An ant colony based timetabling tool

Thatchai Theppakorn <sup>a</sup>, Pupong Pongcharoen <sup>a</sup>✉, Chris Hicks <sup>b</sup>

✉ Show more

<https://doi.org/10.1016/j.ijpe.2013.04.026>

Get rights

Ant@optima

## International Journal of Innovative Research in Computer and Communication Engineering

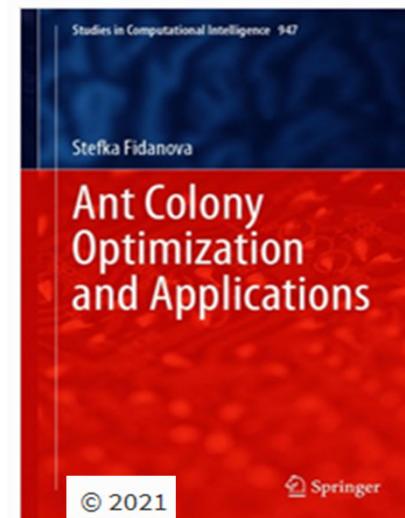
(An ISO 3297: 2007 Certified Organization)

Website: [www.ijircce.com](http://www.ijircce.com)

Vol. 5, Issue 3, March 2017

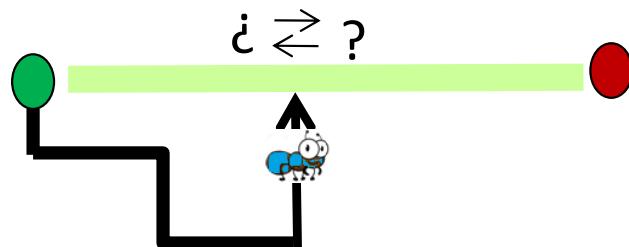
## Timetable Generation Using Ant Colony Optimization Algorithm

Priyanka Gore<sup>1</sup>, Poonam Sonawane<sup>1</sup>, Suneha Potdar<sup>2</sup>

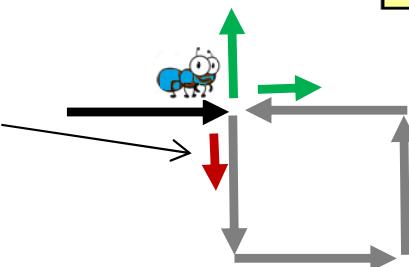


# Aplicación del Algoritmo de las Hormigas en Laberintos

Información de feromonas: valor y dirección

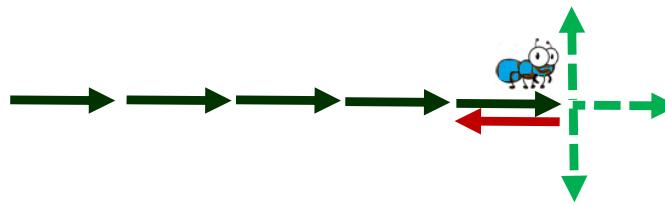


Ciclos: Olvidar ciclo, penalizar selección

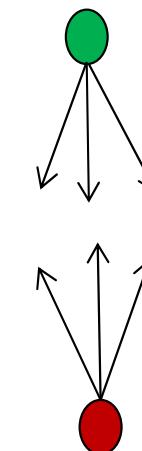
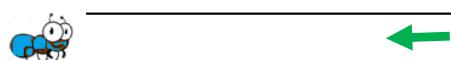


Búsqueda Bidireccional

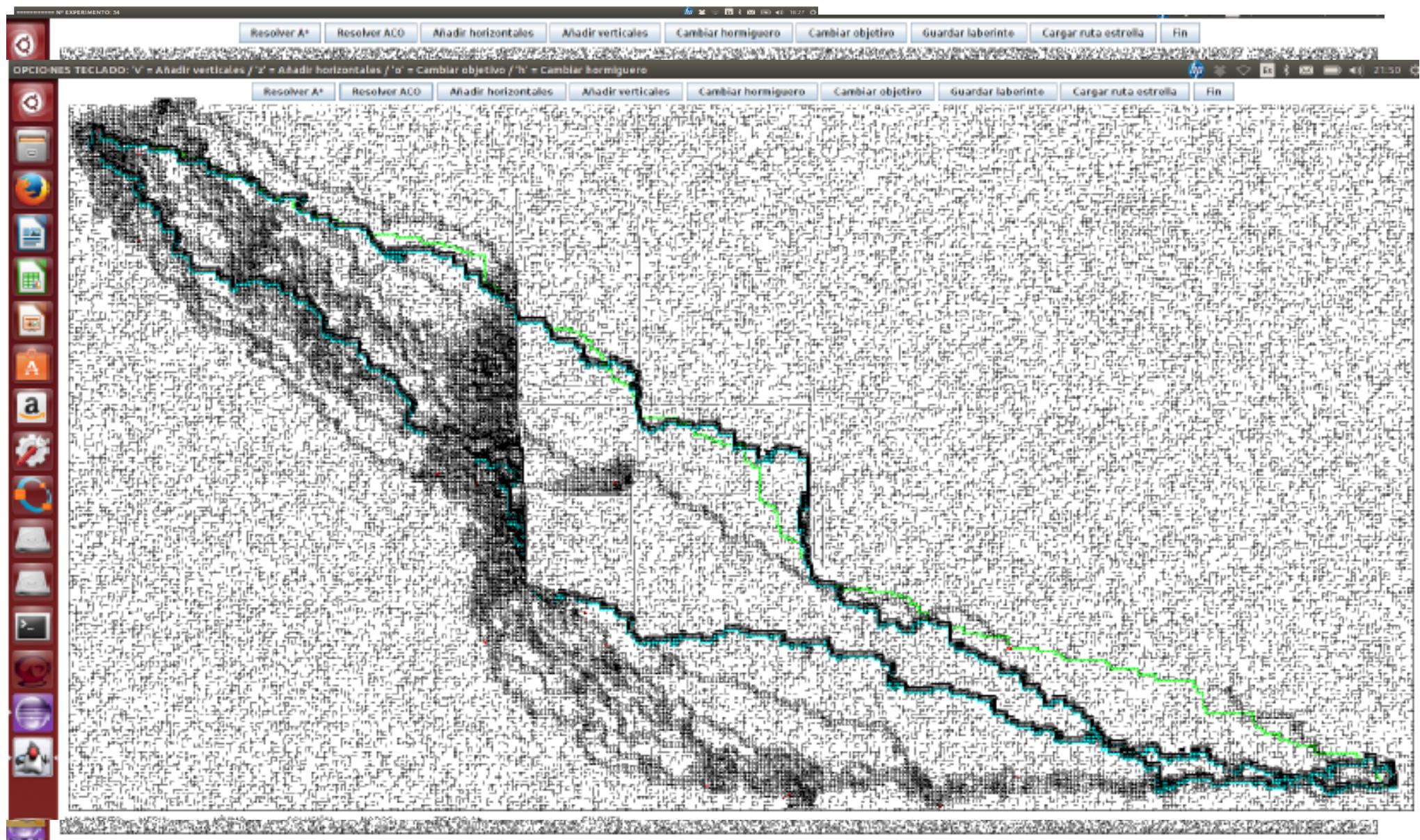
Penalizar vuelta atrás



Bloqueo del camino: Permitir vuelta atrás

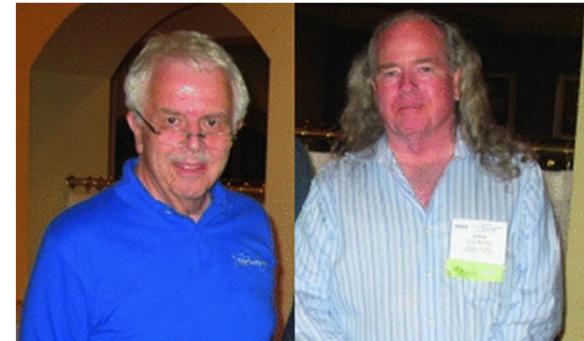


## Ejemplos de Laberintos



# **Enjambre de Partículas**

*(Particle Swarm Optimization - PSO)*



Russel Eberhart and James Kennedy

*Kennedy & Eberhart, 1995,*



**Particle swarm optimization.** James Kennedy and Russell Eberhart.

Proceedings of IEEE International Conference on Neural Networks, pg 1942–1948, 1995. IEEE Press.

<http://www.swarmintelligence.org/>

<http://www.particleswarm.info/>

# Bibliografía (Artículos en Poliformat)

Google Académico



Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: [www.elsevier.com/locate/eswa](http://www.elsevier.com/locate/eswa)

Short communication

Salp Swarm Optimization: A critical review

## Chapter 11 Different Applications of PSO

Applied Intelligence (2019) 49:335–351  
<https://doi.org/10.1007/s10489-018-1258-3>

Enhanced particle swarm optimization with multi-swarm and multi-velocity for optimizing high-dimensional problems

- Swarm Optimization A critical review (2022)*  
*State of the art review of intelligent scheduling (2020)*  
*Bee-inspired metaheuristics (2021)*  
*Comparison of 5 novel nature inspired metaheuristic (2020)*  
*Different Applications of PSO (2021)*  
*From ants to whales, metaheuristics for all tastes (2020)*  
*LIBRO Ant Colony Optimization and Applications (2021)*  
*Metaheuristic Review 2019*  
*Multi-objective particle swarm optimization (2020)*  
*Particle Swarm Optimisation A Historical Review (2020)*  
*Particle swarm optimization (2021)*  
*Particle Swarm Optimization A Comprehensive Survey*  
*Particle Swarm Optimization Algorithms (2017)*  
*Application of particle swarm in power systems (2017)*  
*PSO para muchos objetivos*  
*PSO Viajante Multiobjetivo*  
*SO for optimizing high-dimensional problems (2019)*

## Particle Swarm Optimization: A Comprehensive Survey

Metaheurística bioinspirada, basada en la interacción social de las bandadas de pájaros, bancos de peces, etc. : Enjambres de Partículas.

- En un grupo (enjambre) se establecen diferentes jerarquías, de acuerdo a las características de cada individuo.
- Particularmente, existe un líder, reconocido y seguido por los demás individuos del grupo.
  - El líder tiene habilidades que le permiten ser guía para buscar alimento.
  - Los individuos de menor jerarquía confían en la capacidad del líder para dirigirlos.

En el proceso de búsqueda de alimento (búsqueda solución):

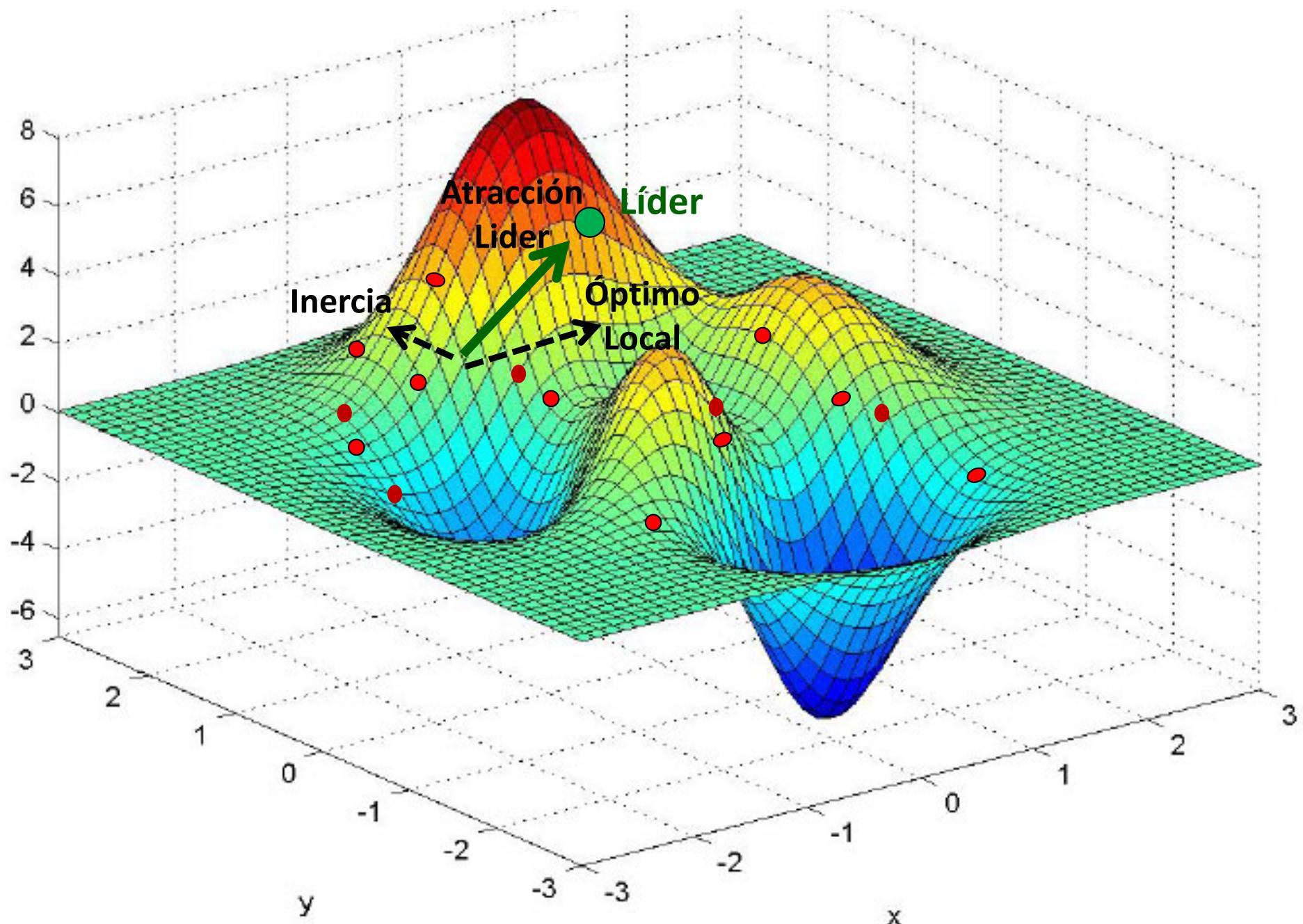
- El líder guía al grupo por donde sospecha encontrarán alimento.
- Todos los individuos siguen al líder, pero tratando también a su vez de localizar alimento y evitar colisiones.
- Eventualmente, comunican al grupo nuevas ubicaciones de alimento (existe una comunicación entre el conjunto de partículas del enjambre).



El grupo no es estático: Puede surgir un individuo con mejores capacidades que el actual, sustituyéndolo en el mando del grupo: nuevo líder.

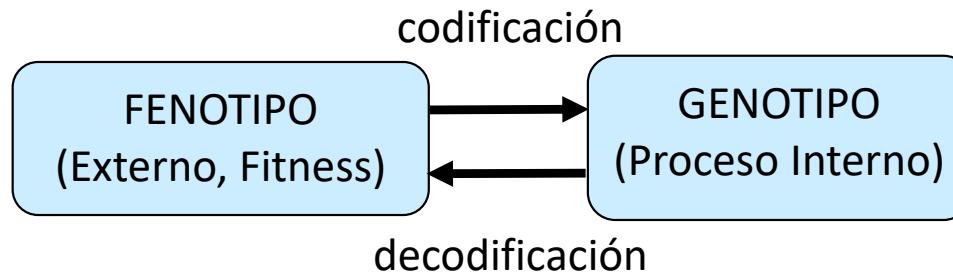
- El proceso de selección se basa en comparar del valor de aptitud (valor de la función objetivo) de *cada miembro* del grupo con el valor de aptitud del actual líder: el miembro del grupo con mejor evaluación será el nuevo líder si supera al líder actual.

Idea Básica: Cada **partícula** sobrevuela el espacio de soluciones **n-dimensional** en busca de soluciones óptimas.



## Modelización del problema y Proceso

Cada partícula (individuo) del enjambre (o nube de partículas) representa una solución y realiza un proceso de búsqueda de mejores soluciones.



- La búsqueda de cada partícula<sub>i</sub> en el espacio de soluciones se realiza en base al “conocimiento propio”, a la influencia del líder y al movimiento de la partícula:
  - Solución propia: Pbest<sub>i</sub>*
  - Solución aportada por el líder: Gbest<sub>i</sub>*
  - Movimiento de la partícula: V<sub>i</sub>*
- Se plantean unas reglas matemáticas para el movimiento de cada partícula.
- El objetivo final es que el enjambre de partículas converja hacia las mejores soluciones

## MODELIZACIÓN

$$\text{Partícula}_i \cong \text{Solución}_i \\ \Rightarrow$$

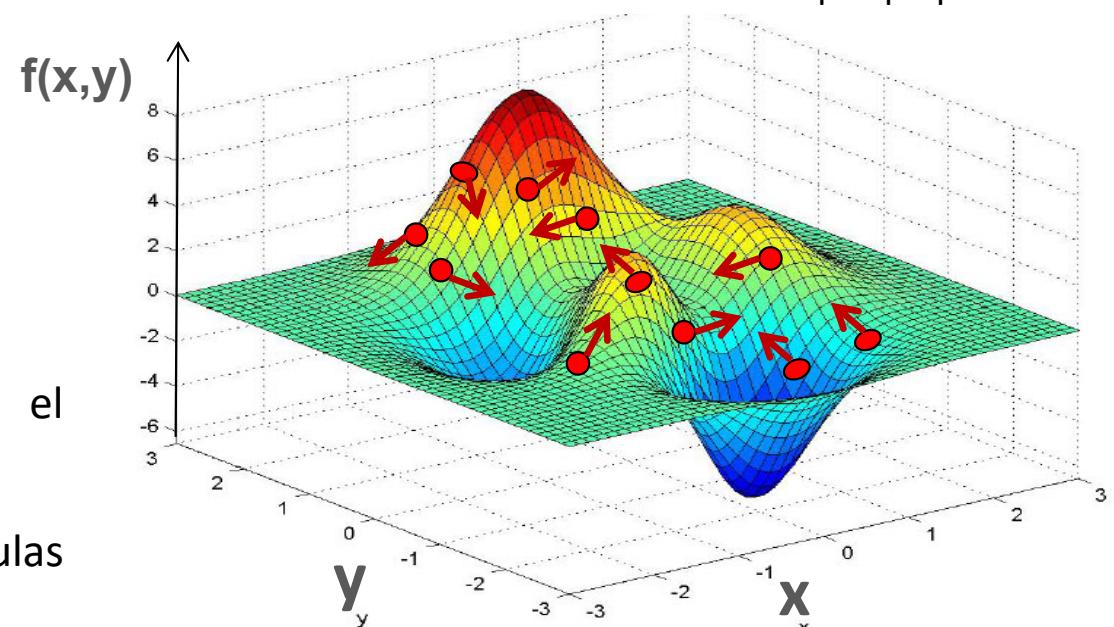
Conjunto de decisiones (o valores) sobre cada variable del problema:

$$\text{Partícula}_i = [x_{i1}, x_{i2}, x_{i3}, \dots, x_{in}]$$

Típicamente vector de Reales o Enteros

↓  
**Espacio de Búsqueda  $\cong$  Espacio de Decisión n-dimensional**

**Ejemplo 2-dimensional:** Partícula<sub>i</sub>: (x<sub>i</sub>, y<sub>i</sub>)



# Esquema Básico Algoritmo PSO

Crear **enjambre inicial** de partículas (soluciones en el espacio de búsqueda) y **velocidades iniciales** (típicamente de manera aleatoria).

*Soluciones : { (Posición<sub>i</sub>, Velocidad<sub>i</sub>) }*

**Mientras** no\_fin (*nº evaluaciones, optimalidad, etc.*)

**1.** Evaluar cada partícula<sub>i</sub>:      *Evaluación de Soluciones*

- Pbest<sub>i</sub>: Mejor solución encontrada por la partícula<sub>i</sub>
- V<sub>i</sub>: Velocidad de la partícula<sub>i</sub> (incluye dirección)
- Gbest: Mejor solución encontrada por el enjambre (líder).

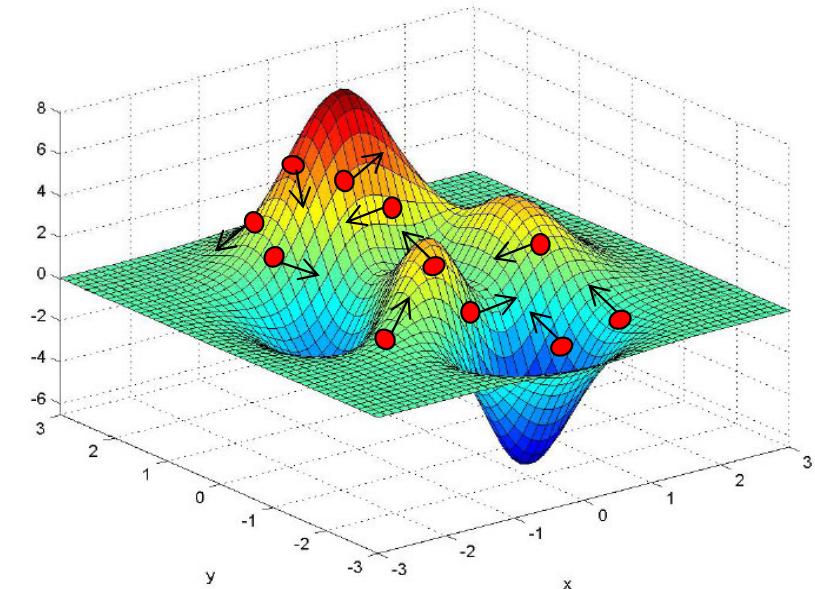
*Búsqueda nuevas soluciones*

**2.** Obtener función de vuelo de las partículas (V'<sub>i</sub>):

Obtener nueva velocidad y dirección de cada partícula ( $V'_i$ ) en base a posición del líder (**Gbest**) y al propio proceso de búsqueda local ( $V_i$ ,  $Pbest_i$ ).

**3.** Actualizar la posición de cada partícula (nueva solución).

Aplicar función de vuelo a la posición actual y obtener nueva posición ( $X'_i$ )



**Para cada partícula<sub>i</sub>:**

Posición<sub>i</sub>: Codificación de una solución S<sub>i</sub>  
Genotipo (n dimensional)  
[ $x_{i1}, x_{i2}, x_{i3}, \dots, x_{in}$ ]

Velocidad<sub>i</sub>: Velocidad de la particula,  
hacia una nueva solución  
 $S_i \rightarrow S'_i$

Pbest<sub>i</sub>: Mejor solución de la partícula<sub>i</sub>  
[ $p_{i1}, p_{i2}, \dots, p_{in}$ ]

**Gbest:** Mejor solución del enjambre.

# Esquema Básico Algoritmo PSO

Crear enjambre inicial de partículas (soluciones en el espacio de búsqueda) de manera aleatoria y velocidades iniciales.

$$\text{Soluciones} : \{ (\text{Posición}_i, \text{Velocidad}_i) \}$$

Mientras no\_fin ( $n^o$  evaluaciones, optimalidad, etc.)

1. Evaluar cada partícula<sub>i</sub>:      *Evaluación de Soluciones*

- Pbest<sub>i</sub>: Mejor solución encontrada por la partícula<sub>i</sub>
- V<sub>i</sub>: Velocidad de la partícula<sub>i</sub> (incluye dirección)
- Gbest: Mejor solución encontrada por el enjambre (líder).

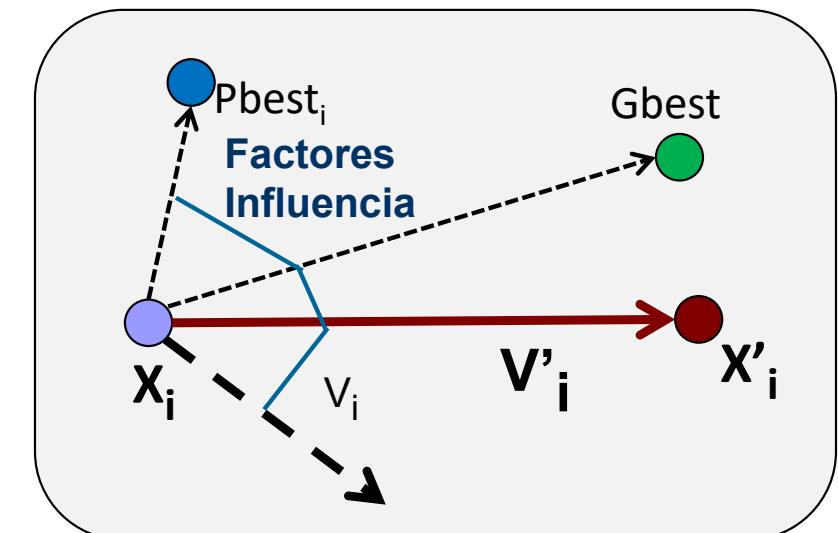
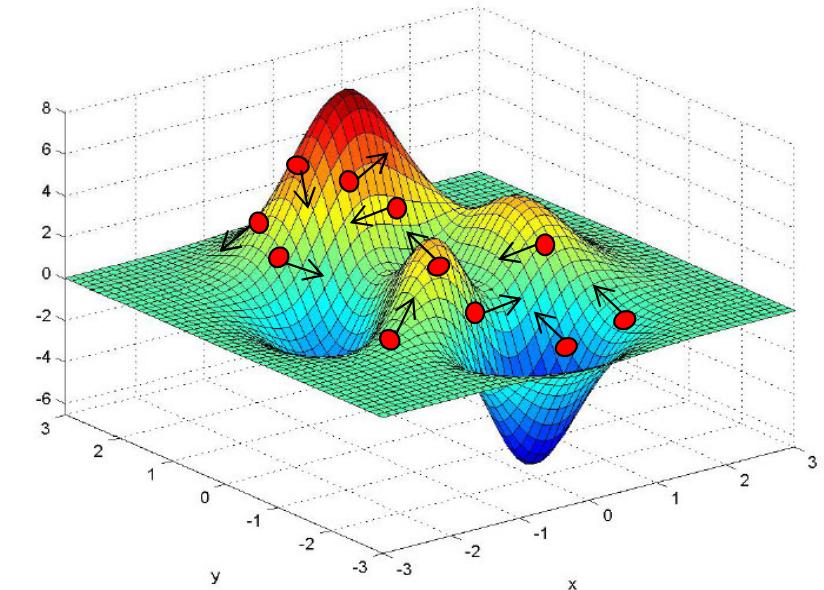
*Búsqueda nuevas soluciones*

2. Obtener función de vuelo de las partículas ( $V'_i$ ):

Obtener nueva velocidad y dirección de cada partícula ( $V'_i$ ) en base a posición del líder (**Gbest**) y al propio proceso de búsqueda local ( $V_i$ , Pbest<sub>i</sub>).

3. Actualizar la posición de cada partícula (nueva solución).

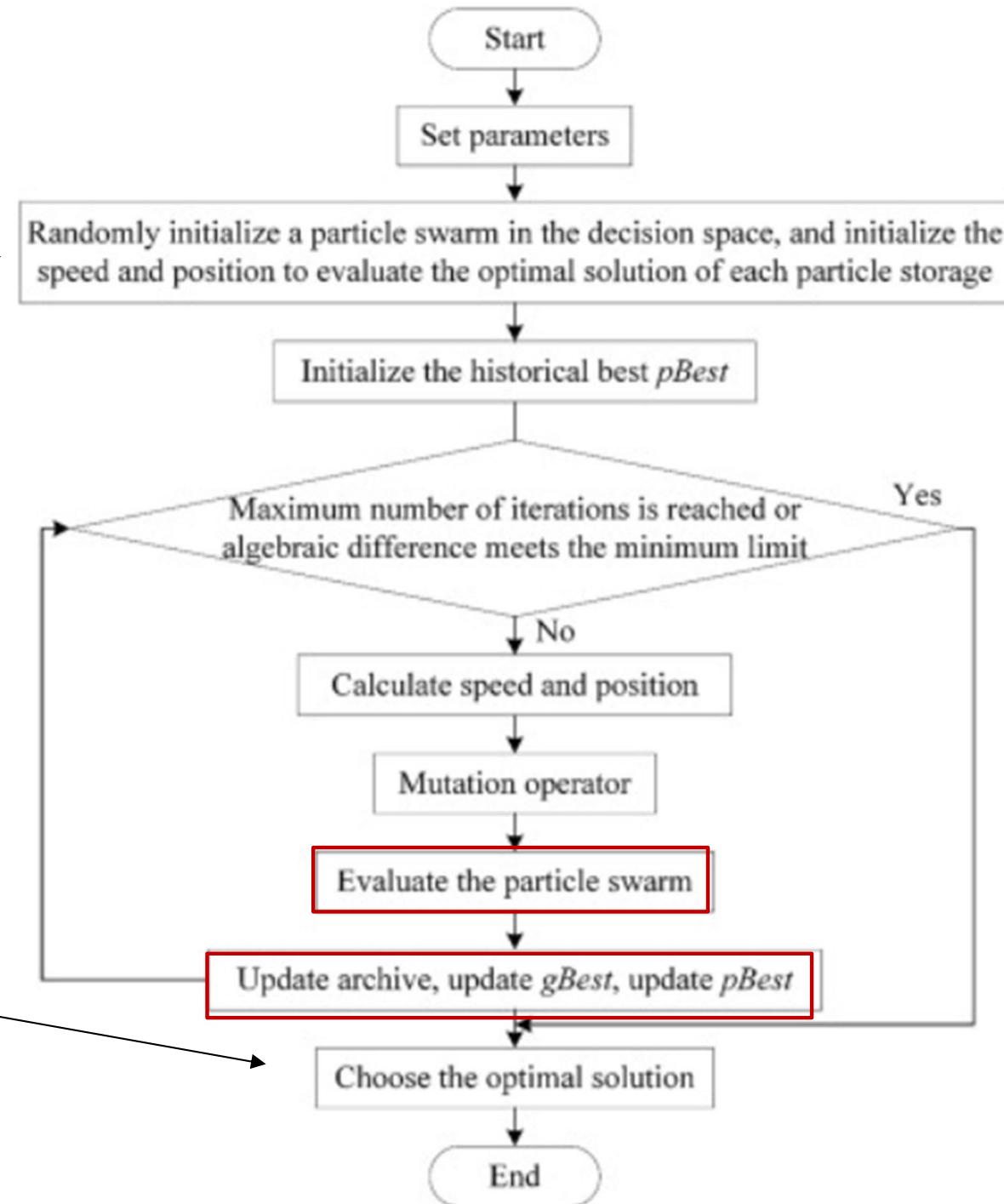
Aplicar función de vuelo a la posición actual y obtener nueva posición ( $X'_i = X_i \otimes V'_i$ )



Cada partícula explora el espacio de soluciones a partir de su mejor posición y de la mejor posición global

# Esquema Básico Algoritmo PSO

Inicializacion  
Partticulas



Devuelve Gbest

# Ecuación del Movimiento: $V_i$

Cada Partícula-i se caracteriza por (*espacio n-dimesional*):

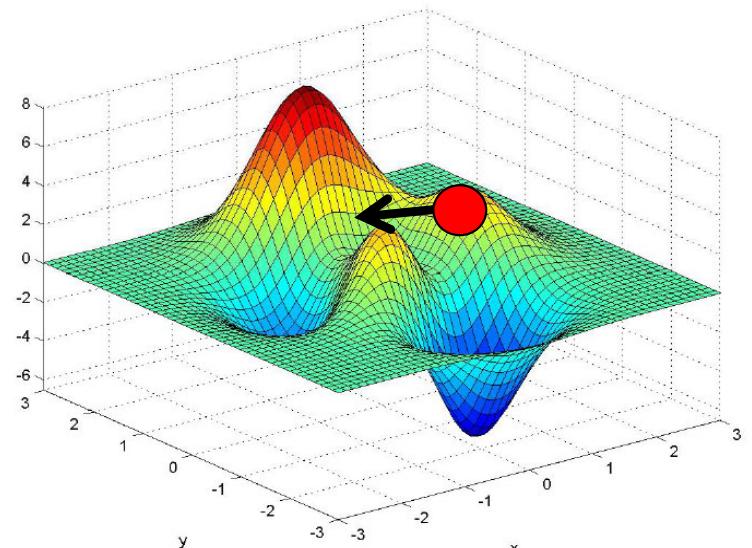
Posición  $X_i = [x_{i1}, x_{i2}, \dots, x_{in}]$

Velocidad  $V_i = [v_{i1}, v_{i2}, \dots, v_{in}]$

Mejor posición personal (en su evolución):  $P_i = [p_{i1}, p_{i2}, \dots, p_{in}]$

Además, mejor posición global alcanzada por todas las partículas:

$G = [g_1, g_2, \dots, g_n]$



En cada iteración del proceso de búsqueda, cada Partícula-i actualiza su velocidad  $V'_i$ ,

donde cada componente  $[v'_{i1}, v'_{i2}, \dots, v'_{in}]$ , se obtiene:

$$\text{Típicamente: } w + c1 + c2 = 1$$

$$v'_{ik} = w * v_{ik}$$

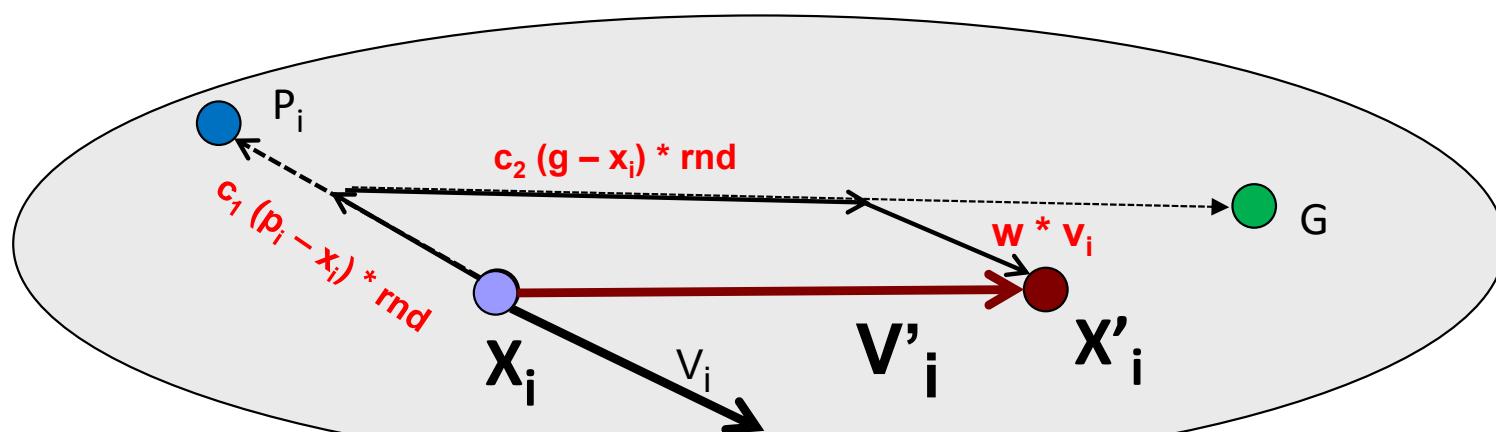
$w \in [0, 1]$ : *Inercia*, o impacto velocidad previa  $V_i$ .

$$+ c_1 (p_{ik} - x_{ik}) * \text{random}(0, 1)$$

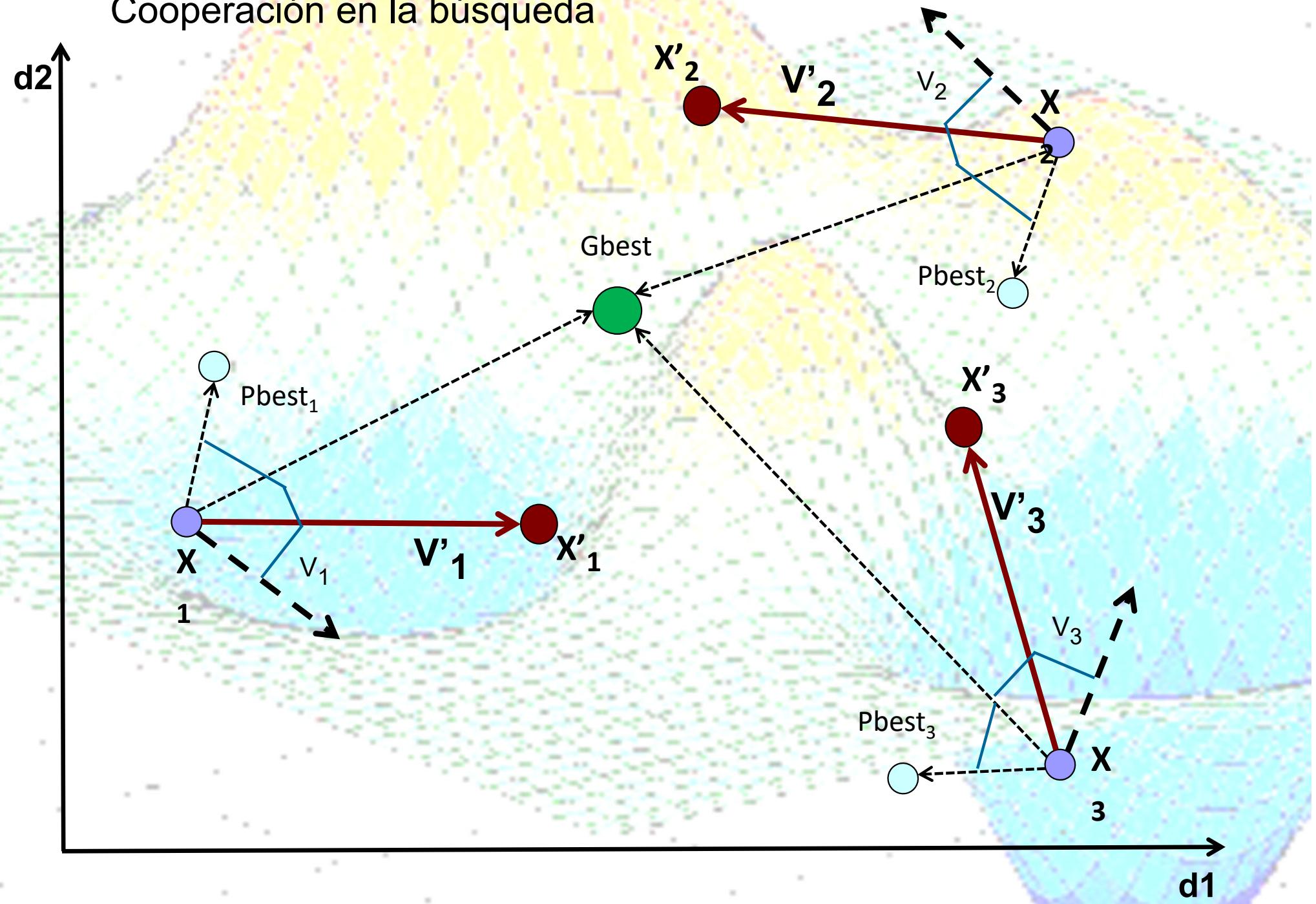
$c_1$ : Parámetro *cognitivo* (experiencia individual)

$$+ c_2 (g_k - x_{ik}) * \text{random}(0, 1)$$

$c_2$ : Parámetro *social* (influencia enjambre, cooperación)



## Cooperación en la búsqueda



# Algoritmo PSO Básico

---

## Algorithm 1: classical PSO

---

**input** : Objective function  $f : S \rightarrow \mathbb{R}$  to be minimized

**output**:  $G \in \mathbb{R}^D$

**Inicialización**

```
// Initialization
1 for  $n = 1 \rightarrow N$  do      Para cada partícula
2   Initialize position  $X^n \in \mathbb{R}^D$  randomly;
3   Initialize velocity  $V^n \in \mathbb{R}^D$ ;
4   Initialize local attractor  $L^n := X^n$ ;
5 Initialize  $G := \text{argmin}_{\{L^1, \dots, L^n\}} f$ ;    Mejor Posición Global inicial
```

*Posición y Velocidad inicial*

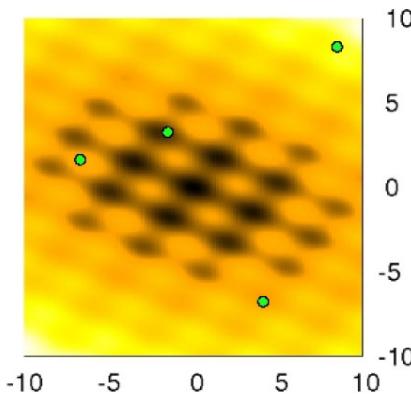
*Mejor Posición Local inicial*

// Movement

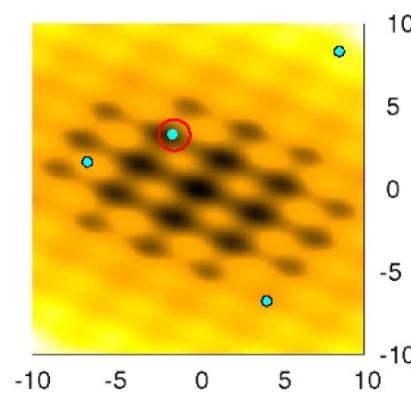
**Optimización**

```
6 repeat
7   for  $n = 1 \rightarrow N$  do      Para cada partícula
8     for  $d = 1 \rightarrow D$  do          Para cada dimensión           Actualiza Velocidad
9        $V^{n,d} := \chi \cdot V^{n,d} + c_1 \cdot \text{rand}() \cdot (L^{n,d} - X^{n,d}) + c_2 \cdot \text{rand}() \cdot (G^d - X^{n,d})$ ;
10       $X^{n,d} := X^{n,d} + V^{n,d}$ ;   Actualiza Posición  $\otimes$ 
11      if  $f(X^n) \leq f(L^n)$  then  $L^n := X^n$ ;    Actualiza Óptimo Local
12      if  $f(X^n) \leq f(G)$  then  $G := X^n$ ;    Actualiza Óptimo Global
13 until Termination criterion holds;
14 return  $G$ ;
```

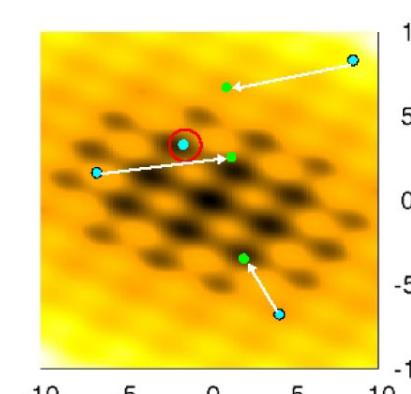
# Ejemplo (M. Montes, U. L. Bruxelles)



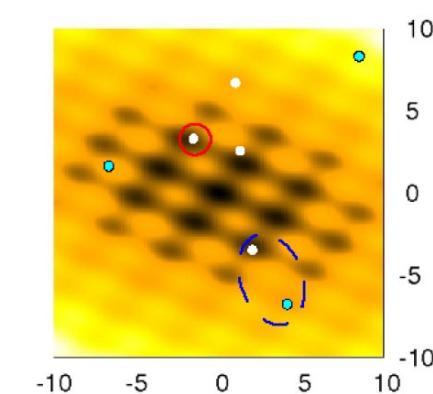
Creación de partículas y evaluación según  $f(x)$



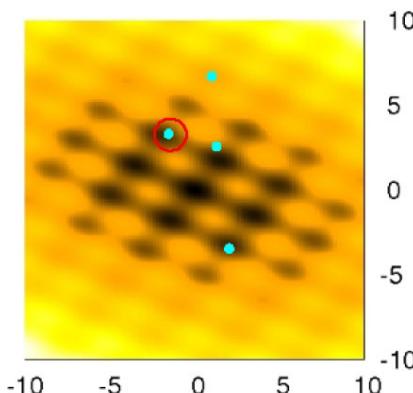
Determinación mejor partícula.



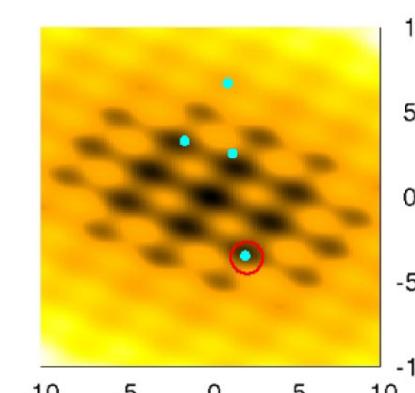
Actualización posición partículas según reglas movimiento.



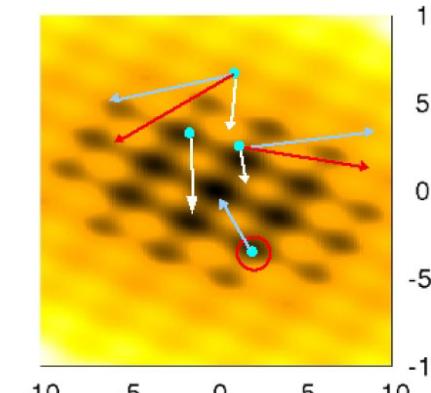
Evaluación nuevas posiciones.  
Si nueva posición es mejor que la previa, actualizar posición.



Nuevas posiciones.



Determinación nueva mejor partícula.



Actualizar dirección y velocidades según reglas movimiento.

*Repetir (3) hasta condición parada se cumpla.*

# Parámetros PSO

Como en toda metaheurística, es importante el ajuste de parámetros (diferentes variantes propuestas en PSO).

## nº partículas (tamaño del enjambre)

Dependiente de la irregularidad de la hiper-superficie del fitness: *Cuanto más irregular, serán necesarias más partículas.*

$$v'_{ik} = w * v_{ik} + c_1 (p_{ik} - x_{ik}) * RND(0, 1) + c_2 (g_k - x_{ik}) * RND(0, 1)$$

## w (inercia).

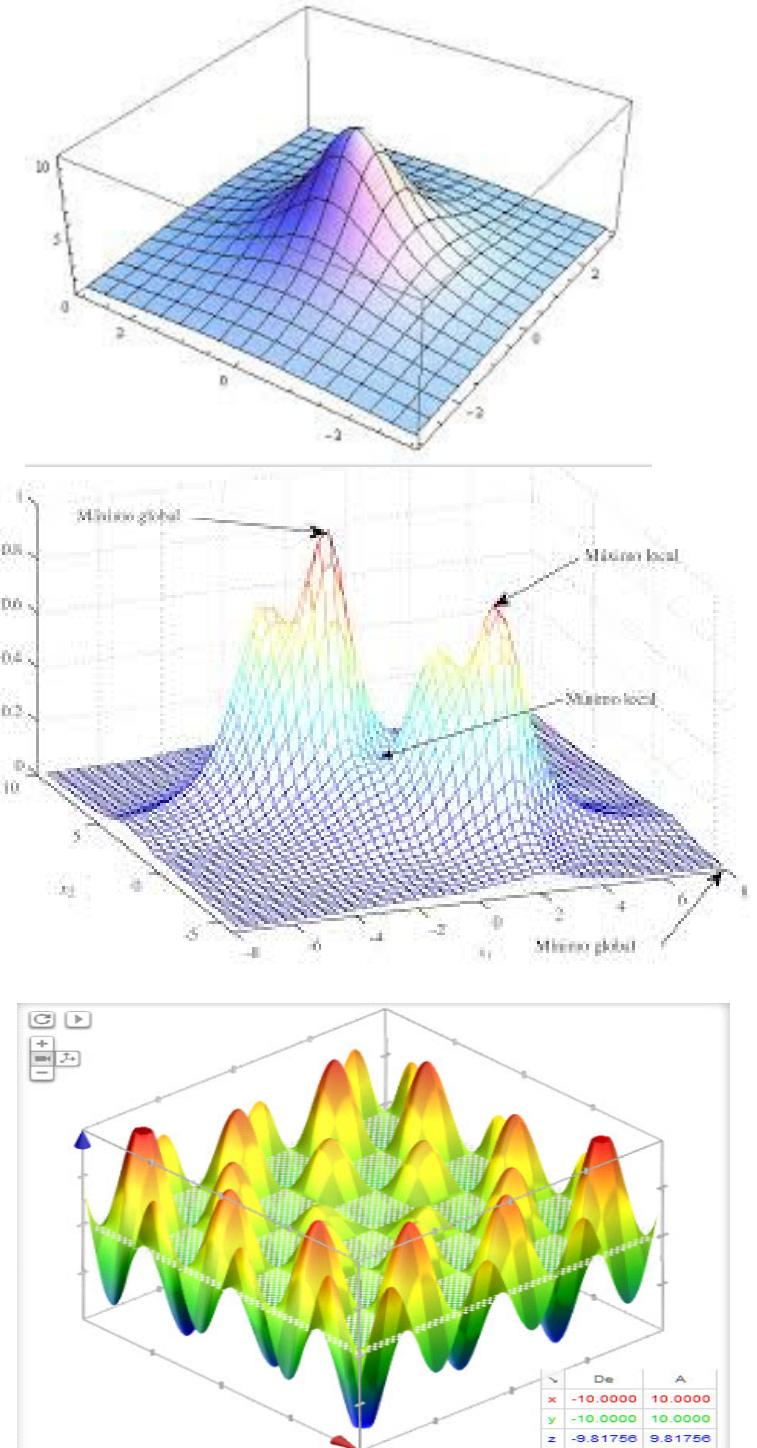
- Factor de *exploración* en búsqueda local
- Usualmente tiende a decrementarse en la búsqueda

## c1: factor cognitivo personal (memoria)

- Controla el movimiento hacia su propio óptimo (*explotación local*)

## c2: factor cognitivo social (cooperación)

- Factor de *explotación global*. Controla el movimiento hacia óptimo global.
- Un valor alto puede producir caer en óptimos locales.



## Función de Vuelo: $X_i \otimes V'_i \rightarrow X_{i+1}$

La función de vuelo obtiene la nueva posición  $X_{i+1}$  de la partícula, a partir de:

la posición actual:  $X_i : [x_{i1}, x_{i2}, x_{i3}, \dots, x_{in}]$

y la nueva velocidad  $V'_i : [v_{i1}, v_{i2}, v_{i3}, \dots, v_{in}]$

$$V'_{ik} = c_2 * \text{RND}() * (g_k - x_{ik}) + c1 * \text{RND}() * (p_{ik} - x_{ik}) + w * v_{ik}$$

$$X_{i+1} = X_i \otimes V'_i$$

$V'_i$  combina la mejor posición del enjambre ( $G$ , como información global), la mejor posición local ( $P_i$ , como información local) , y la velocidad anterior ( $V_i$ ).

*La operación de vuelo es dependiente del diseño del sistema!  
(dimensiones / #variables)*

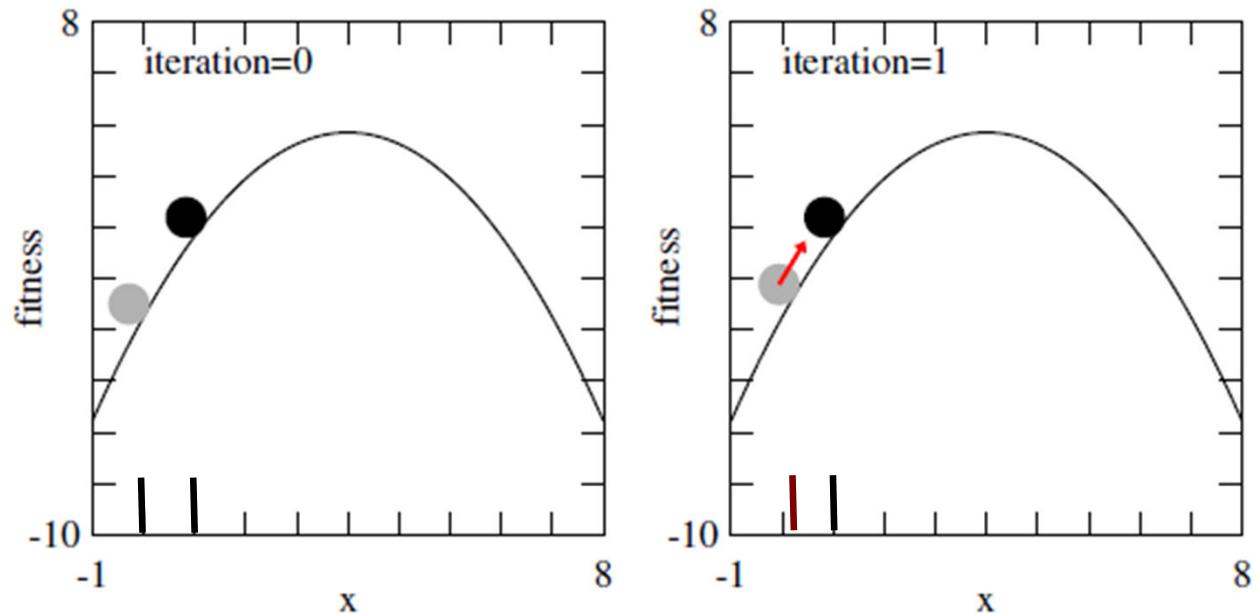
# Ejemplo 1. Una dimensión

Variable:  $x \in [-1, 8]$

Objetivo: maximizar  $f(x)$

Nº partículas: 2

Parámetros:  $W= 0.9$ ,  $C1, C2 = 1$



**Función de Vuelo  $v'$ , Actualización de la posición  $x'$ :**

$$v'_{ik} = W * v_{ik} + C_1 (p_{ik} - x_{ik}) + C_2 (g_k - x_{ik})$$

$$v_i(t+1) = 0.9v_i(t) + [b_i(t) - x_i(t)] + [B(t) - x_i(t)]$$

$$x_i(t+1) = x_i(t) + 0.2v_i(t+1)$$

**Inicialmente ( $t=0$ ),**

Partícula<sub>1</sub>:  $x_1(0)=0$ ,  $b_1(0)=0$ ,  $v_1(0)=0$

Partícula<sub>2</sub>:  $x_2(0)=1$ ,  $b_2(0)=1$ ,  $v_2(0)=0$

$B(0)=1$ ,  $v_1(0) = v_2(0) = 0$

**Paso-1 ( $t=1$ ),**

Partícula<sub>1</sub>:  $x_1(1) = x_1(0) + 0.2 v_1(1) = 0 + 0.2 [ 0.9 * 0 + (0 - 0) + (1 - 0)] = 0.2$ ;  $b_1(1)=0.2$

Partícula<sub>2</sub>:  $x_2(1) = x_2(0) + 0.2 v_2(1) = 1 + 0.2 [ 0.9 * 0 + (1 - 1) + (1 - 1)] = 1$ ;  $b_2(1)=1$

*An Introduction to Metaheuristics for Optimization, Chopard, Tomassini, Springer (2018)*

**Donde:**

$b_i(t)$  is the particle-best,  
 $B(t)$  is the global-best.

$$v_i(t+1) = 0.9v_i(t) + [b_i(t) - x_i(t)] + [B(t) - x_i(t)]$$

$$x_i(t+1) = x_i(t) + 0.2v_i(t+1)$$

**En t=1:**

$$x_1(1)=0.2, \quad b_1(1)= 0.2, \quad v_1(1)=1.0$$

$$x_2(1)=1, \quad b_2(1)=1, \quad v_2(1)=0.0$$

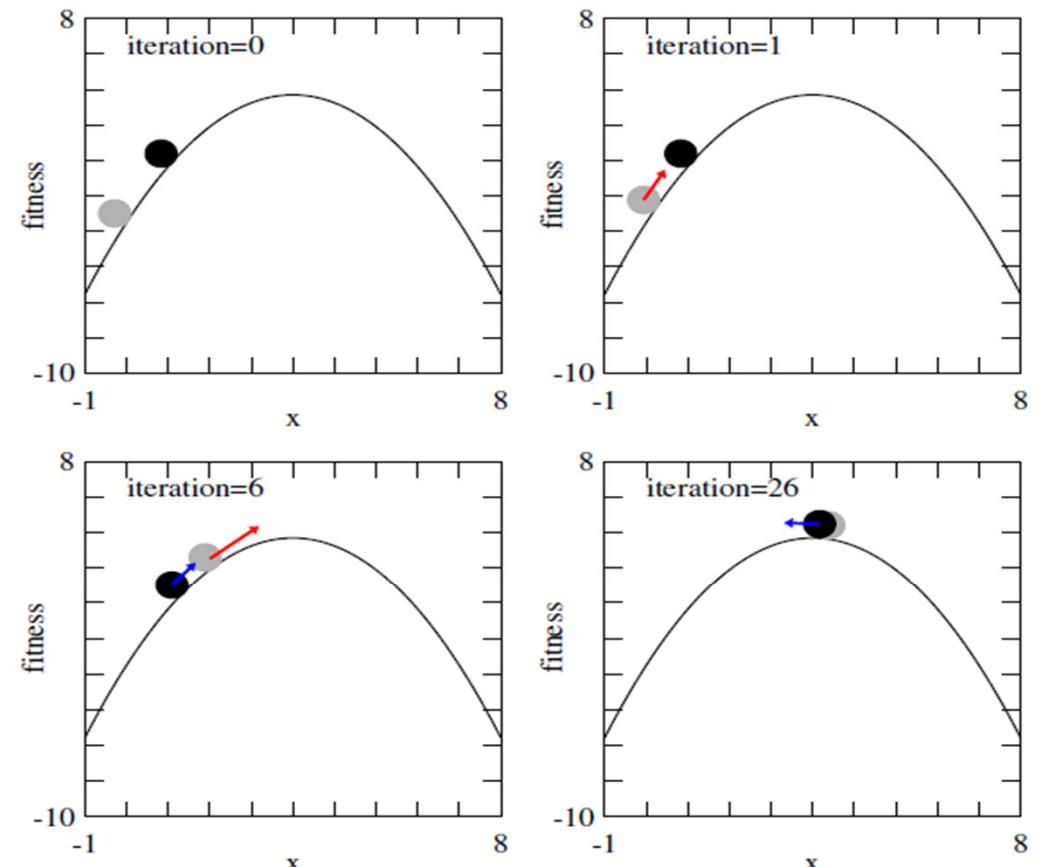
$$B(1)=1$$

**Paso-2 (t=2),**

$$\text{Partícula}_1: \quad x_1(2) = x_1(1) + 0.2 v_1(2) = 0.2 + 0.2 [ 0.9 * 1 + (0.2 - 0.2) + (1 - 0.2) ] = 0.2 + 0.2 * 1.7 = 0.54; \\ \quad b_1(2) = 0.54$$

$$\text{Partícula}_2: \quad x_2(2) = x_2(1) + 0.2 v_2(2) = 1 + 0.2 [ 0.9 * 0 + (1 - 1) + (1 - 1) ] = 1; \quad b_2(1)=1$$

**.... Paso 6:** Debido a la inercia, la Particula<sub>1</sub> arrastrará a la Particula<sub>2</sub> .....



PSO example with two particles in the one-dimensional space. To better illustrate the evolution, particles are shown here moving on the fitness curve; actually, **they only move along the x axis**

## Ejemplo 2a. Dos dimensiones

Variable:  $x_1, x_2 \in [-10, +10]$

Objetivo: maximizar  $f(x,y) = 100 - (x^2+y^2)$

Nº partículas: 2

W= 0.9, C1, C2 =1

Inicialmente,

$$x_1(0) = (6,4), \quad b_1(0) = 100 - (6^2 + 4^2) = 48$$

$$x_2(0) = (3,8), \quad b_2(0) = 100 - (3^2 + 8^2) = 27$$

$B(0) = (6,4)$ , mejor posición global

$$v_1(0) = v_2(0) = (0,0)$$

**Función de Vuelo  $v'$ , Actualización de la posición  $x'$ :**

$$v_i(t+1) = 0.9v_i(t) + [b_i(t) - x_i(t)] + [B(t) - x_i(t)]$$

$$x_i(t+1) = x_i(t) + 0.2v_i(t+1)$$

**Paso-1,**

$$\text{Partícula}_1: \quad x_1(1) = x_1(0) + 0.2 v_1(1) = (6,4) + 0.2 [ 0.9 * (0,0) + ((6,4) - (6,4)) + ((6,4) - (6,4)) ] = (6,4);$$

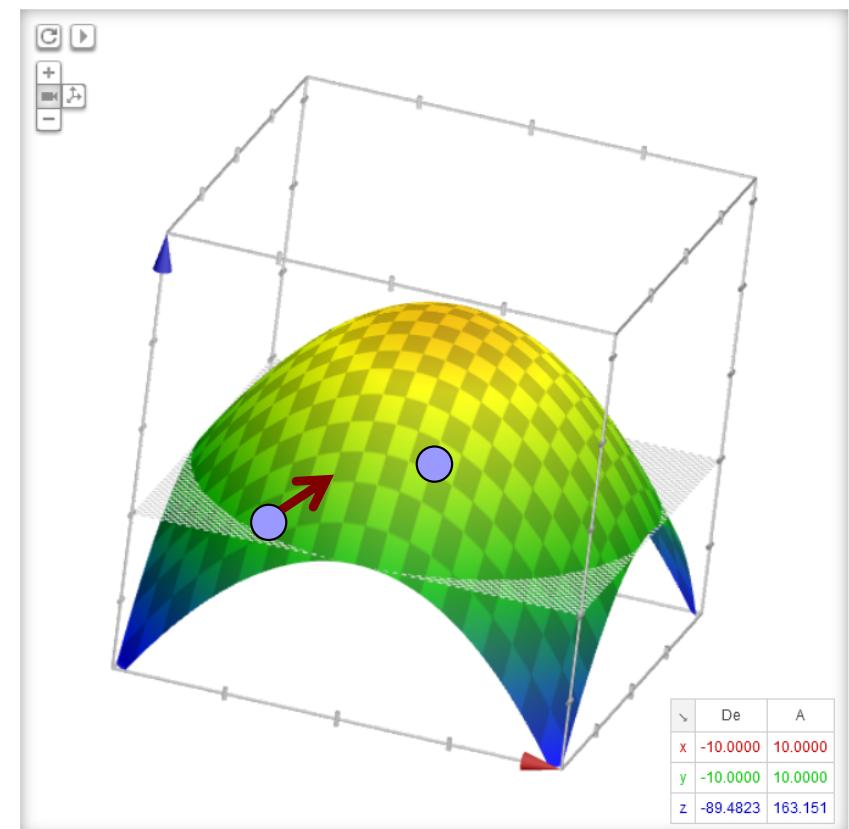
$$\mathbf{b}_1(1) = (6,4)$$

$$\begin{aligned} \text{Partícula}_2: \quad x_2(1) &= x_2(0) + 0.2 v_2(1) = (3,8) + 0.2 [ 0.9 * (0,0) + ((3,8) - (3,8)) + ((6,4) - (3,8)) ] = \\ &(3,8) + 0.2 (3, -4) = (3, 8) - (0.6, -0.8) = \underline{(2.4, 7.2)} \quad f(2.4, 7.2) = 42.4 ; \quad \mathbf{b}_2(1) = (2.4, 7.2) \end{aligned}$$

$$\mathbf{B}(1) = (6,4)$$

Gráfico de  $100-x^2+y^2$

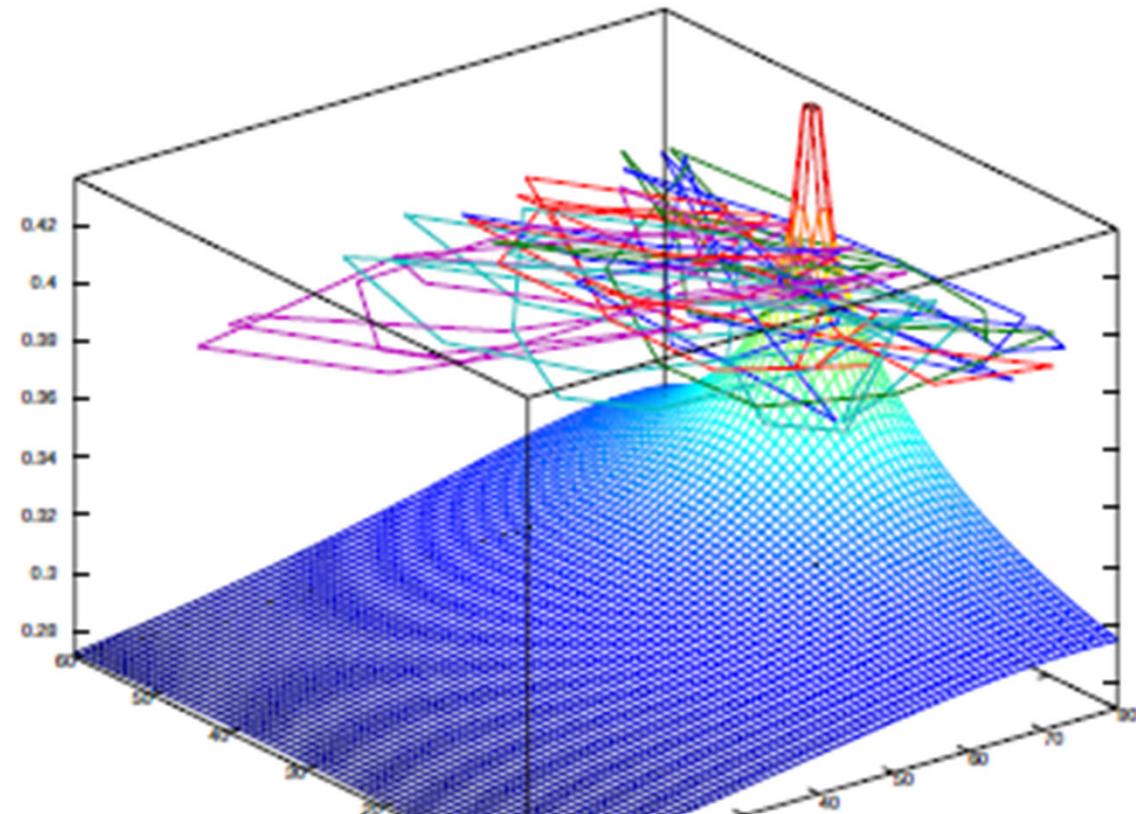
$$z=100 - (x^2+y^2)$$



## Ejemplo 2b. Dos dimensiones

*An example of PSO in 2D on a single-maximum fitness landscape.*

Global optimum at: (75, 36);  
fitness value at the global optimum: 0.436

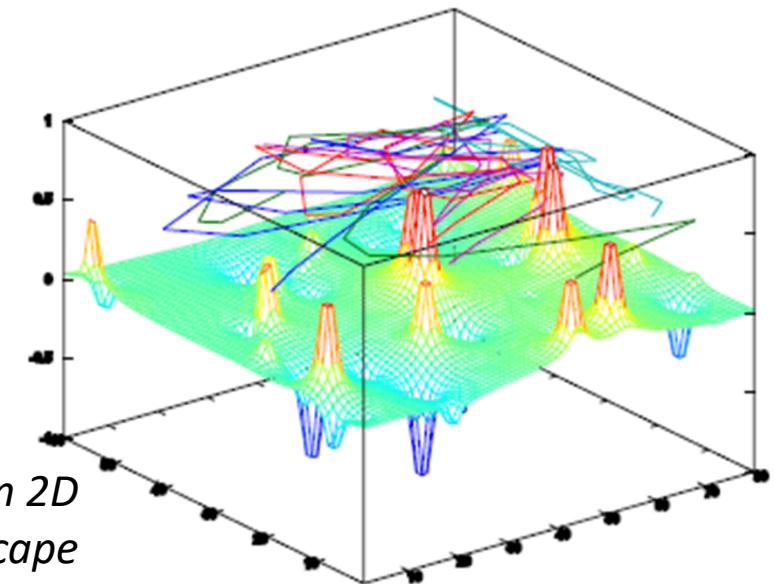


*With five particles, 50 iterations,  
the best solution found was  $B = (75; 39)$   $f(B) = 0.384$ .*

*With 20 particles, 100 iterations, the optimal solution was found*

*An Introduction to Metaheuristics for Optimization,  
Chopard, Tomassini, Springer (2018)*

*An example of PSO in 2D  
with a multimodal fitness landscape*



## Función de Vuelo

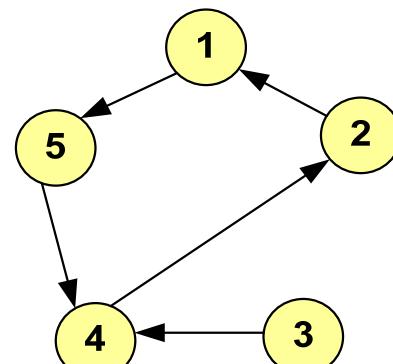
$$v'_{ik} = c_2 * \text{RND}() * (g_k - x_{ik}) + c1 * \text{RND}() * (p_{ik} - x_{ik}) + w * v_{ik}$$

$$X_{i+1} = X_i \otimes V'_i$$

*La operación de vuelo es dependiente del diseño del sistema, y tiene diferentes variantes.*

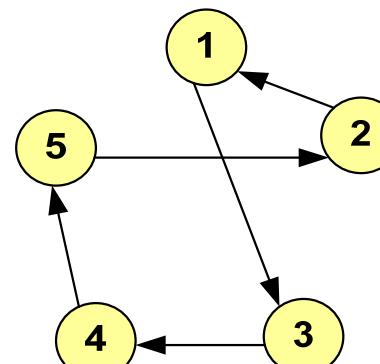
Ejemplo -3 TSP: Cada posición  $X_i$  de las partículas es un recorrido (solución en un espacio de soluciones)

Mejor del Enjambre



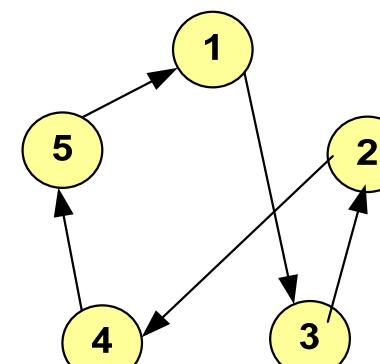
$$G = [3 \ 4 \ 2 \ 1 \ 5]$$

Mejor de la Partícula



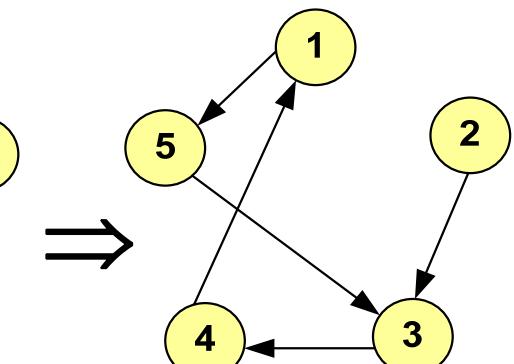
$$P_i = [5 \ 2 \ 1 \ 3 \ 4]$$

Posición Actual



$$X_i = [4 \ 5 \ 1 \ 3 \ 2]$$

Nueva Posición

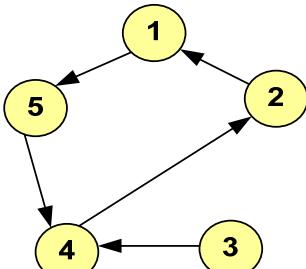


$$X'_i = [2 \ 3 \ 4 \ 1 \ 5]$$

**Varias alternativas.** Ejemplos en: *Particle Swarm Optimization Algorithm for the Traveling Salesman Problem.* E. Goldbarg, M. Goldbarg, G. Souza. In: 'Travelling Salesman Problem'. **En Poliformat**

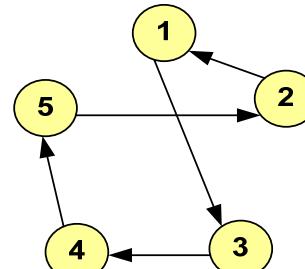
### Ejemplo-3 TSP

Mejor del Enjambre



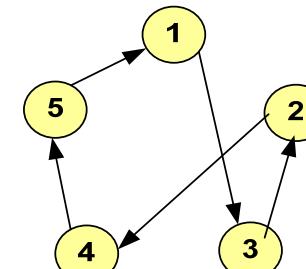
$G: [3 \ 4 \ 2 \ 1 \ 5]$

Mejor de la Partícula



$P_i: [5 \ 2 \ 1 \ 3 \ 4]$

Posición Actual



$X_i: [4 \ 5 \ 1 \ 3 \ 2]$

$$G = [3 \ 4 \ 2 \ 1 \ 5]$$

$$X_i = [4 \ 5 \ 1 \ 3 \ 2]$$

$$G - X_i = [-1 \ -1 \ 1 \ -2 \ 3]$$

$$P_i = [5 \ 2 \ 1 \ 3 \ 4]$$

$$X_i = [4 \ 5 \ 1 \ 3 \ 2]$$

$$P_i - X_i = [1 \ -3 \ 0 \ 0 \ 2]$$

Supongamos Velocidad Partícula:

$$V_i = [3 \ 5 \ 2 \ 6 \ -8]$$

$$v'_{ik} = c_2 * \text{RND}() * (g_k - x_{ik}) + c1 * \text{RND}() * (p_{ik} - x_{ik}) + w * v_{ik}$$

Asumiendo **C2=0,1, C1=0,3, w=0,6**, resulta:  $v'_{ik} = 0.1 * \text{RND}() * (g_k - x_{ik}) + 0.3 * \text{RND}() * (p_{ik} - x_{ik}) + 0.6 * v_{ik}$

$$v'_{i1} = 0.1 * \text{RND}() * (-1) + 0.3 * \text{RND}() * 1 + 0.6 * 3 = 2$$

$$v'_{i2} = 0.1 * \text{RND}() * (-1) + 0.3 * \text{RND}() * (-3) + 0.6 * 5 = 3$$

$$v'_{i3} = 0.1 * \text{RND}() * 1 + 0.3 * \text{RND}() * 0 + 0.6 * 2 = 1$$

$$v'_{i4} = 0.1 * \text{RND}() * (-2) + 0.3 * \text{RND}() * 0 + 0.6 * 6 = 3$$

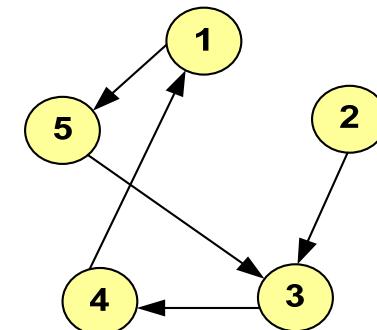
$$v'_{i5} = 0.1 * \text{RND}() * 3 + 0.3 * \text{RND}() * 2 + 0.6 * (-8) = -4$$

Luego,  $V'i = [2 \ 3 \ 1 \ 3 \ -4]$ . Tal que:  $X_{i+i} = X_i \otimes V'_i$

⊗: Representa los desplazamientos (swap) de cada nodo en el vector:

$$X_{i+i} = X_i \otimes V'_i = [4 \ 5 \ 1 \ 3 \ 2] \otimes [2 \ 3 \ 1 \ 3 \ -4] = [2 \ 3 \ 4 \ 1 \ 5]$$

Nueva Posición



[2 \ 3 \ 4 \ 1 \ 5]

## Ejemplo-4 TSP

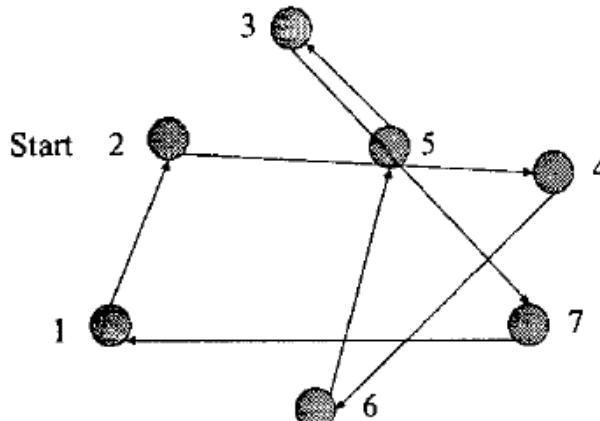
Aplicamos una nueva **Función de vuelo** para obtener la nueva posición  $X_{i+1}$  de la partícula<sub>i</sub>:

$$X_{i+1} \leftarrow [RND] \otimes (C_1(\%) * [X_i], C_2(\%) * [P_i], C_3(\%) * [G])$$

*La función de vuelo  $\otimes$  representa la determinación probabilística [RND] de la nueva posición  $X_{i+1}$  en base a la posición actual  $[X_i]$ , a la información local  $[P_i]$ , y a la información global  $[G]$ .*

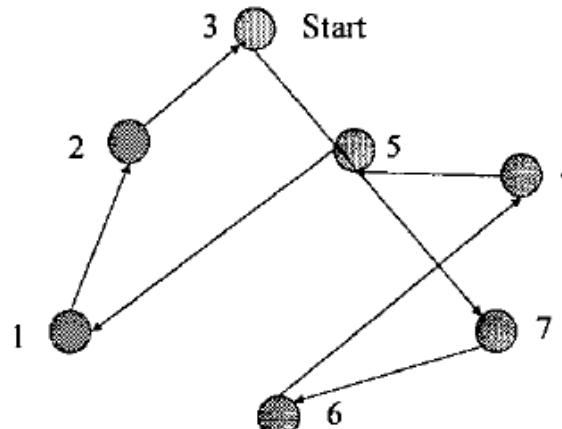
*La determinación se basa en los valores (probabilidades) de  $C_1$ ,  $C_2$ ,  $C_3$ , donde  $C_1 + C_2 + C_3 = 100\%$ .*

Current Position



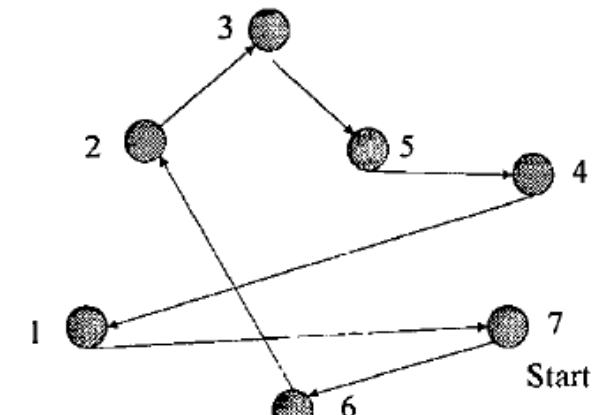
$$X_i = \{2, 4, 6, 5, 3, 7, 1\}$$

Global Best



$$G = \{3, 7, 6, 4, 5, 1, 2\}$$

Local Best



$$P_i = \{7, 6, 2, 3, 5, 4, 1\}$$

*TSP for surveillance mission using particle swarm optimization , Ph. D. Thesis. Bany Secrest,  
Air force Institute, of Technology. Ohio, USA.*

Aplicamos una nueva **Función de vuelo** para obtener la nueva posición  $X_{i+1}$  de la partícula<sub>i</sub>:

$$X_{i+1} \leftarrow [RND] \otimes (C_1(\%) * [X_i], C_2(\%) * [P_i], C_3(\%) * [G])$$

Asumamos:  $C_1 = 10\%$ ,  $C_2 = 10\%$ ,  $C_3 = 80\%$ , y la aleatoriedad [RND] resulta: {60, 82, 87, 26, 94}

RULETA

**Estado Actual:**  $X_i = \{2, 4, 6, 5, 3, \textcolor{red}{7}, 1\}$        $G = \{3, 7, 6, 4, 5, 1, \textcolor{red}{2}\}$        $P_i = \{7, 6, 2, \textcolor{red}{3}, \textcolor{red}{5}, \textcolor{red}{4}, 1\}$

### PROCESO:

**1<sup>a</sup> Ciudad:** *Por defecto*, se elige como ciudad inicial la última ciudad de  $X_i$ :  $X_{i+1}[1] = \underline{1}$

**2<sup>a</sup> Ciudad:** Dado que  $RND(1)=60$  es menor que  $C_3$ , se elige la información global (G). Se elige 2, ya que es la que sigue a 1 en G.

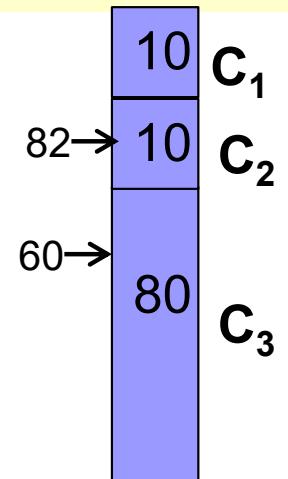
**3<sup>a</sup> Ciudad:** Dado que  $RND(2)=82$  es mayor que  $C_3$ , pero menor que  $C_3+C_2$ , se elige la información local (P). Se elige 3, ya que es la que sigue a 2 en  $P_i$ .

**4<sup>a</sup> Ciudad:** Dado que  $RND(3)=87 \in \{C_3, C_3+C_2\}$ , se elige la información local (P). Se elige 5, ya que es la que sigue a la ciudad 3.

**5<sup>a</sup> Ciudad:** Dado que  $RND(4)=26 < C_3$ , se elegiría 1 por la información global (G), pero 1 ya ha sido elegida. Se intenta por la información local (P) y se elige 4, ya que es la que sigue a 5.

**6<sup>a</sup> Ciudad:** Por aleatoriedad, dado que  $RND(5)=94 > C_3+C_2$ , se elige la información previa (X). Se elige 7, ya que es la penúltima ciudad del recorrido de  $X_i$ .  $X_{i+1}[6] = 7$

**7<sup>a</sup> Ciudad:** La ciudad que queda es la 6.  $X_{i+1}[7] = 6$



Función de vuelo para obtener la nueva posición  $X_{i+1}$  de la partícula<sub>i</sub>:

$$X_{i+1} \leftarrow RND() \otimes [C_1(\%) X_i, C_2(\%) P_i, C_3(\%) G]$$

Donde la función  $\otimes$  representa la selección probabilística (RND) de la nueva posición  $X_{i+1}$  en base a la posición actual ( $X_i$ ), a la información local ( $P_i$ ), y a la información global ( $G$ ).

Estado Actual:  $X_i = \{2, 4, 6, 5, 3, 7, 1\}$        $G = \{3, 7, 6, 4, 5, 1, 2\}$        $P_i = \{7, 6, 2, 3, 5, 4, 1\}$

RULETA

10

$C_1$

82 → 10

$C_2$

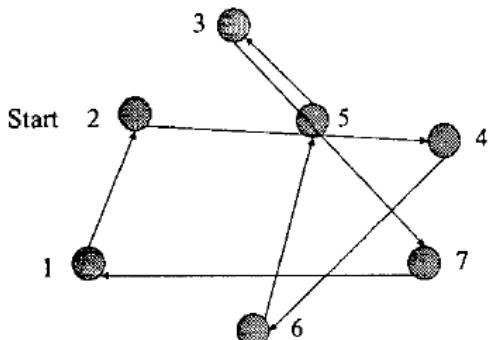
60 →

80

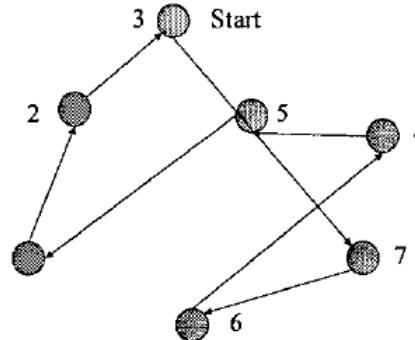
$C_3$

Nuevo recorrido  $X_{i+1} = \{1, 2, 3, 5, 4, 7, 6\}$

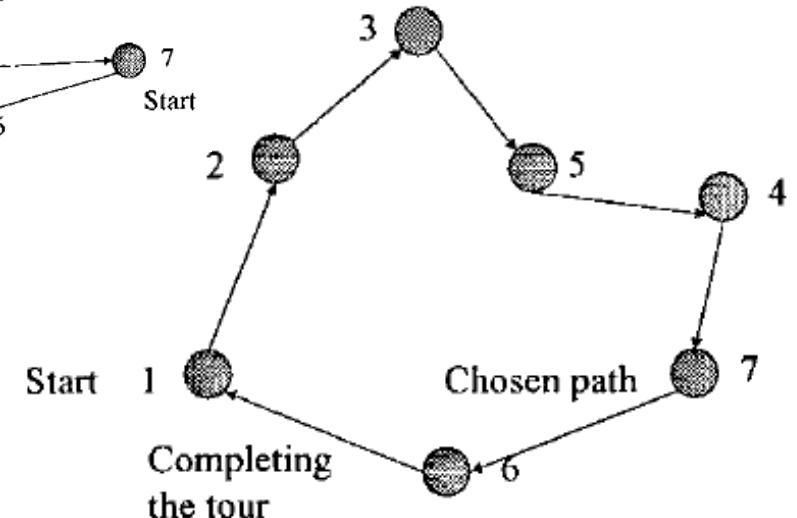
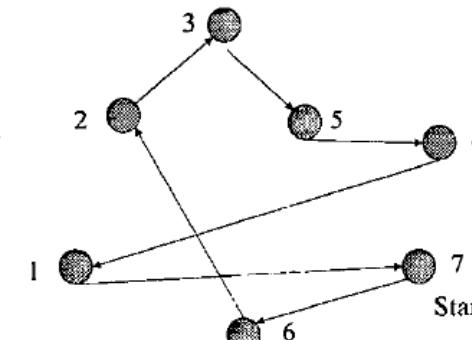
Current Position



Global Best



Local Best



Resumen:

<https://www.youtube.com/watch?v=JhgDMAm-imI>

- Metaheurística de optimización basada en modelos de conductas sociales.
- Requiere un conjunto de ‘soluciones iniciales (partículas)’ que van ‘moviéndose’ en espacio de soluciones siguiendo unas ‘reglas matemáticas’ (inercia, propias, sociales).
- Combina *exploración* y *explotación*, ajustable con los parámetros del método:

$$\dot{c}_1 \quad w \text{ (inercia)}, \quad c_1 \text{ (factor propio)}, \quad c_2 \text{ (factor social)} \quad ??$$

Todavía no se conoce bien la relación entre los parámetros y los conceptos de exploración y explotación.

- Asume muy pocas hipótesis del problema o del espacio de soluciones (posición y velocidad de cada partícula y la mejor posición global).
- Doble concepto de convergencia:
  - Convergencia hacia la ‘*mejor solución*’ (mejor posición global g)
  - Convergencia del enjambre hacia una *zona de ‘buenas soluciones’* (que pueden no incluir la óptima)
- Amplia utilización actual. Variantes y aplicaciones.

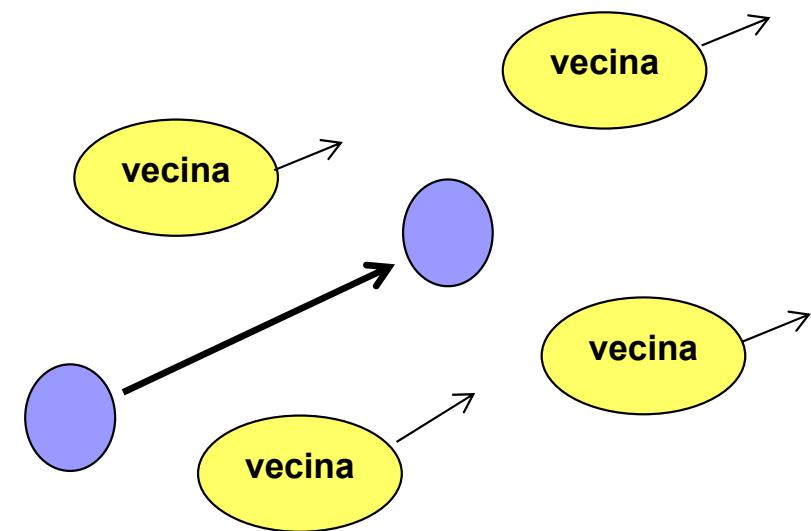
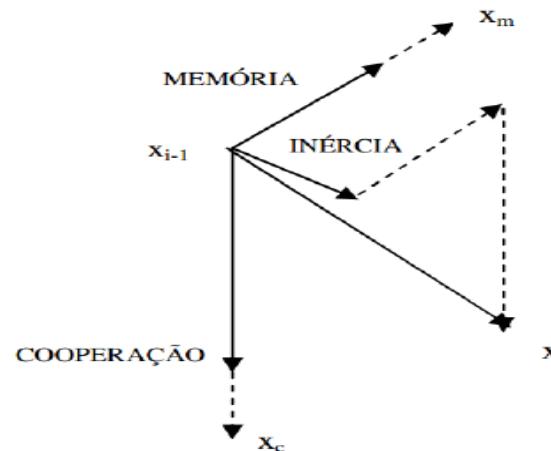
Existen muy diversas variantes PSO:

- Aplicación a problemas multi-objetivo (considerando Frontera de Pareto).
- Con o sin líder (únicamente *factor inercial* y memoria)
- Clustering: Agrupación de partículas (mejores globales de cada grupo o líderes locales),
- Introducir una fuerza repulsiva entre las partículas (para evitar convergencia prematura),
- Perturbaciones momentáneas de las partículas ( $\cong$  mejora local)
- **Variante APSO:** Utilizar solo 'g' (mejor global): *accelerated particle swarm optimization (APSO, Xin-She Yang 2021)*:

$$v_i^{t+1} = v_i^t + \beta(g^* - x_i^t) + \alpha\epsilon_t \quad \text{simplificada como} \quad x_i^{t+1} = (1 - \beta)x_i^t + \beta g^* + \alpha\epsilon_t$$

donde  $\epsilon$  es una distribución gaussiana  $\{0,1\}$ , y típicamente  $\alpha \approx 0.1 \sim 1$  and  $\beta \approx 0.1 \sim 0.7$ , ( $\alpha \approx 1$ ,  $\beta \approx 0.5$ )

- **Parametrización:** Incremento/decremento de la inercia (w), memoria (c1), cooperación social (c2),
- etc., etc, etc.....



### Integrated Particle Swarm Optimization (iPSO)

- En PSO estándar, la actualización de cada partícula ( $X_i$ ) se guía por dos posiciones significativas del espacio de búsqueda:  $P_i$ ,  $G$ .
- Si la partícula actual se encuentra muy cerca de cualquiera de estas dos posiciones, sus capacidades de orientación se reducen o incluso se desvanecen.

iPSO incorpora una tercera partícula, llamada partícula ponderada ( $X^W$ ):

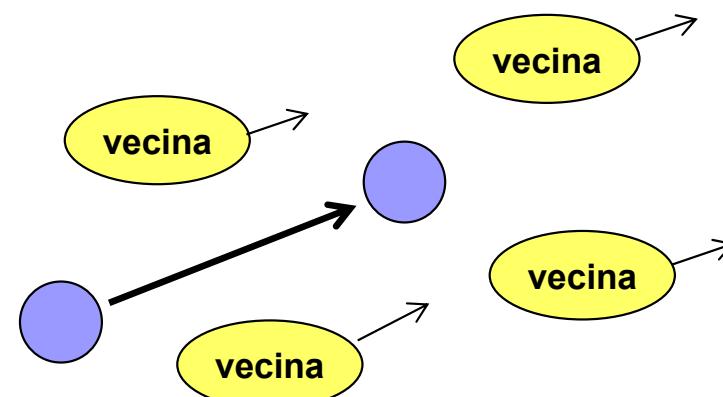
$X^W$  es la posición ponderada media de las mejores posiciones de cada una de las M partículas del enjambre:

$$X^W = \sum_{j=1,M} c_j^w * P_j \quad ; \text{ siendo } c_j^w \text{ el factor de ponderación para cada partícula } P_j$$

Esta nueva partícula se incorpora en una revisada función de cruce

(*Interactive search algorithm: A new hybrid metaheuristic optimization algorithm.*

*Mortazavi, Toğan, Nuhoglu. Engi.App. of Artificial Intelligence, 71, May 2018).* **En Poliformat**



### Hybrid Gravitational Search Algorithm & Particle Swarm Optimization (PSO – GSA)

Las partículas en un PSO clásico siguen unas reglas matemáticas de movimiento:

$$x_i(t+1) = x_i(t) + v_i(t), \quad \text{donde} \quad v_i = w * v_i + C_1(p_{ik} - x_{ik}) * \text{RND} + C_2(g_k - x_{ik}) * \text{RND}$$

En el **método PSO, combinado con una Búsqueda Gravitacional (GSA)**, se asume que:

- Las partículas tienen '*masa*', tal que a mayor masa, generan mayor fuerza gravitacional. *La masa depende de la bondad (fitness) de la partícula.*
- Aparece una '*fuerza de atracción*' entre partículas, que depende de sus masas (fitness) y de su distancia relativa (en el espacio de soluciones).
- La fuerza de atracción provoca una '*aceleración*' ( $a_i$ ) del movimiento de cada partícula, que depende de la combinación ( $\Sigma$ ) de las fuerzas de atracción a la que está sometida por el resto de partículas.
- La nueva ecuación de velocidad es:

$$v_i(t+1) = w \times v_i(t) + C_1 \times a_i(t) + \left( \frac{C_1}{C_2} \right) \times (p_g - x_i)$$

Mejora:

El PSO clásico no considera el fitness ni la distancia entre (todas) las partículas en el movimiento.

*El método PSO-GSA incluye una afectación común del 'fitness' y 'distancia' entre todas las partículas.*

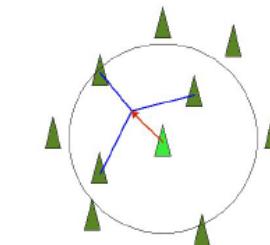
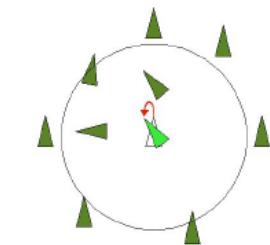
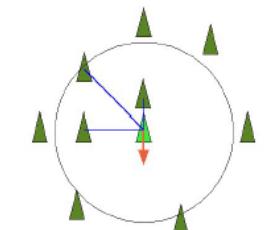
**A novel hybrid gravitational search particle swarm optimization algorithm.** Kanh, Ling.  
Engineering Applications of Artificial Intelligence Volume 102, June 2021 (En Poliformat)

### Modelado de movimientos de grupos, vida artificial, flocking

Flocking: Movimiento colectivo de una alto número de entidades autónomas (rebaños, bandadas, etc.)

Consideración de otros factores o reglas en el movimiento de las partículas:

- *Separación*: Cada partícula tiende a moverse lejos de sus vecinos si estos demasiado cerca. Mantiene distancia razonable, pero no demasiado lejos (*repulsión posición vecinos*).
- *Alineación*: Cada partícula sigue la *dirección media* del movimiento de sus vecinas e igualando su velocidad: búsqueda en el entorno local alineada con la búsqueda de sus partículas vecinas.
- *Cohesión*: Las partículas siguen una dirección global y mantienen una cohesión de grupo con sus vecinas, tendiendo ir al centro del enjambre. Cada partícula tiende a moverse hacia la *posición media* de su vecindad.



Modelo inicial flocking: Flocks, herds and scools: A *Distributed Behavioral Model*, Craig W. Reynolds. *Computer Graphics*, 21(4), July 1987. *En Poliformat*

**Acknowledgements:** Me gustaría agradecer a las bandadas, rebaños y escuelas por existir.

Entorno Boids: <http://www.red3d.com/cwr/boids/>

Demo: <https://owenmcnaughton.github.io/Boids.js/>



## PSO aplicable con buenos resultados en problemas multi-objetivo con variables reales.

- *When Evolutionary Computing Meets Astro and Geoinformatics.* Chellym Mirchev. In Knowledge Discovery in Big Data from Astronomy and Earth Observation Elsevier (2020)
- *Different Applications of PSO* Altaf Q. H. Badar. Applying Particle Swarm Optimization, International Series (2021)
- *Application of particle swarm optimization algorithm in power system problems.* Zargari, Heris, vatloo. Handbook of Neural Computation. Elsevier 2017
- *Particle Swarm Optimization: A Comprehensive Survey.* Varios autores, IEEE Access, 2022
- *Enhanced particle swarm optimization with multi-swarm and multi-velocity for optimizing high-dimensional problems.* Applied Intelligence (2019)

PSO Tutorial: <http://www.swarmintelligence.org/tutorials.php>

PSO Software: <http://www.borgelt.net/psopt.html>

Nuevas Variantes y adaptaciones a tipos de problemas.

.

# ....Otras Metaheurísticas Sociales

Bio-inspirada: El objetivo de una colonia de abejas es encontrar buenas zonas de alimento (flores) y maximizar la cantidad de néctar recogido.

El método supone 3 tipos de abejas:

- **Exploradoras:** exploran áreas amplias, buscando nuevas fuentes de alimento.



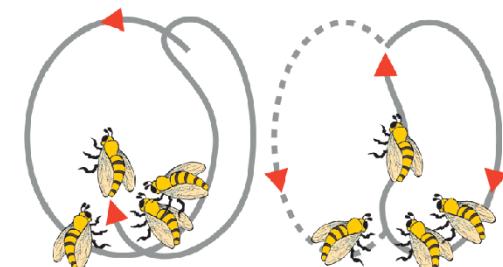
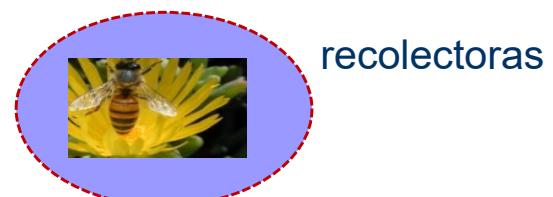
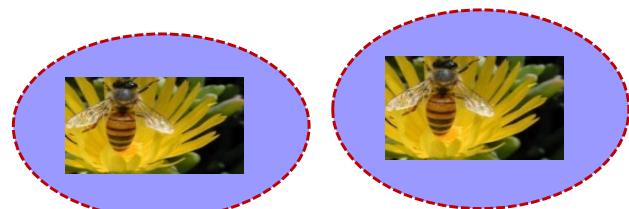
Cuando encuentran nuevas fuentes de alimento, evalúan la cantidad de néctar y retienen en su memoria solo la mejor zona encontrada. Aproximadamente son el 10%.



- **Recolectoras:** viajan a una fuente concreta de alimento y recolectan néctar. También examinan y recolectan en las zonas de alimento cercanas.

Cuando las abejas exploradoras y las recolectoras regresan a la colmena, inician un baile **compartiendo la información** que poseen (ubicación y calidad de las fuentes encontradas).

- **Expectadoras:** esperan en la entrada de la colmena a que vuelvan las exploradoras y las recolectoras. Reciben la información del baile, y pasan a ser abejas recolectoras de las mejores zonas.



expectadoras

Solución  $\Leftrightarrow$  Posición de la fuente de alimento

Calidad Solución  $\Leftrightarrow$  Cantidad de Néctar de la fuente

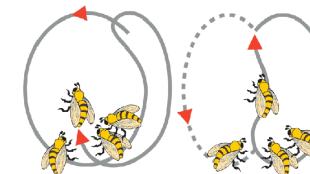
Abeja (exploradora, recolectora)  $\Leftrightarrow$  Proceso de búsqueda

**Demos:**

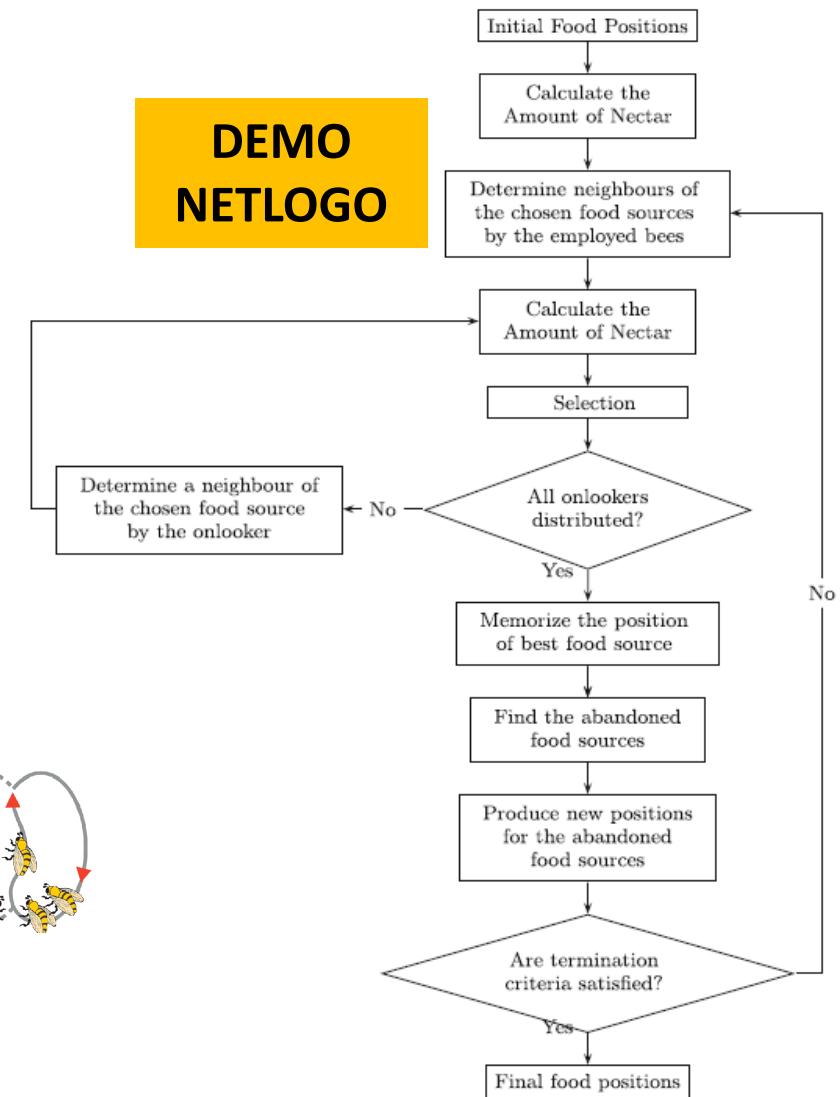
<http://mf.erciyes.edu.tr/abc/>

## ESQUEMA BÁSICO:

- Nivel exploración: Obtención de nuevas soluciones.
  - Lanzar procesos exploradores en busca de soluciones.
  - Se forma una población de soluciones.
- Nivel explotación: Búsqueda Local para mejora de soluciones
  - Lanzar procesos explotadores, explotando soluciones cercanas a obtenidas.
  - Se añaden a la población de soluciones
- Evaluación de soluciones (Danza de las abejas)
  - Evaluación de soluciones (néctar obtenido por exploradoras y recolectoras).
  - Se lanzan nuevos procesos (abejas expectadoras) que explotan las mejores soluciones de la población.
- Ciclo, hasta condición de terminación



## DEMO NETLOGO



## Parámetros:

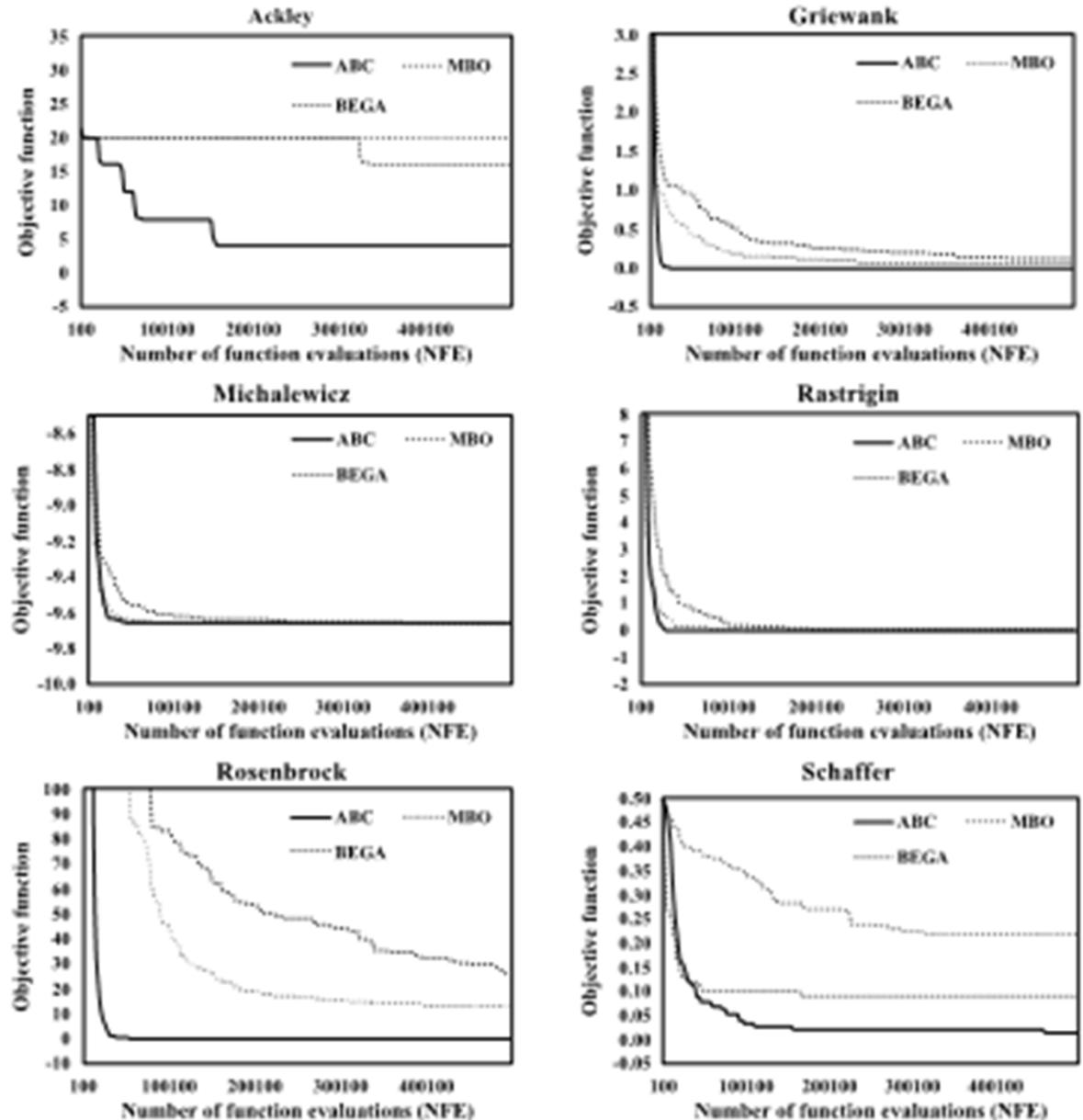
- Número de procesos totales,
- Número de procesos de cada tipo (%), o soluciones seleccionadas para ser mejoradas.

## Resumen Método

- Combina de forma equilibrada exploración y explotación.
- Algoritmo de aprendizaje cooperativo: Combina ‘autonomía’, ‘búsqueda distribuida’, ‘auto-organización’ y ‘aprendizaje’.
- Realiza búsquedas exploración/explotación en paralelo.: mejora soluciones (búsqueda), o construye soluciones (constructivo).

## Evaluación sobre 6 diferentes problemas

- Bee Colony (ABC)
- Genetic Algorithm (BEGA)
- Particle Swarm (bee swarm)



Average objective function of runs of ABC, MBO, and BEGA for 10-dimensional test functions

**Bee-inspired metaheuristics for global optimization: a performance comparison.** Solgi, Loálciga,

*Artificial Intelligence Review, 54 (2021)*

# Algoritmo inspirado en murciélagos (Bat Algorithm - BA)

*A New Metaheuristic Bat-Inspired Algorithm.* X. S. Yang

Nature Inspired Cooperative Strategies for Optimization, Studies in Computational Intelligence, 284, (2010)

Cada murciélagos virtual vuela **al azar** con una velocidad  $v_i$  a la posición (solución)  $x_i$  con una **frecuencia de onda e intensidad**  $A_i$ .

Cuando localiza una presa, **cambia su frecuencia e intensidad** y la búsqueda se intensifica como una búsqueda **local**, que continua hasta que se cumpla el criterio de parada.

El proceso completo se repite iterativamente hacia una nueva presa.

De esta forma, se asume que, mediante el ajuste de la frecuencia, se puede controlar el balance entre exploración y explotación.

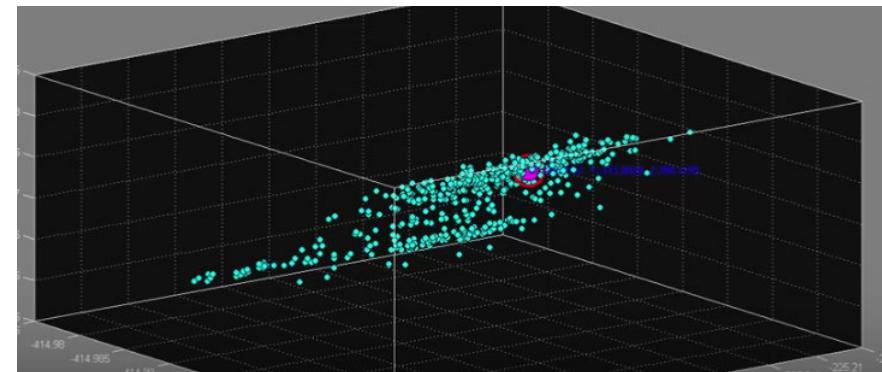


An improved discrete bat algorithm for symmetric and asymmetric Traveling Salesman Problems

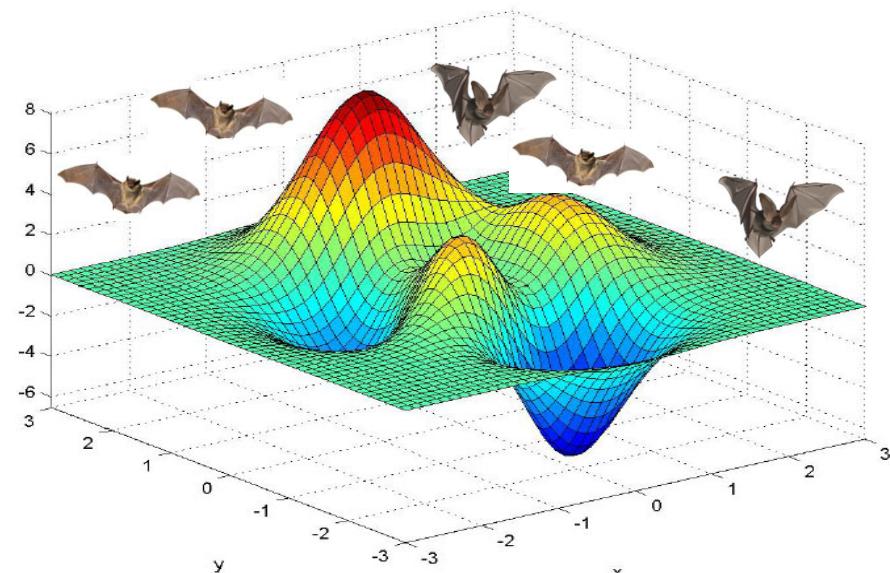
Eneko Osaba <sup>a,b,\*</sup>, Xin-She Yang <sup>b</sup>, Fernando Diaz <sup>a</sup>, Pedro Lopez-Garcia <sup>a</sup>, Roberto Carballido <sup>a</sup>

<sup>a</sup> Deusto Institute of Technology (DeustoTech), University of Deusto, Av. Universidades 24, Bilbao 48007, Spain

<sup>b</sup> School of Science and Technology, Middlesex University, Hendon Campus, London, NW4 4BT, United Kingdom



La interacción entre partículas se basa en la **frecuencia e intensidad** de las vecinas



- Enjambre de Luciérnagas (**Glow worm swarm optimization**, Krishnanand, Ghose, 2005),
- Algoritmo de Luciérnagas (**Firefly Algorithm** , X. S. Yang 2009)

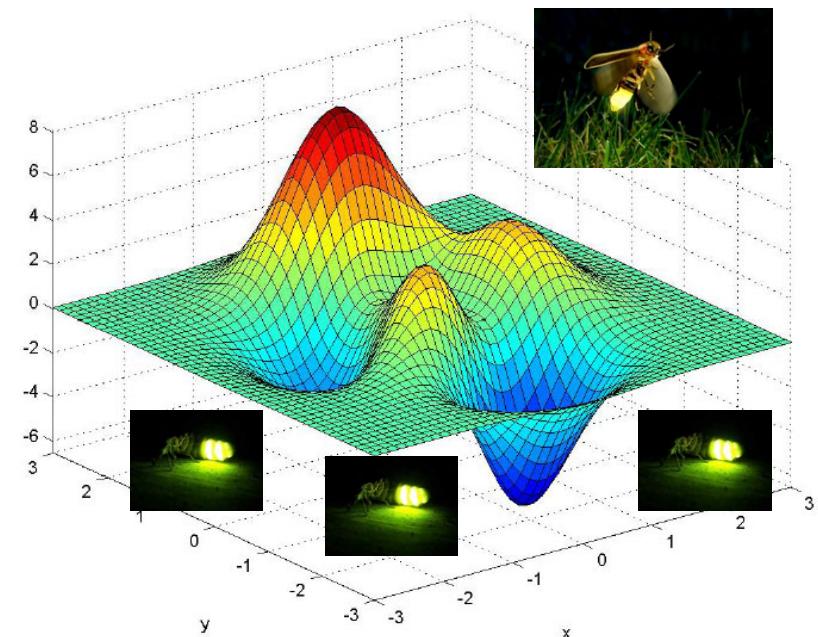
Google

- Las luciérnagas basan su comportamiento social en la luminosidad que emiten (luciferina).
- En su cortejo nocturno, los machos vuelan en busca de pareja mientras emiten destellos de luz característicos de cada especie.
- Las hembras de la misma especie pueden responder con destellos específicos y así el apareamiento puede ocurrir.
- Con similar planteamiento, la luminosidad de una luciérnaga depende de la calidad de la solución encontrada y la distancia desde donde las otras compañeras están buscando soluciones.
- Cada luciérnaga selecciona, utilizando un mecanismo probabilístico, un vecino que tiene un valor más alto de luciferina que su propio y se mueve hacia él.

**Aplicaciones:** Diseño de redes ópticas, Canales de comunicación, Infraestructuras (puentes), etc.



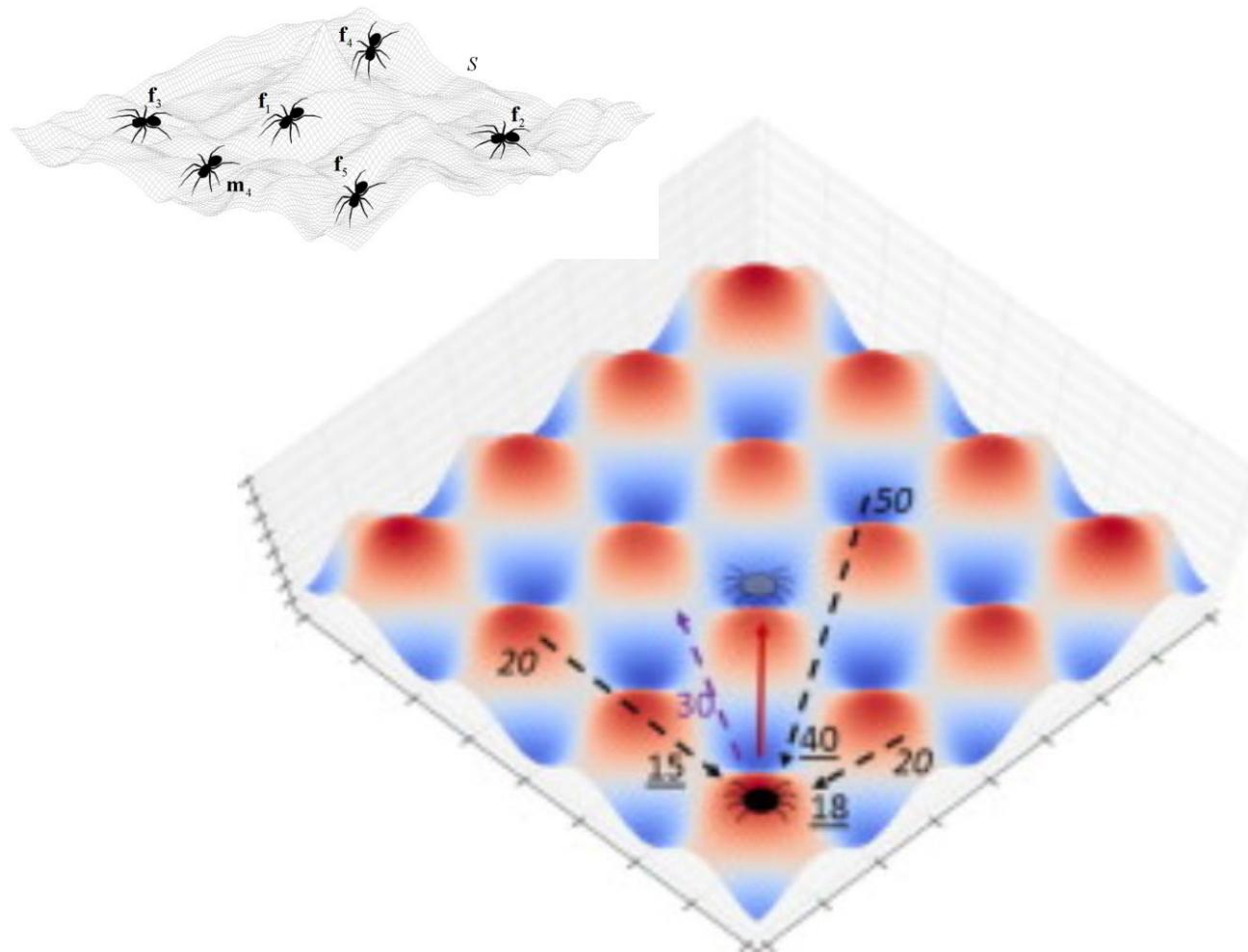
La interacción entre partículas se basa en la luminosidad (intensidad y color)



# Metaheurísticas de arañas sociales

A swarm optimization algorithm inspired in the behavior of the social-spider.

Cuevas, Cienfuegos, Zaldívar, Pérez-Cisneros. Expert Systems with Applications, 40 (16), (2013), pp. 6374-6384



Applied Soft Computing

Volume 30, May 2015, Pages 614–627



A social spider algorithm for global optimization

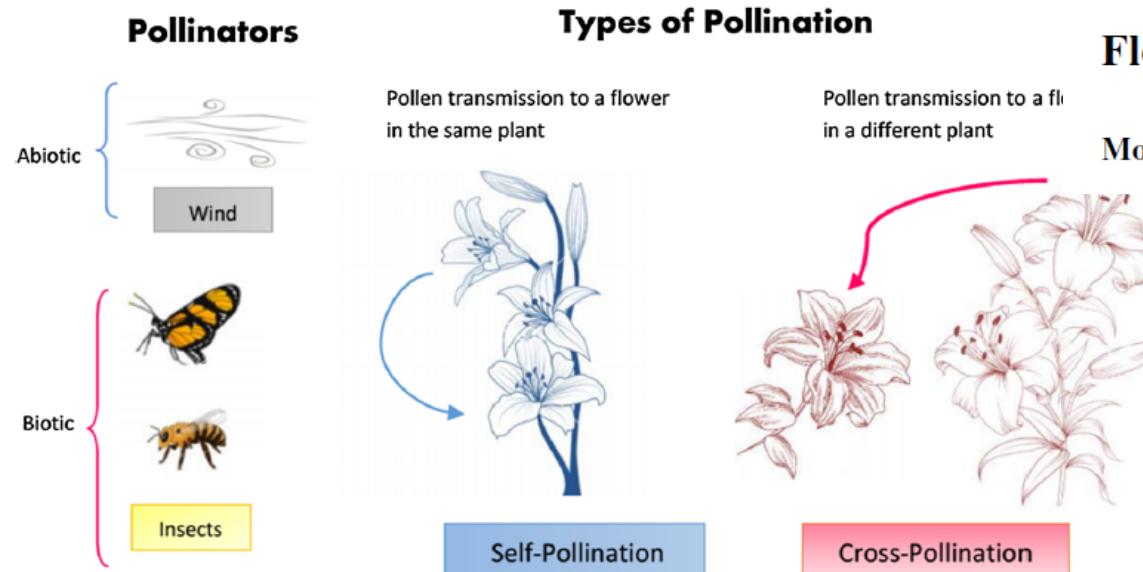
James J.Q. Yu , Victor O.K. Li

[Show more](#)

[doi:10.1016/j.asoc.2015.02.014](https://doi.org/10.1016/j.asoc.2015.02.014)

[Get rights and content](#)

# Flower pollination algorithm



## Flower pollination algorithm: a comprehensive review

Mohamed Abdel-Basset<sup>1</sup> · Laila A. Shawky<sup>1</sup>

*Metaheurística que toma su metáfora en la proliferación de flores en las plantas.*

```
1: Initialize parameters with switching probability  $p \in [0,1]$  ;  
2: Generate initial population of flowers randomly;  
3: Evaluate initial population and find the current best solution  $gbest$ ;  
4: while (stopping criterion not satisfied) do  
5:   For each flower  
6:     if  $rand() < p$   
7:       Global pollination :  $x_i^{t+1} = x_i^t + L(x_i^t - gbest)$  ; // Based on Lévy step  
8:     else  
9:       Select two random solutions  $x_j^t$  and  $x_k^t$  ;  
10:      Local pollination :  $x_i^{t+1} = x_i^t + \epsilon(x_j^t - x_k^t)$ ;  
11:    end if  
12:    Evaluate new solutions;  
13:    Update solutions with better new ones;  
14:  end for  
15:  Keep the current best solution;  
16: end while
```

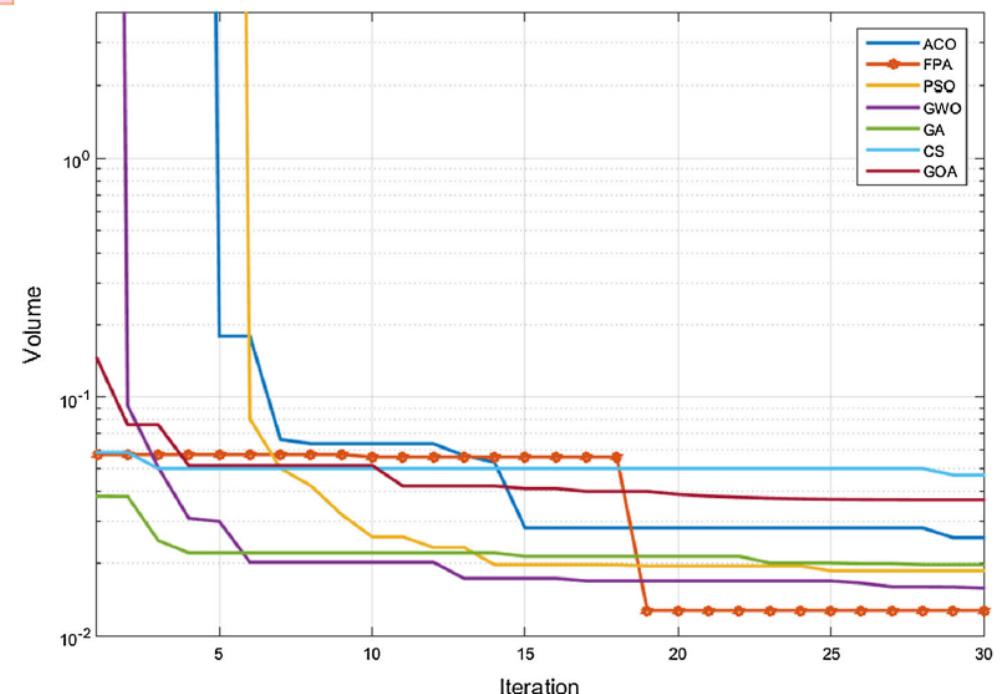
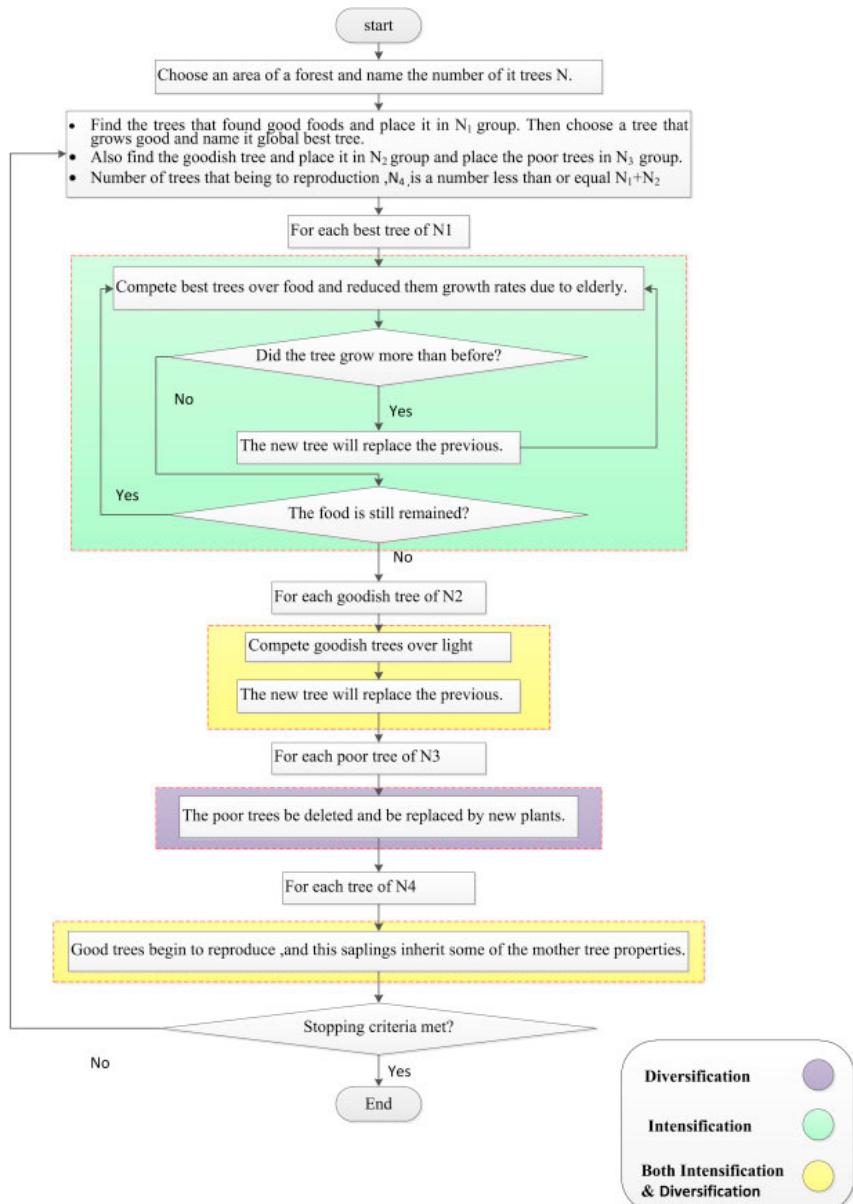


Fig. 4 Convergence of FPA and other metaheuristics

# Metaheurística basada en la conducta de los árboles

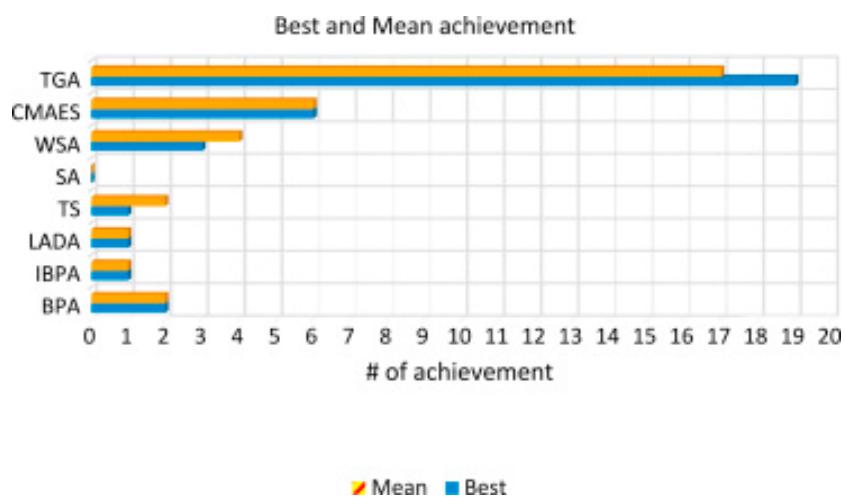
- Competencia para obtener luz y comida.



Engineering Applications of Artificial Intelligence  
Volume 72, June 2018, Pages 393–414



Tree Growth Algorithm (TGA): A novel approach for solving optimization problems



A comparison of novel metaheuristic algorithms on color aerial image multilevel thresholding.  
Kurban, Durmus, Karakose.

*Engineering Applications of Artificial Intelligence.*  
Vol. 105, Oct. 2021

Multilevel color image thresholding can be considered as an optimization problem that an algorithm should determine the optimum threshold values to obtain a perfectly segmented image. In recent years, metaheuristic algorithms become popular in several fields including image thresholding by their advantage of flexible structure. The motivation of this study relies on the fact that using same algorithm to solve any particular problem do not guarantee the best results as stated in no-free lunch theorem in optimization.

Therefore, six novel nature inspired algorithms such as **equilibrium optimization (EO), political optimizer (PO), turbulent flow of water-based optimization (TFWO), henry gas solubility optimization (HGSO), marine predators algorithm (MPA), and slime mould algorithm (SMA)** are chosen to be compared by determining the multilevel color image thresholding values.



## A comparison of novel metaheuristic algorithms on color aerial image multilevel thresholding

Rifat Kurban <sup>a</sup>, Ali Durmus <sup>b</sup>, Ercan Karakose <sup>c</sup>

Show more ▾

+ Add to Mendeley Share Cite

<https://doi.org/10.1016/j.engappai.2021.104410>

Get rights and content

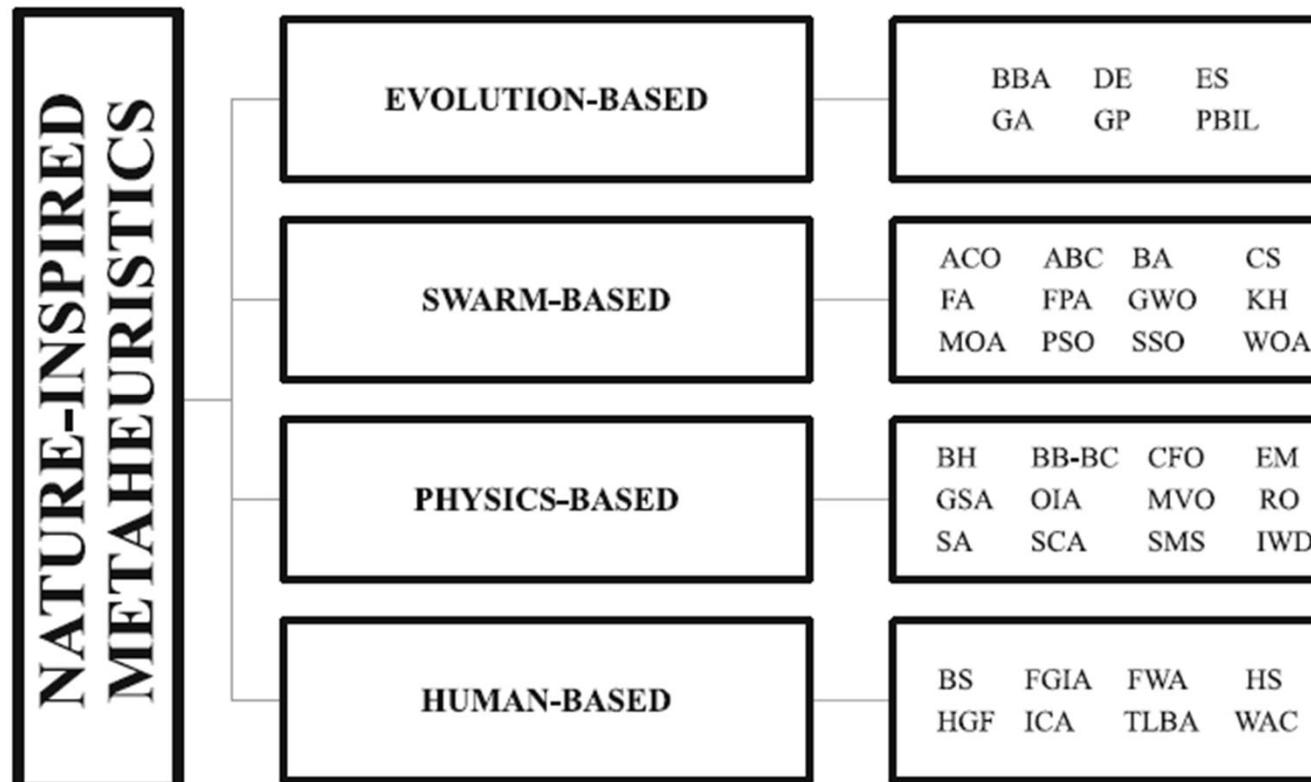
### Abstract

Color image thresholding is a well-known approach for image segmentation. An objective function, based on image entropy, is defined by threshold numbers and locations in the color histogram. Multiple classes of images can be created with multilevel thresholding. The main problem with thresholding techniques is to decide the threshold color values for each image. For a human operator, it is very hard to determine the specific threshold values of the images to be segmented. From this perspective, multilevel color image thresholding can be considered as an optimization problem that an algorithm should determine the optimum threshold values to obtain a perfectly segmented image. In recent years, metaheuristic algorithms become popular in several fields including image thresholding by their advantage of flexible structure. The motivation of this study relies on the fact that using same algorithm to solve any particular problem do not guarantee the best results as stated in no-free lunch theorem in optimization. Therefore, six novel nature inspired algorithms such as equilibrium optimization (EO), political optimizer (PO), turbulent flow of water-based optimization (TFWO), henry gas solubility optimization (HGSO), marine predators algorithm (MPA), and slime mould algorithm (SMA) are chosen to be compared by determining the multilevel color image thresholding values. These algorithms, which are used for the first time to solve this problem, are also compared statistically with extensive experiments. Aerial test images obtained by drones are used in the experiments. Kapur's entropy and between-class variance (Otsu's method) objectives are maximized by metaheuristic algorithms. Experimental results are evaluated with structural similarity (SSIM), peak-signal noise ratio (PSNR), blind/referenceless image spatial quality evaluator (BRISQUE), perception-based image quality evaluator (PIQE), natural image quality evaluator (NIQE), and computing CPU time consumption of the algorithms. Another problem of thresholding is that the question of how to compare the results objectively. Many image quality metrics may give incompatible results. Correlation analysis of the average ranking results show that image quality metrics used in this study are compatible to each other. Extensive experiments show that MPA and TFWO outperformed SMA, EO, PO and HGSO in terms of PSNR, SSIM, BRISQUE, PIQE, NIQE, and CPU time consumption for multilevel color aerial image thresholding.



## From ants to whales: metaheuristics for all tastes

Fernando Fausto<sup>1</sup> · Adolfo Reyna-Orta<sup>2</sup> · Erik Cuevas<sup>1</sup> · Ángel G. Andrade<sup>2</sup> ·  
Marco Pérez-Cisneros<sup>1</sup>



# *YinYang Optimization (Punnathanam, Kotecha 2016)*

Engineering Applications of Artificial Intelligence 54 (2016) 62–79



Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: [www.elsevier.com/locate/engappai](http://www.elsevier.com/locate/engappai)



Yin-Yang-pair Optimization: A novel lightweight optimization algorithm

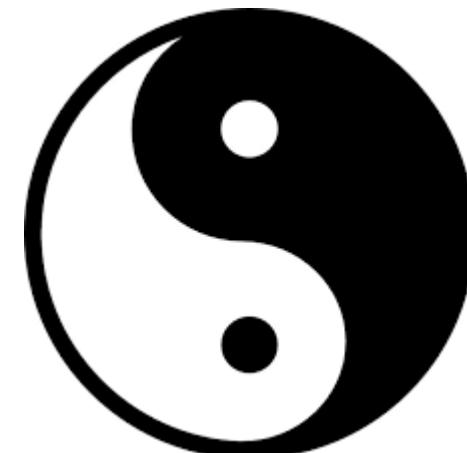


Varun Punnathanam, Prakash Kotecha \*

Department of Chemical Engineering, Indian Institute of Technology Guwahati, Guwahati 781039, Assam, India

*Based on the YinYang philosophy of balance between conflicting forces:*

*Exploration and Exploitation.*



See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/343251814>

# Coronavirus Optimization Algorithm: A Bioinspired Metaheuristic Based on the COVID-19 Propagation Model

Article *in* Big Data · July 2020

DOI: 10.1089/blg.2020.0051

---

CITATIONS

33

READS

658

9 authors, including:



Francisco Martínez-Álvarez

Universidad Pablo de Olavide

139 PUBLICATIONS 2,464 CITATIONS

[SEE PROFILE](#)



Gualberto Asencio Cortés

University of Pablo de Olavide

51 PUBLICATIONS 516 CITATIONS

[SEE PROFILE](#)



José Francisco Torres

Universidad Pablo de Olavide

13 PUBLICATIONS 252 CITATIONS

[SEE PROFILE](#)



David Gutiérrez-Avilés

Universidad Pablo de Olavide

24 PUBLICATIONS 162 CITATIONS

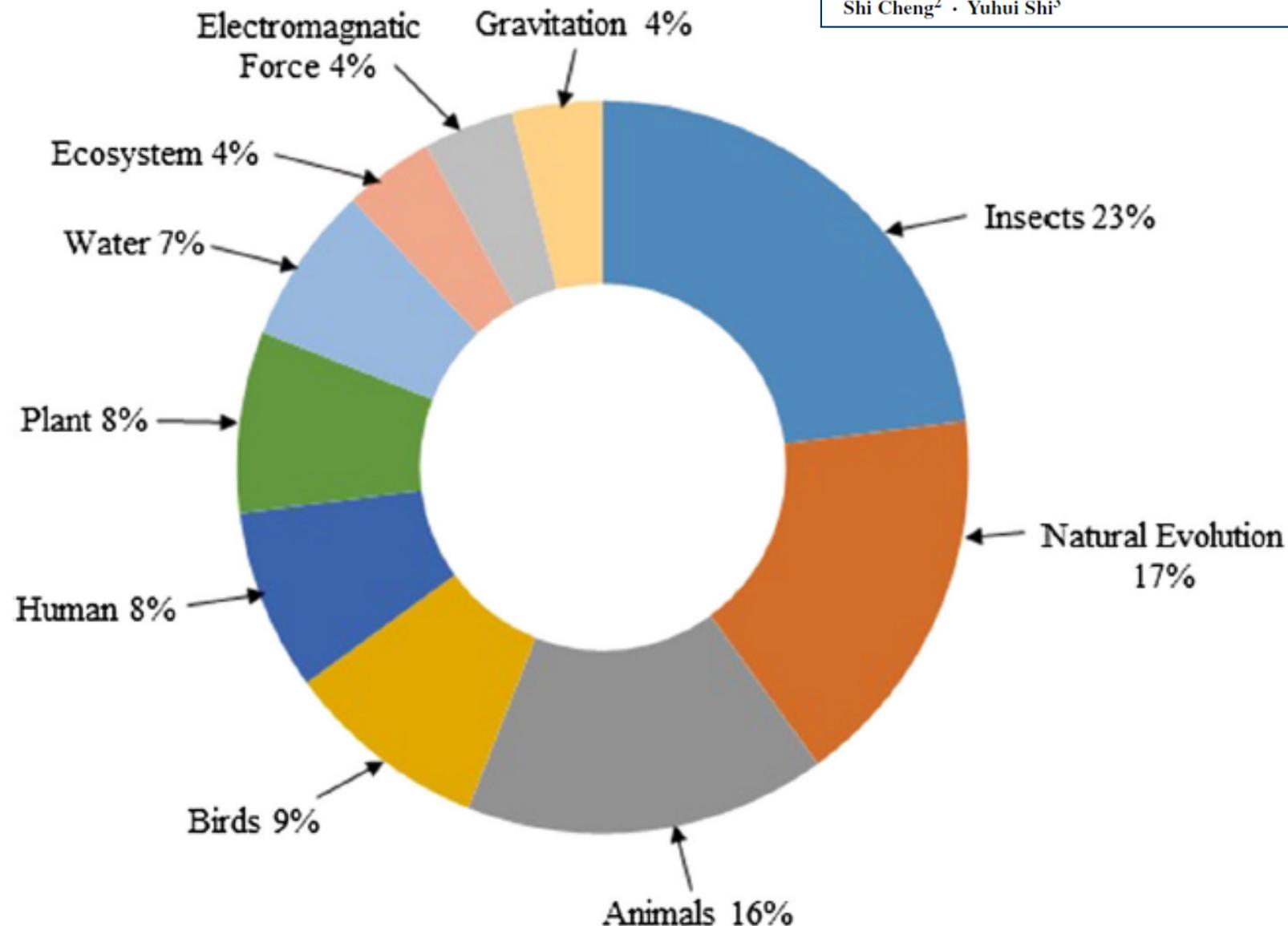
[SEE PROFILE](#)

# En resumen..... Metaheurísticas Bio-Inspiradas



## Metaheuristic research: a comprehensive survey

Kashif Hussain<sup>1</sup> · Mohd Najib Mohd Salleh<sup>1</sup> ·  
Shi Cheng<sup>2</sup> · Yuhui Shi<sup>3</sup>



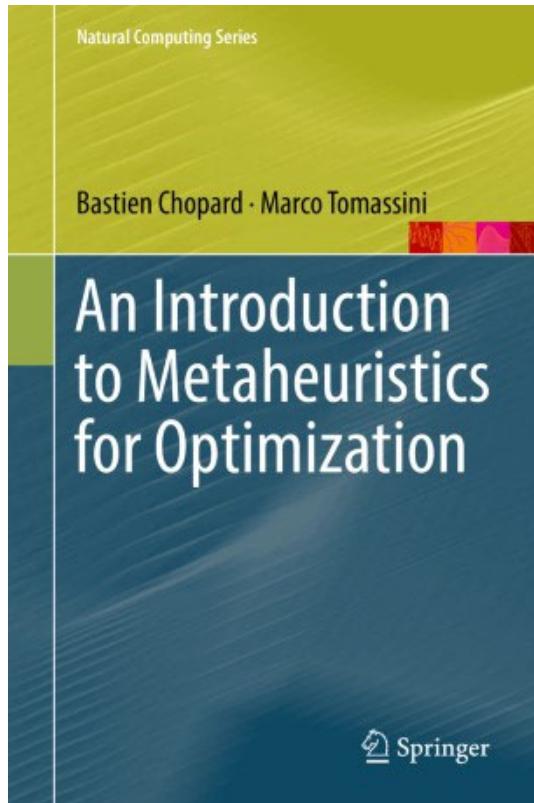
# Variantes Metaheurísticas

Reference	Algorithm name	Reference	Algorithm name
<a href="#">Robbins and Monroe (1951)</a>	Stochastic Approximation Method	<a href="#">Wierstra et al. (2008)</a>	Natural Evolution Strategies
<a href="#">Hooke and Jeeves (1961)</a>	Pattern Search (PS)	<a href="#">Yang (2009)</a>	Firefly Algorithm (FA)
<a href="#">Rastrigin (1963)</a>	Random Search (RS)	<a href="#">Teodorović (2009)</a>	Bee Colony Optimization (BCO)
<a href="#">Matyas (1965)</a>	Random Optimization	<a href="#">He et al. (2009)</a>	Group Search Optimizer (GSO)
<a href="#">Fogel et al. (1966)</a>	Evolutionary strategy (ES)	<a href="#">Yang and Deb (2009)</a>	Cuckoo Search (CS)
<a href="#">Hastings (1970)</a>	Metropolis–Hastings algorithm	<a href="#">Rashedi et al. (2009)</a>	Gravitational Search Algorithm (GSA)
<a href="#">Kernighan and Lin (1970)</a>		<a href="#">Husseinzadeh Kashan (2014)</a>	League Championship Algorithm (LCA)
<a href="#">Holland (1975)</a>	Graph Partitioning Method	<a href="#">Kadioglu and Sellmann (2009)</a>	Dialectic Search
<a href="#">Glover (1977)</a>	Genetic Algorithm (GA)	<a href="#">Yang (2010a)</a>	Bat Algorithm (BA)
<a href="#">Kirkpatrick et al. (1983)</a>	Scatter Search	<a href="#">Lourenço et al. (2010)</a>	Iterated Local Search(ILS)
<a href="#">Glover (1986)</a>	Simulated Annealing (SA)	<a href="#">Shah-Hosseini (2011)</a>	Galaxy-based Search Algorithm (GbSA)
<a href="#">Farmer et al. (1986)</a>	Tabu Search (TS)	<a href="#">Tamura and Yasuda (2011b) ; Tamura and Yasuda (2011a)</a>	
<a href="#">Moscato (1989)</a>	Artificial Immune System (AIS)	<a href="#">Rajabioun (2011)</a>	Spiral Optimization (SO)
<a href="#">Koza (1992)</a>	Memetic Algorithm	<a href="#">Rao et al. (2011)</a>	Cuckoo Optimization Algorithm (COA)
<a href="#">Dorigo et al. (1996)</a>	Genetic Programming (GP)	<a href="#">Alsheddy (2011)</a>	Teaching-Learning-Based Optimization (TLBO)
<a href="#">Fonseca and Fleming (1993)</a>	Ant Colony Optimization (ACO)	<a href="#">Gandomi and Alavi (2012)</a>	Guided Local Search (GLS)
<a href="#">Battiti and Brunato (2010)</a>	Multi-Objective GA (MOGA)	<a href="#">Hajiaghaei-Keshteli and Aminnayeri (2014)</a>	Krill Herd (KH) Algorithm
<a href="#">Srinivas and Deb (1994)</a>		<a href="#">Civicioglu (2012)</a>	
<a href="#">Eberhart and Kennedy (1995)</a>	Reactive Search Optimization (RSO)	<a href="#">Alatas (2012)</a>	Keshtel Algorithm (KA)
<a href="#">Hansen and Ostermeier (2001)</a>	NSGA for Multi-Objective Optimization	<a href="#">Husseinzadeh Kashan (2013)</a>	Differential Search Algorithm (DSA)
<a href="#">Storn and Price (1997)</a>	Particle Swarm Optimization (PSO)	<a href="#">Husseinzadeh Kashan et al. (2015)</a>	Artificial Chemical Reaction Optimization Algorithm (ACROA)
<a href="#">Rubinstein (1997)</a>	CMA-ES	<a href="#">Sadollah et al. (2013)</a>	Optics Inspired Optimization(OIO)
<a href="#">Hansen et al. (1997)</a>	Differential Evolution (DE)	<a href="#">Hatamlou (2013)</a>	Grouping Evolution Strategies (GES)
<a href="#">Taillard and Voss (1999)</a>	Cross Entropy Method (CEM)	<a href="#">Civicioglu (2013a)</a>	Mine Blast Algorithm (MBA)
<a href="#">Geem et al. (2001)</a>	Variable Neighborhood search(VNS)	<a href="#">Kaveh and Mahdavi (2015)</a>	Black Hole (BH)
<a href="#">Hanseth and Aanestad (2001)</a>	Partial Optimization Metaheuristic Under Special Intensification Conditions (POPMUSIC)	<a href="#">Gandomi (2014)</a>	Artificial Cooperative Search Algorithm(ACS)
<a href="#">Larrañaga and Lozano (2002)</a>	Harmony Search (HS)	<a href="#">Salimi (2014)</a>	Colliding Bodies Optimization (CBO)
<a href="#">Deb et al. (2002)</a>	Bootstrap Algorithm (BA)	<a href="#">Cheng and Prayogo (2014)</a>	Interior Search Algorithm (ISA)
<a href="#">Liu and Passino (2002)</a>	Estimation of Distribution Algorithms (EDA)	<a href="#">Zheng (2015)</a>	Stochastic Fractal Search (SFS)
<a href="#">Nakrani and Tovey (2004)</a>	NSGA-II for Multi-Objective Optimization	<a href="#">Dogan and Ölmez (2015)</a>	Symbiotic Organisms Search (SOS)
<a href="#">Krishnanand and Ghose (2005) ; Krishnanand and Ghose (2006)</a>	Bacterial Foraging behavior Optimization (BFO)	<a href="#">Wang et al. (2015)</a>	Water Wave Optimization (WWO)
<a href="#">Basturk and Karaboga (2006)</a>	Bees Optimization	<a href="#">Baykasoglu and Akpinar (2017)</a>	Vortex Search Algorithm (VSA)
<a href="#">Pham et al. (2005)</a>	Glowworm Swarm Optimization(GSO)	<a href="#">Mirjalili (2016b)</a>	Elephant Herding Optimization (EHO)
<a href="#">Haddad et al. (2006)</a>	Artificial Bee Colony Algorithm (ABC)	<a href="#">Liang et al. (2016)</a>	Weighted Superposition Attraction (WSA)
<a href="#">Shah-Hosseini (2009)</a>	Bees Algorithms (BA)	<a href="#">Mirjalili (2016a)</a>	Dragonfly algorithm
<a href="#">Atashpaz-Gargari and Lucas (2007)</a>	Honey-bee Mating Optimization (HMO)	<a href="#">Ebrahimi and Khamehchi (2016)</a>	Virus Optimization Algorithm
<a href="#">Mucherino and Seref (2007)</a>	Intelligent Water Drops (IWD)	<a href="#">Mirjalili et al. (2017)</a>	Sine Cosine Algorithm (SCA)
	Imperialist Competitive Algorithm (ICA)		Sperm Whale Algorithm (SWA)
	Monkey Search (MS)		Salp Swarm Algorithm

## Lecturas Recomendadas



Artif Intell Rev (2019) 52:2191–2233  
<https://doi.org/10.1007/s10462-017-9605-z>



### Metaheuristic research: a comprehensive survey

Kashif Hussain<sup>1</sup> · Mohd Najib Mohd Salleh<sup>1</sup> · Shi Cheng<sup>2</sup> · Yuhui Shi<sup>3</sup>

Artificial Intelligence Review (2020) 53:501–593  
<https://doi.org/10.1007/s10462-018-9667-6>



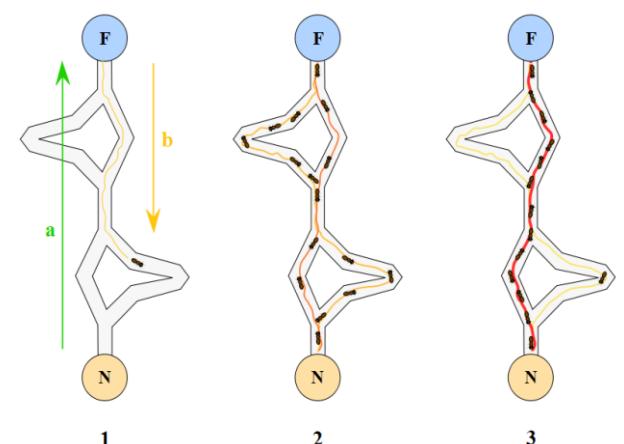
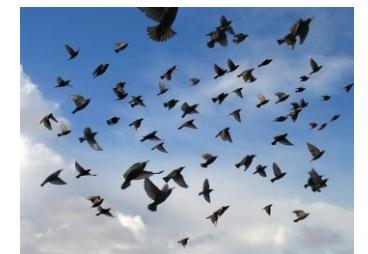
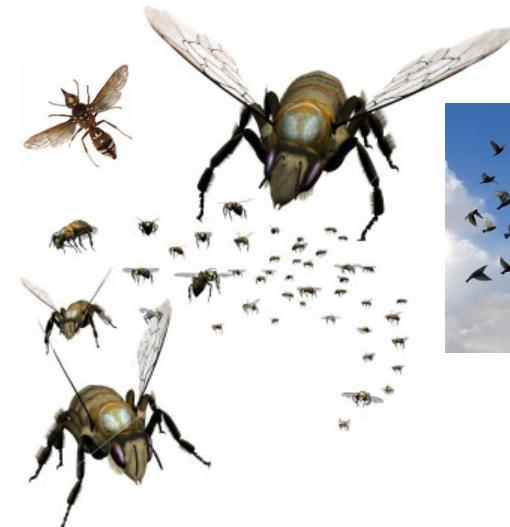
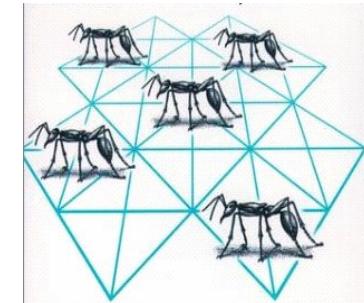
### A state of the art review of intelligent scheduling

Mohammad Hossein Fazel Zarandi<sup>1</sup> · Ali Akbar Sadat Asl<sup>1</sup> · Shahabeddin Sotudian<sup>2</sup> · Oscar Castillo<sup>3</sup>

Published online: 19 November 2018  
© Springer Nature B.V. 2018

# Inteligencia de Enjambre: Conclusiones

- Sistemas bio-inspirados
- Basados en el comportamiento colectivo de sistemas descentralizados y auto-organizados (hormigas, abeja, pájaros, peces, etc.)
- Las partículas representan soluciones o procesos de búsqueda, que inciden en:
  - Exploración: Generación de soluciones nuevas
  - Explotación: Mejora de la solución en su vecindad
- La partículas interaccionan entre si compartiendo soluciones encontradas (para que otras partículas utilicen/mejoren las buenas soluciones. La interacción se basa en reglas muy simples.
- La interacción entre las partículas permite un comportamiento global de la búsqueda más inteligente y adaptativo que la suma de comportamientos individuales.



## Path-finding (Obtención de sendas en laberintos)

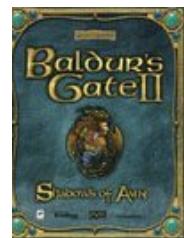
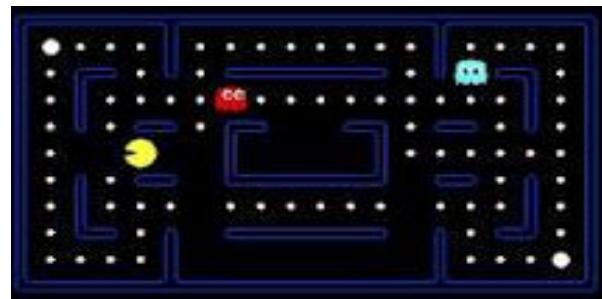
**Objetivo:** encontrar el mejor camino (habitualmente, el más corto) entre dos puntos a través de un mapa con obstáculos. Problemas de laberinto, *Aplicaciones para videojuegos*, etc.

**Algoritmos típicos:** Dijkstra (muy costoso en laberintos complejos),

Algoritmos A (equivalente a Dijkstra, con  $h(n)=0$ ),

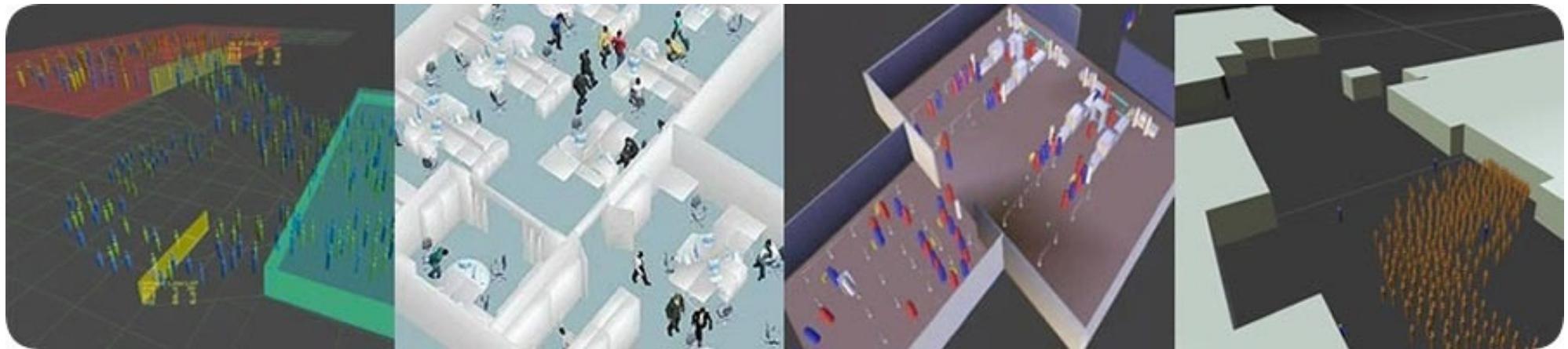
A\* sobre-informados, etc.

**Metaheurísticas:** Grasp, **inteligencia de enjambre (algoritmo de las hormigas,)**, ...



<http://www.movingai.com/benchmarks/>  
Benchmarks for Grid-Based Path finding.  
IEEE Trans. on Comp. Int. and AI in games (2012)

# APLICACIÓN: Inteligencia de Enjambre: Simulación de comportamientos sociales

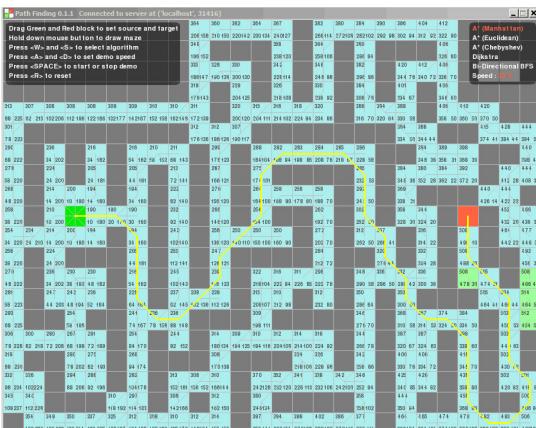


<http://massivesoftware.com>

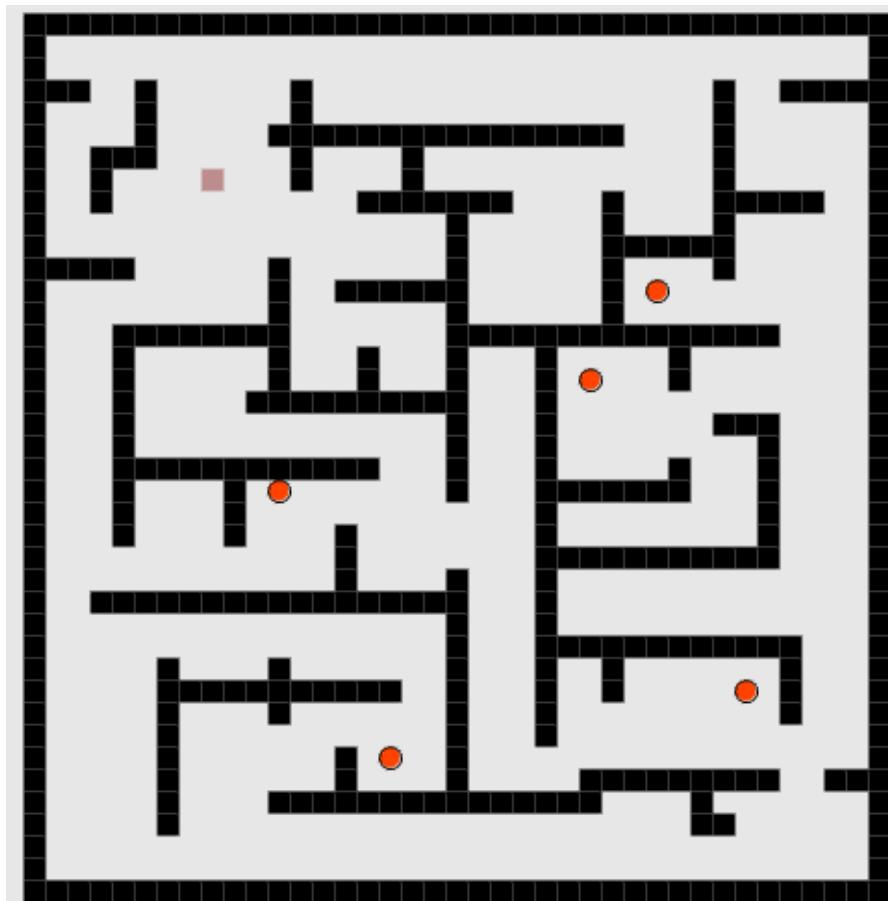


# Path-finding Problem (Obtención de sendas en laberintos)

## Algoritmos Heurísticos A



Laberinto Simple  
(Alg. Hormigas)



Laberinto con  
Múltiples Objetivos

Simulación de colectivos coordinados  
(movimiento de masas)



*Flocking: Modelado de movimientos de grupos, vida artificial.*

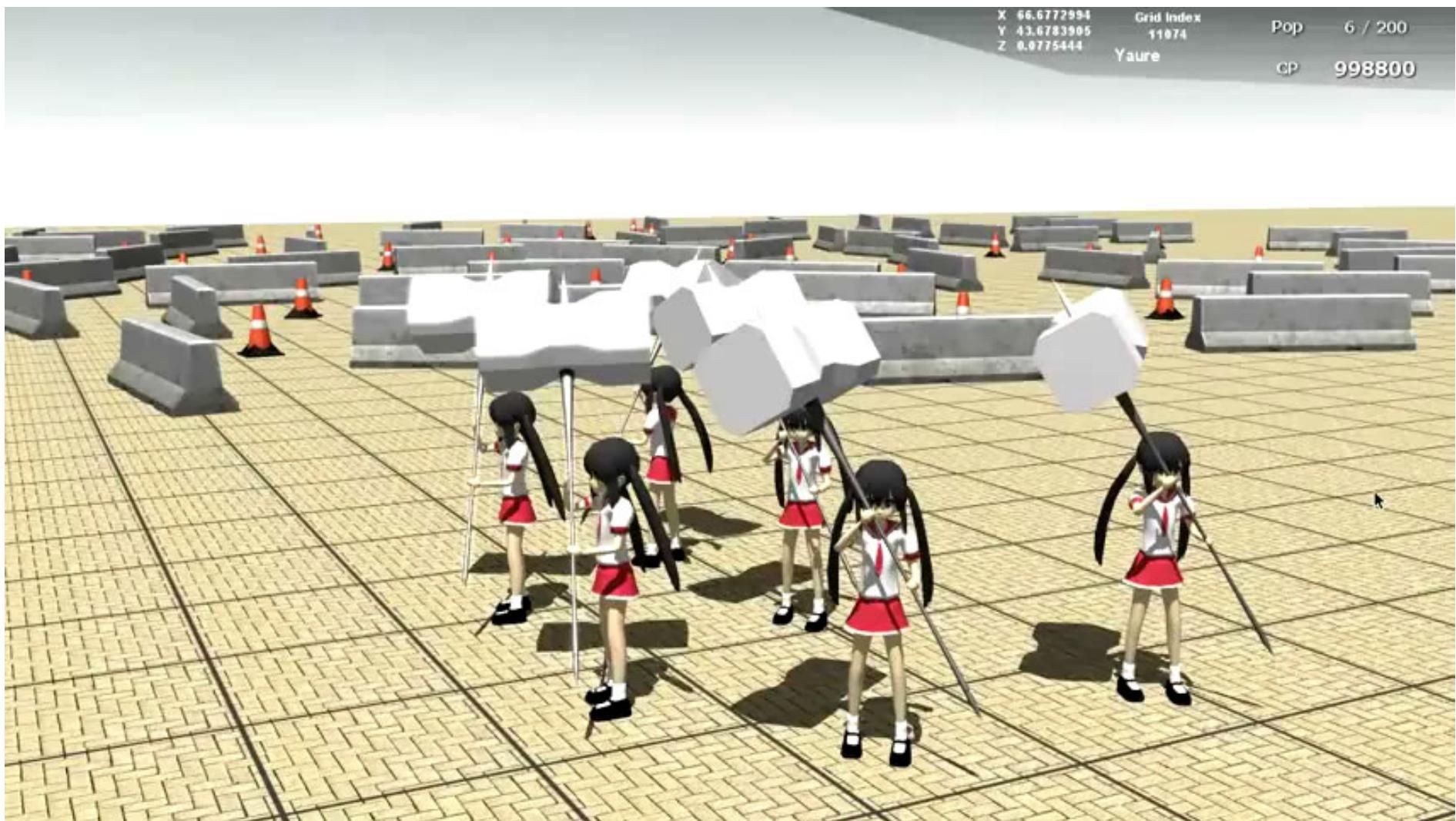
Las partículas se desplazan en una dirección global, persiguiendo una meta y evitando colisiones:

- Cada partículas mantiene una distancia con sus vecinos (*separación*).
- Cada partículas sigue la dirección media del movimiento de sus vecinas (*alineación*).
- Cada partículas mantiene una cohesión de grupo con sus vecinas (*cohesión*).
- Puede, o no, haber un líder en el grupo



*Aplicaciones: Collective animal behaviour, movimiento de drones, juegos sociales, simulación en películas, etc.*

## Movimiento de individuos (*swarm behaviour*)



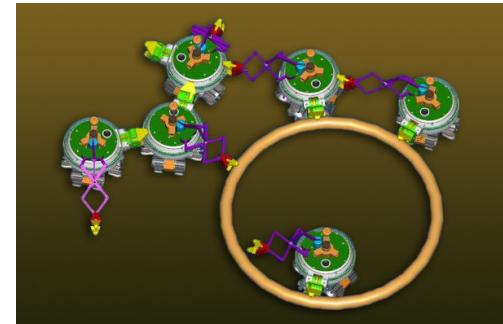
## Swarm Robotics

Enfoque inspirado en el comportamiento colectivo de los animales sociales y que se centra en la interacción de múltiples robots.

Hace hincapié en el control descentralizado, comunicación limitada entre agentes, la información local, el seguimiento de un comportamiento global y la robustez del grupo.

Los sistemas de enjambre robóticos difieren de otros sistemas multi-robot en:

- Son robots autónomos ubicados en un determinado medio ambiente,
- El enjambre tiene un gran número de robots, posiblemente agrupados en pequeños grupos de robots homogéneos,
- Los robots son relativamente simples, tienen sensores locales y sus habilidades de comunicación son limitadas.
- La coordinación entre los robots se distribuye tal que el sistema es tolerante a fallos, debido a la redundancia de robots cada uno de los agentes en el sistema no es esencial y puede ser sustituido por otro agente.
- Son fácilmente escalable, permitiendo que más agentes para ser agregados o eliminados de acuerdo a las demandas de la tarea



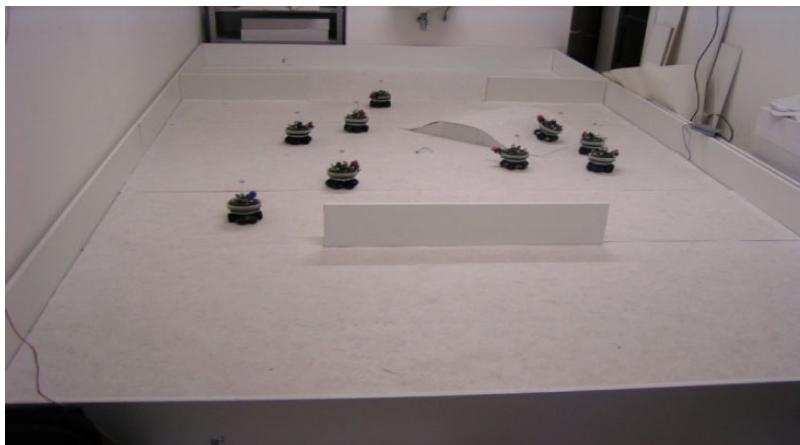
Científicos de Harvard crean  
un enjambre cooperativo de  
1.000 robots

## Objetivo: Aplicar técnicas de Inteligencia Colectiva.

*The main objective of the **Swarm-bots** is the design and implementation of **self-organising** and **self-assembling artefacts**. This novel approach finds its theoretical roots in recent studies in **swarm intelligence**: self-organising and self-assembling capabilities shown by social insects and other animal societies.*

*The main tangible objective is the construction of at least one of such artefact (swarm-bot). That is, an artifact composed of a number of simpler, insect-like, robots(s-bots), built out of relatively cheap components, capable of self-assembling and self-organising to adapt to its environment.*

Simulación de Escenarios  
(Lausanne Bruselas)



Swarm-Bots



Hardware: <https://www.k-team.com/mobile-robotics-products/kilobot>

## Ejemplo de resolución

(encontrar senda optima y transportar objeto)



## Dificultades y Colaboración



**Algoritmo de las Hormigas**

**Búsqueda en Haz**

**Enfriamiento Simulado**

**BÚSQUEDA DISPERSA**

**Algoritmo A\***

**Algoritmo de las Abejas**

**GRASP**

**Algoritmos Genéticos**

**Enjambre de Particulas**

**Algoritmos Meméticos**

**Búsqueda Tabú**

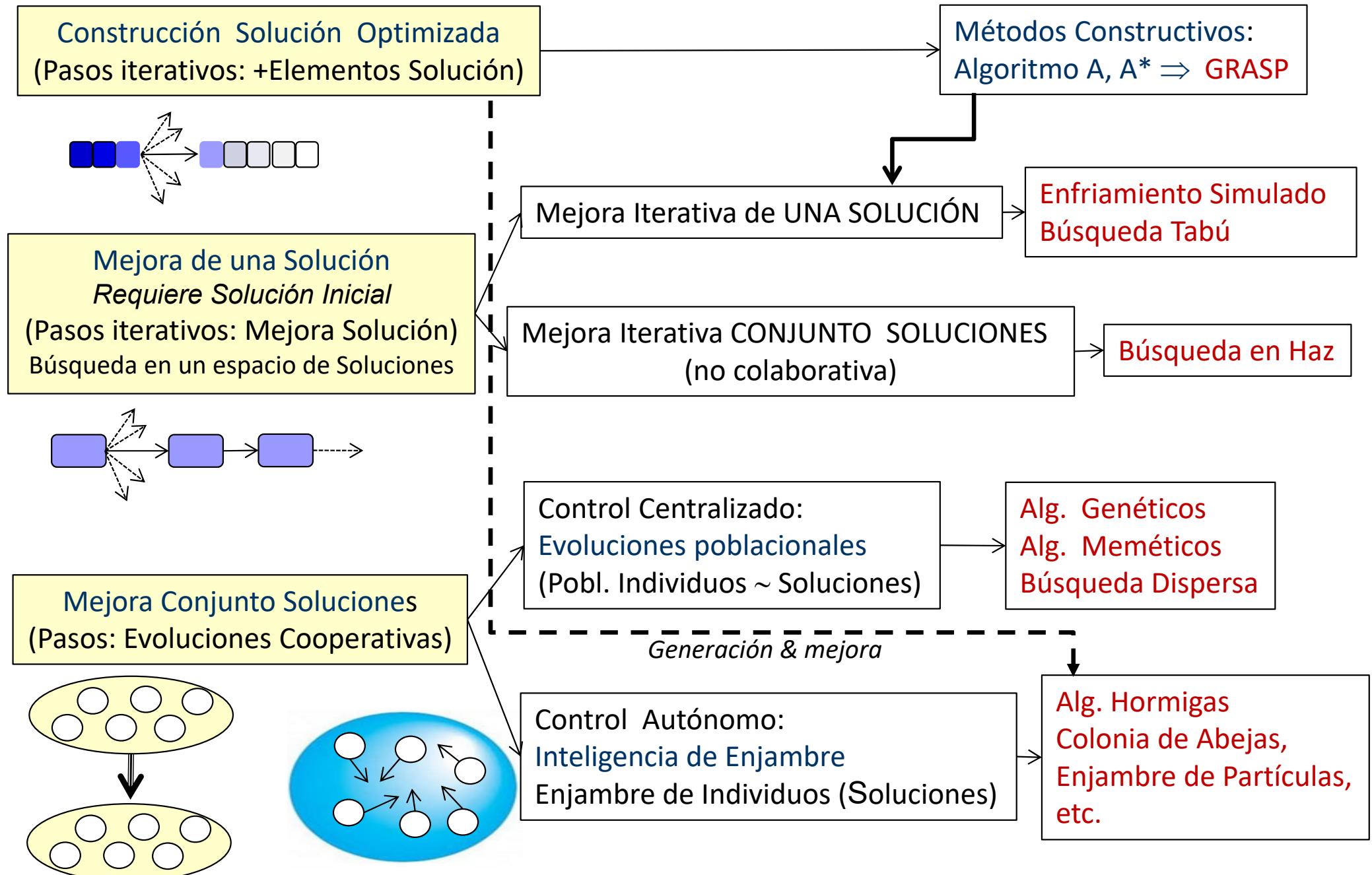
**Heurística h1(n)**

**Heurística h2(n)**

**Heurística h3(n)**

**Cual es mejor?**

# Métodos Metaheurísticos. s.



# No Free Lunch Theorems for Optimization

David H. Wolpert and William G. Macready



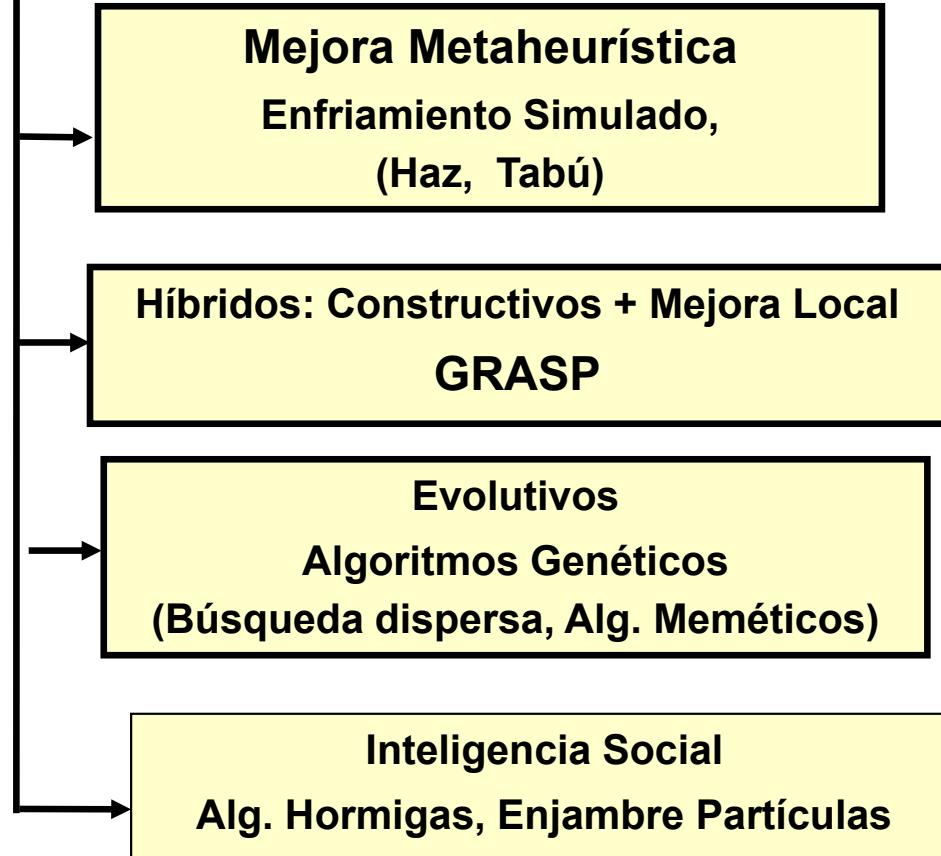
## 'No free lunch' Theorem

(*No free lunch theorems for optimization*, Wolpert, D.H.; Macready, W.G.; *IEEE Transactions on Evolutionary Computation*, 1, 1997)

*"The computational cost of finding a solution,  
averaged over all problems in the class, is the same for any solution method."*

*"For any algorithm, any elevated performance over one class of problems  
is exactly paid for in performance over another class."*

# PROBLEMA



IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 1, NO. 1, APRIL 1997

## No Free Lunch Theorems for Optimization

David H. Wolpert and William G. Macready



Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: [www.elsevier.com/locate/ins](http://www.elsevier.com/locate/ins)



A survey on optimization metaheuristics

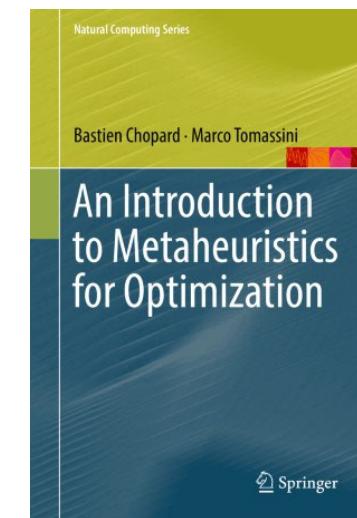
Ilhem Boussaïd <sup>a</sup>, Julien Lepagnot <sup>b</sup>, Patrick Siarry <sup>b,\*</sup>

<sup>a</sup>Université des sciences et de la technologie Houari Boumediene, Electrical Engineering and Computer Science Department, El-Alia BP 32 Bab-Ezzouar, 16111 Algiers, Algeria

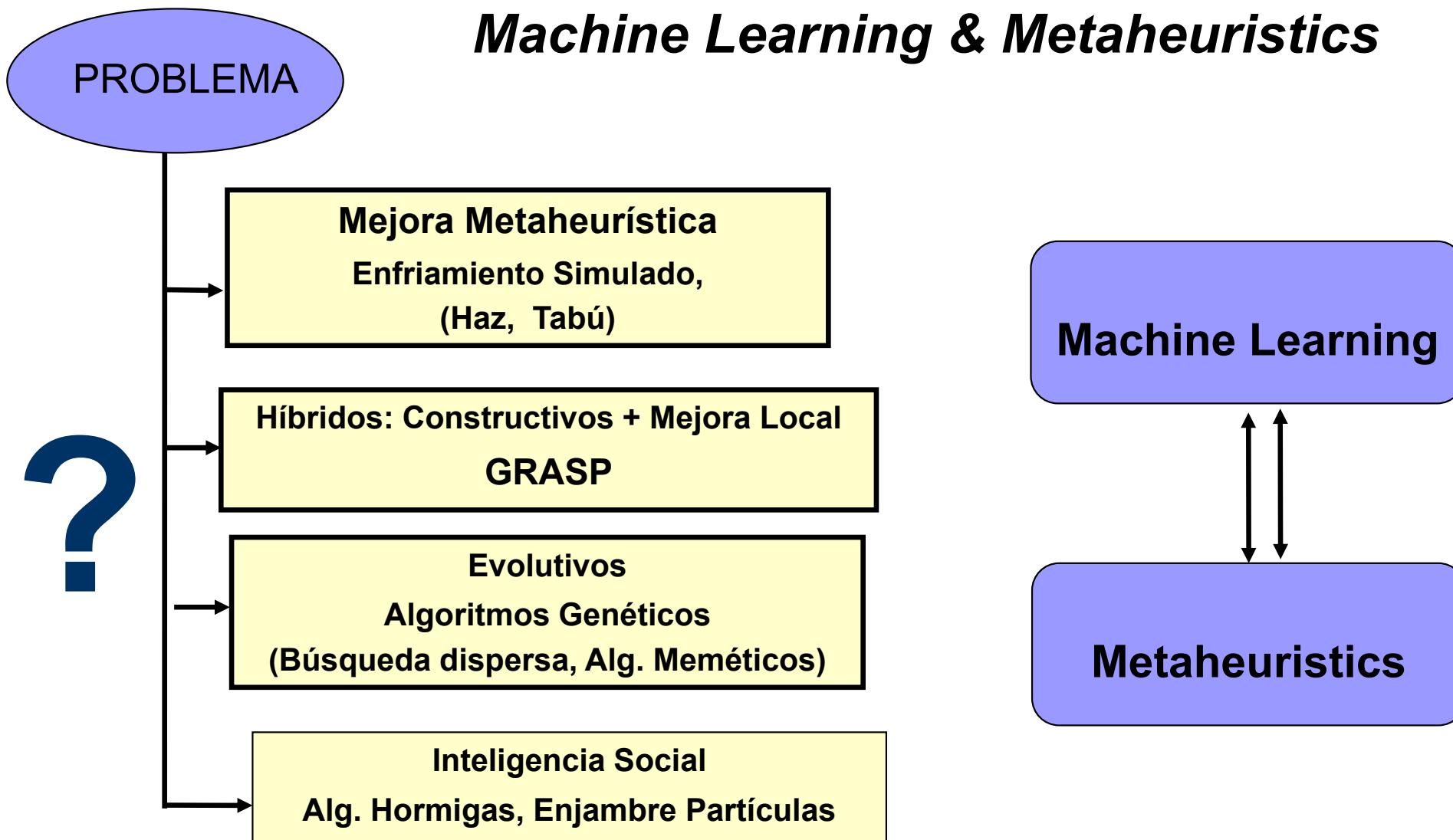
<sup>b</sup>Université Paris Est Créteil, LISSI, 61 avenue du Général de Gaulle 94010 Créteil, France

Artif Intell Rev (2019) 52:2191–2233  
<https://doi.org/10.1007/s10462-017-9605-z>

Metaheuristic research: a comprehensive survey



# *Machine Learning & Metaheuristics*

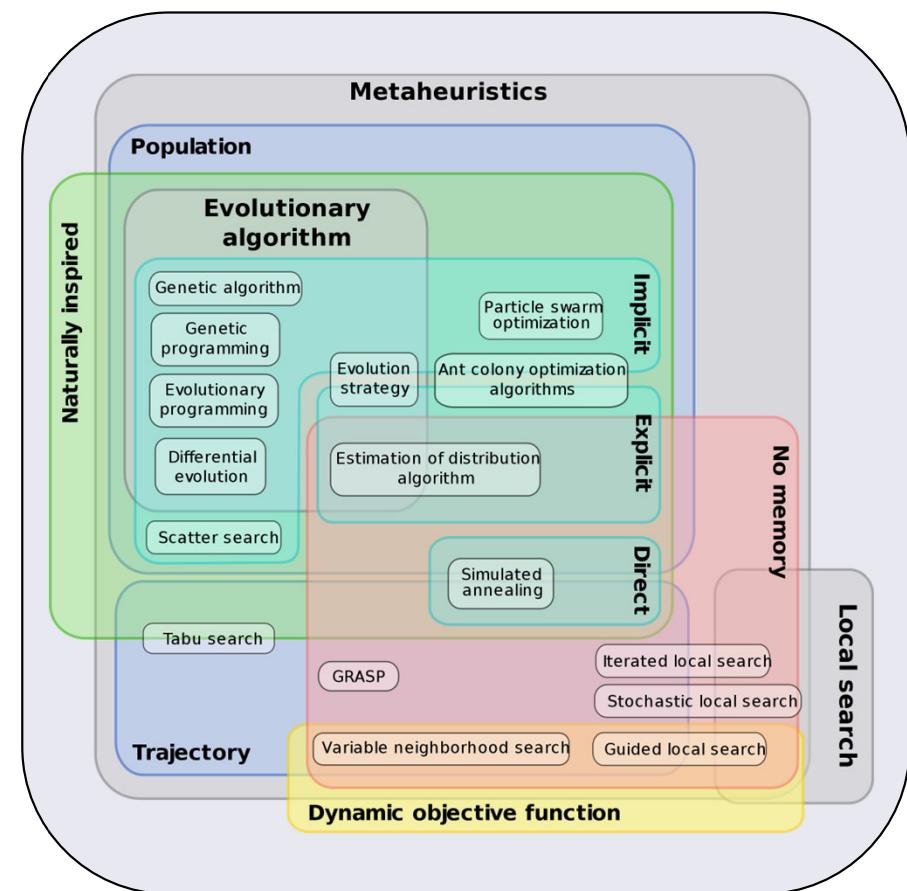
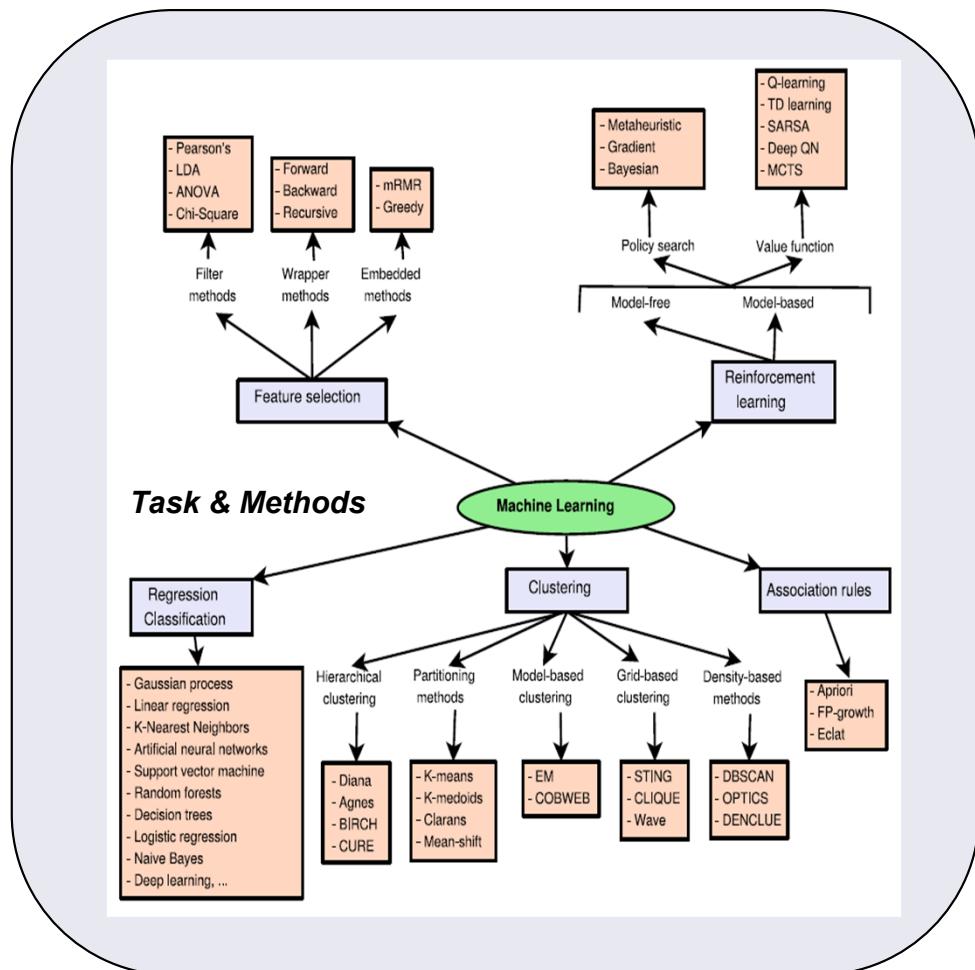


# Aplicación “Machine Learning” a “Metaheurísticas”

**Machine Learning into Metaheuristics: A Survey and Taxonomy,**  
ACM Computing Surveys. Vol. 54, Iss. 6 (2021)



## Aplicación del aprendizaje automático (ML) para diseñar metaheurísticas más eficientes y efectivas.



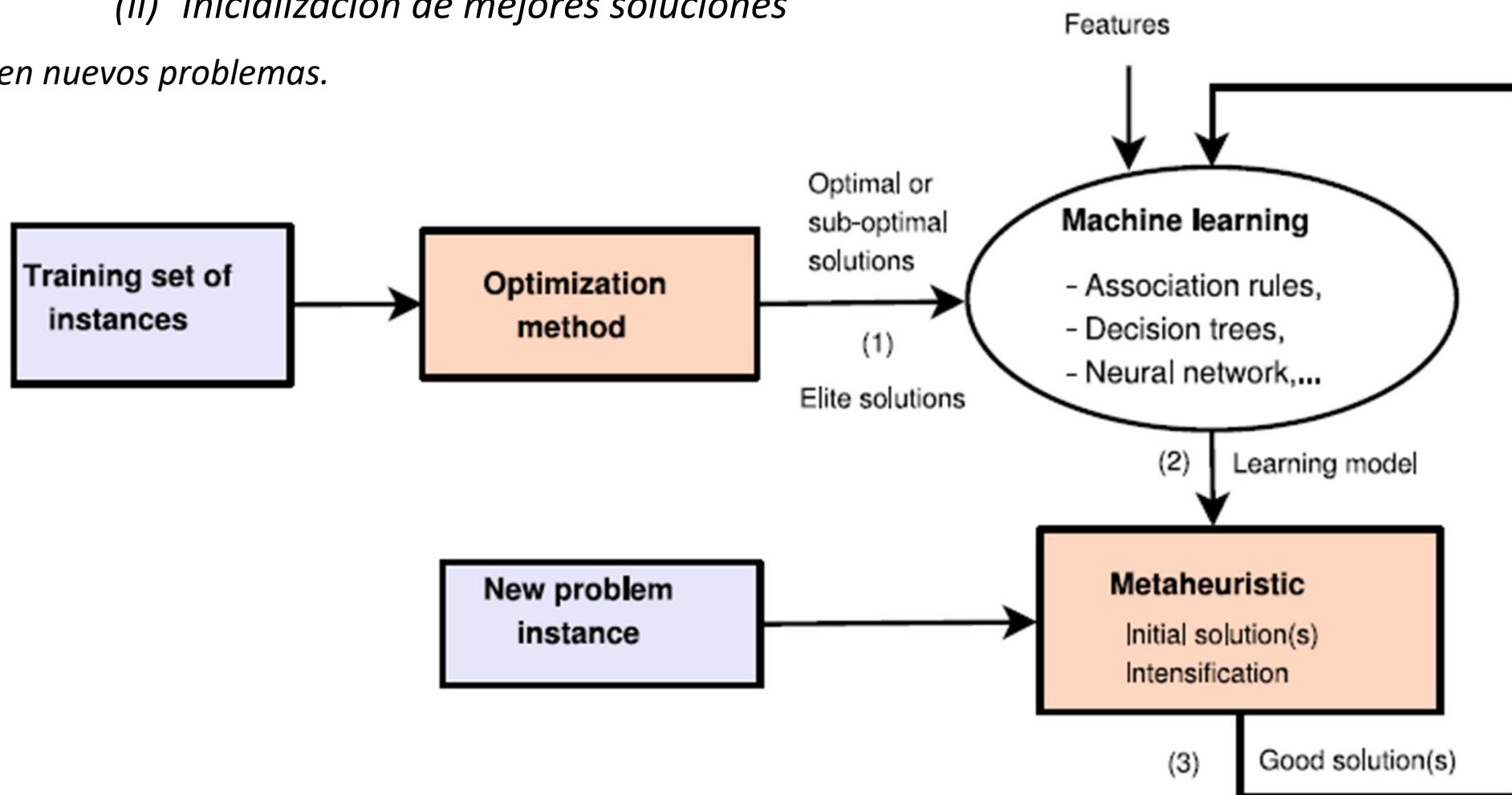
## Ayuda para Soluciones Iniciales

Aprendizaje de características/patrones/propiedades de buenas soluciones en un dominio de problemas,

para ayudar a la metaheurística a:

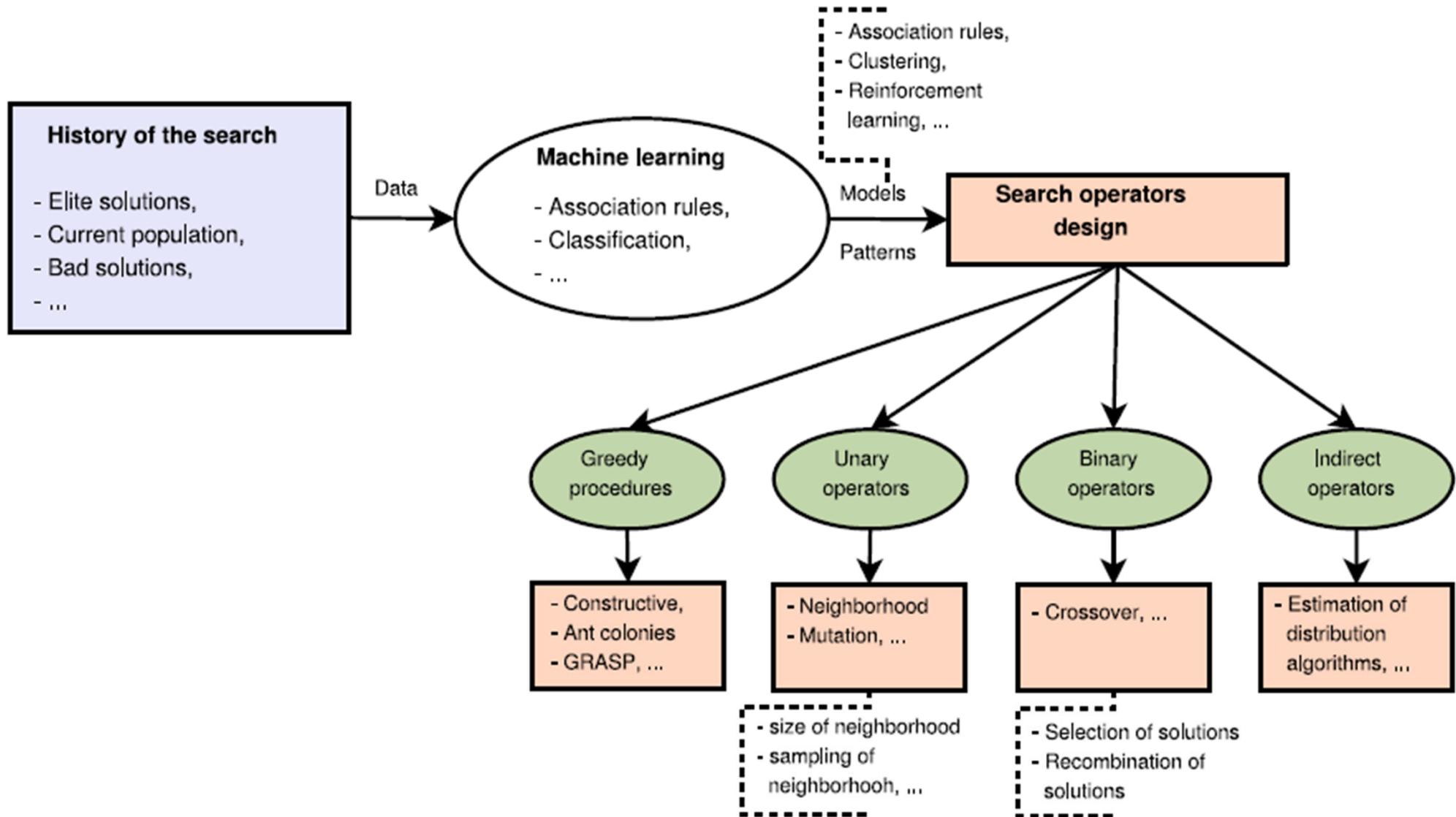
- (i) buscar en regiones prometedoras del espacio de búsqueda, o
- (ii) Inicialización de mejores soluciones

en nuevos problemas.

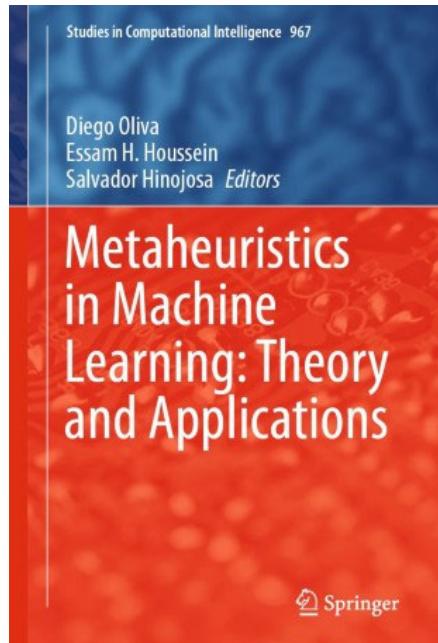
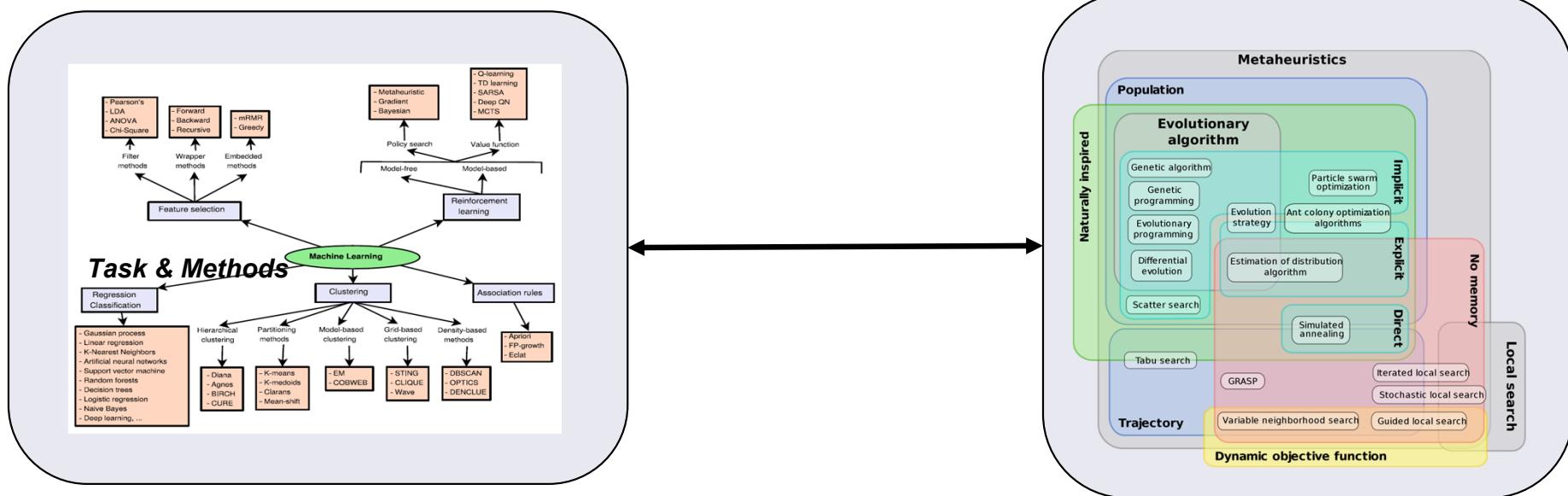


## Ayuda para Parameter Tuning

Extracción de conocimiento de búsquedas previas para la obtención de buenos operadores en nuevos problemas: ***parametrización de la metaheurística dependiente del problema (tamaño, tipología, etc.)***



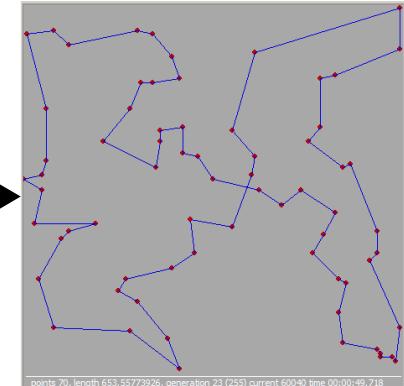
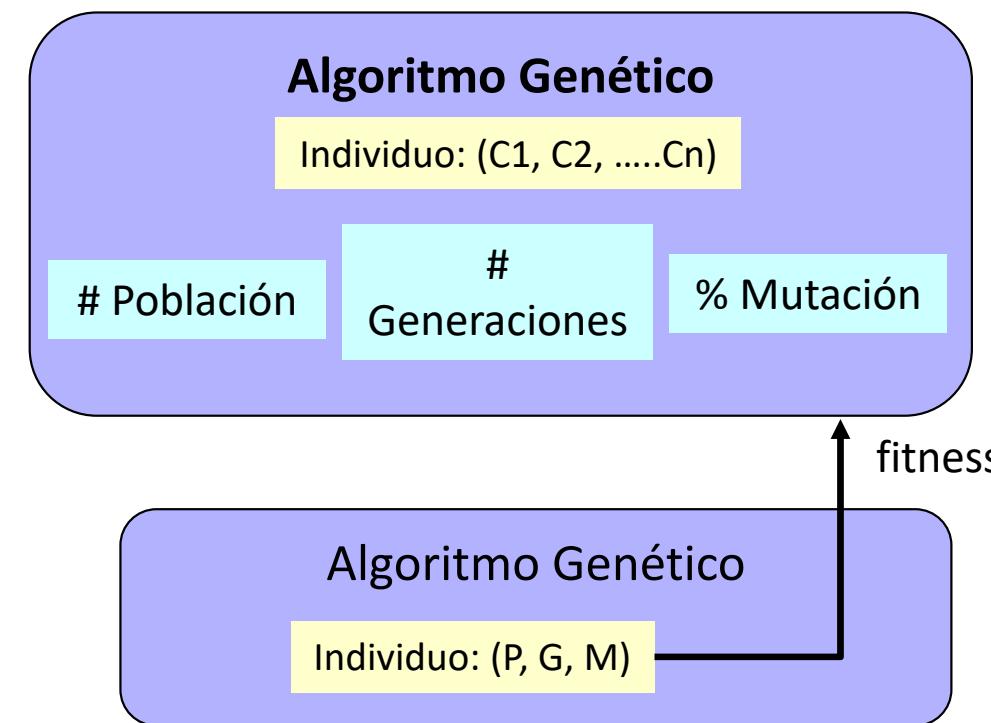
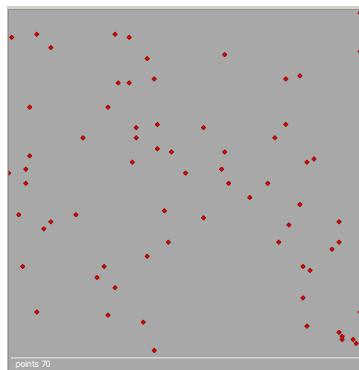
# Aplicación de Metaheurísticas a Machine Learning



- Ajuste de la arquitectura del sistema de aprendizaje.
- Mejora del entrenamiento, solidez y velocidad.
- Ajuste de parámetros, coeficientes RNA,

# Aplicación de Metaheurísticas a Metaheurísticas

- Un AG es un proceso estocástico (no garantía obtención de buenas soluciones).
- Ante un tipo de problemas, hay que establecer sus parámetros, en función de su tipología, tamaño, etc.
- Esto se hace mediante un análisis razonado del problema, esperando encontrar los parámetros que darán buenas soluciones (sin garantía).



**Objetivo:**  
**Mejor (P, G, M)**

*Se puede plantear un proceso de optimización metaheurístico de los parámetros de un AG, que obtenga los mejores parámetros ante similar clase de problemas: AG sobre AG*

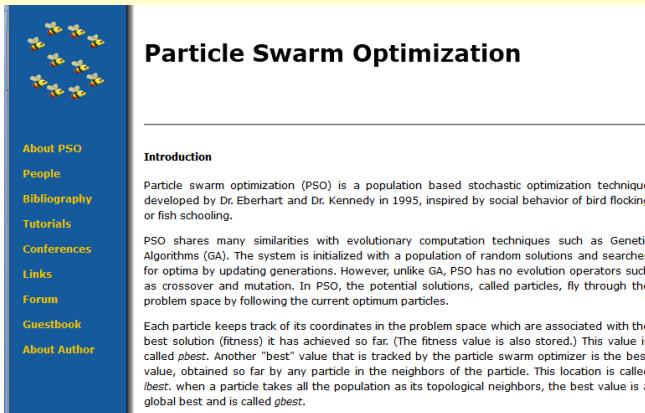
# Tema- 5 Inteligencia de Enjambre (Swarm intelligence)

(Inteligencia Social, Inteligencia Colectiva, Metaheurísticas Bio-Inspiradas)

- **Algoritmo de las Hormigas (Ant Colony Optimization - ACO),**
- **Enjambre de partículas (Particle Swarm Optimization - PSO)**
- **Otras variantes**



<http://www.swarmintelligence.org/>



The screenshot shows the homepage of the Particle Swarm Optimization website. It features a sidebar with links to About PSO, People, Bibliography, Tutorials, Conferences, Links, Forum, Guestbook, and About Author. The main content area has a title "Particle Swarm Optimization" and a section titled "Introduction". The introduction text explains that PSO is a population-based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by the social behavior of bird flocking or fish schooling. It describes how PSO shares similarities with Genetic Algorithms (GA) but lacks evolution operators like crossover and mutation. Particles in PSO move through the problem space to find the best solution.

