



UNA INTRODUCCIÓN A LOS ALGORITMOS DE COMPUTACIÓN EVOLUTIVA

Sancho Salcedo Sanz

Departamento de Teoría de la Señal y Comunicaciones

Universidad de Alcalá



- **Parte I: Computación evolutiva**
- **Teoría:**
 - Introducción a la Metaheurística.
 - Metaheurística basada en Poblaciones
 - Estrategias Evolutivas y Harmony Search.
 - Algoritmos Evolutivos y genéticos.
 - Algoritmos de Particle Swarm.
 - Metaheurísticas constructivas
 - Optimización basada en Colonias de Hormigas
 - Metaheurísticas basadas en trayectorias
 - Algoritmos de Temple Simulado.
 - Metaheurísticas híbridas: CRO

INTRODUCCIÓN A LA METAHEURÍSTICA



- Son algoritmos de propósito general que consisten en procedimientos **iterativos** que guían una heurística subordinada de forma inteligente distintos conceptos para explotar y explorar de manera adecuada el **espacio de búsqueda**
- Combina el prefijo **Meta** mas allá, superior , con la palabra griega **εὕρισκειν, heuriskein**, que significa encontrar

- Algoritmos de propósito general.
- Gran éxito en la práctica
- Fácilmente implementables
- Fácilmente paralelizables



- Algoritmos aproximados.
- Poca base teórica.
- Son muy probabilísticos.



- Es fácil entender que para *encontrar* hay que buscar adecuadamente



- La búsqueda debe combinar adecuadamente dos estrategias

- **Exploration.** Explorar adecuadamente el espacio de búsqueda para evitar caer en máximos locales



- **Exploitation.** Explotar adecuadamente el espacio de búsqueda para evitar esfuerzos inútiles.



INTRODUCCIÓN A LA METAHEURÍSTICA

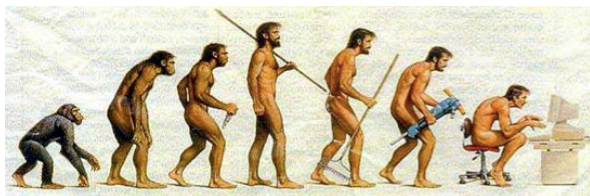
- Podemos intentar una posible clasificación
 - **Heurísticas basadas en poblaciones.** Se consideran conjuntos de individuos (población) donde cada uno representa una solución al problema y se hace evolucionar el conjunto. **Algoritmos genéticos, Particle Swarm**
 - **Heurísticas Constructivas.** Se parte de una solución inicial, se van añadiendo componentes hasta obtener una solución final viable lo más óptima posible. **Ant Colony Optimization.**
 - **Heurísticas basadas en trayectorias.** Parten de una solución inicial e iterativamente tratan de reemplazarla por un vecino mejor. **Simulated Annealing, Búsqueda Local**
 - **Heurísticas híbridas.** Mezclan estrategias anteriores para conseguir procedimientos de búsqueda nuevos y potentes. **Coral Reefs Optimization algorithm.**

INTRODUCCIÓN A LA METAHEURÍSTICA



- Paralelamente definimos el concepto de Algoritmos bioinspirados.
- Son un conjunto de procedimientos que tratan de imitar procesos de la naturaleza, sociales o de los animales para la resolución de problemas de búsqueda, aprendizaje o comportamiento.
- Ejemplos: evolución, temple, colonias de animales, arrecifes de coral, etc.

Algoritmos Evolutivos



Simulated Annealing

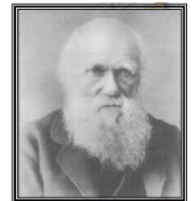


Colonia de Hormigas



COMPUTACIÓN EVOLUTIVA (LÍNEAS GENERALES)

- Basada en la Teoría de la Evolución de Darwing
- Pero en temas relacionados con “Computer Science”
 - I. Rechenberg, 1960, Estrategias de Evolución
 - Lawrence Fogel, 1960 , Algoritmos de Evolución
 - Algoritmos Genéticos(Gas)
 - John Holland, 1975, *"Adaptationin Natural and Artificial Systems"*.
 - *David Goldberg(1989) GeneticAlgorithms"*.
 - Programación Genética John Koza, 1992.





- Son algoritmos bio-inspirados, por lo que hacen uso de los mecanismos de la naturaleza:
 - Los individuos más aptos de una población son los que sobreviven.
 - Los cambios se traducen en transformaciones en los genes de un individuo.
 - Las características más destacadas de un individuo se deben transmitir a sus descendientes.



- En general en la computación evolutiva tenemos:
 - **Población:** Individuos que representan cada uno de ellos una solución codificada de manera adecuada
 - **Operadores:** Conjunto de operaciones a realizar sobre los individuos de una población
 - Selección
 - Cruce
 - Mutación.
 - **Función de Fitness.** Valor que se emplea para medir la adaptación del individuo al medio. En concreto mide lo bueno o malo que es la solución concreta al problema.
- El algoritmo realiza un proceso iterativo donde los individuos mejores tienen más probabilidades de continuar en las siguientes generaciones
- “Al final” el individuo con mejor fitness es la solución del problema.

- En muchas ocasiones uno de los primeros y más graves problemas consiste en la codificación de la solución.
- Preguntas típicas
 - ¿Que significa cada individuo?.
 - ¿Cual es la longitud de cada individuo?
 - Codificación entera, real o binaria
- Ejemplo
 - Codificación entera: Asignación de N=15 terminales a M=4 grupos.

1	4	1	3	2	2	4	4	1	1	3	3	3	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Nodo 



- Es, básicamente, lo que queremos optimizar.
 - En redes de comunicación: Minimización de coste total de la red, maximización del uso de los sistemas
 - En sistemas industriales: Optimización de la asignación de recursos a tareas.
 - En logística: Minimización de los tiempos de transporte de materiales con restricciones en combustible
 - En energía. Maximización de la potencia entregada por aerogenerador por unidad de coste.
- Debemos tener cuidado con las restricciones (constraints)
 - Solución no valida
 - Penalización

PRIMERA METAHEURÍSTICA BIOINSPIRADA:

ESTRATEGIAS EVOLUTIVAS

- Las estrategias evolutivas fueron desarrolladas en 1964 por un grupo de estudiantes de ingeniería alemanes.

- Ingo Rechenberg
 - Hans Paul Schwefel



- Querían resolver problemas hidrodinámicos complejos

■ 1º Versión

- (1+1)-EE. 1 Padre, 1 Hijo, sobrevive el mejor (estrategia extintiva).
- El hijo es generado de la siguiente forma

$$x^{t+1} = x^t + N(0, \sigma)$$

- Donde “t” es la iteración, y $N(0, \sigma)$ es una Gaussiana de media cero y varianza σ
- Debemos tener en cuenta que puede haber muchos “x” en el cromosoma



- En 1973 Rechenberg introduce el concepto de población.
- Propone una estrategia $(u + 1)$ -EE. Significa que hay u padres, se genera un único hijo, y este sustituye al peor de los padres. (Selección extintiva)
- Procedimiento de Mutación
 - Se modifica la varianza de la Normal($0, \sigma$)
 - ¿Cuanto? ¿cuándo?
- ¿cuándo?. Se debe hacer cada k iteraciones (a fijar en el problema)
 - Intensivo vs Extensivo



- ¿cuánto? Regla del éxito 1/5
 - $\sigma = \sigma / c$ si la probabilidad de mutación $< 1/5$
 - $\sigma = \sigma \times c$ si la probabilidad de mutación $> 1/5$
- $0.8 < c < 1$. Schewefel derivó que $c=0.817$
- Más mecanismos de mutación.
 - Consideramos que σ está metida en el cromosoma

x1	x2	x3	x4	x5	x6	x7	σ
----	----	----	----	----	----	----	----------



- σ puede ser mutada

$$\sigma' = \sigma \cdot e^{\tau \cdot N(0,1)} \quad x'_i = x_i + \sigma' N(0,1)$$

- τ : Es la tasa de aprendizaje. Proporcional a $1/n^{0.5}$.
 - N es el número de variables en el cromosoma
- Podemos expandirlo más
 - Usar un σ para cada variable
 - Usar una tasa de aprendizaje global y una de aprendizaje coordinado.
 - Usar matrices de co-varianza.

- Volvemos a la selección
 - Hemos creado λ hijos de u padres.
 - Eliminamos determinísticamente los peores
- La base de selección es:
 - Conjunto de hijos (u, λ) –EE
 - Conjunto de padres e hijos $(u + \lambda)$ –EE
- $(u + \lambda)$ –EE es una estrategia “elitista”
 - Salir de óptimos locales.
 - Para purgar los valores de σ malos
- (u, λ) –EE puede “olvidar”



METAHEURÍSTICAS BIOINSPIRADAS MODERNAS:

ALGORITMOS GENÉTICOS Y EVOLUTIVOS

■ Esquema General de un Algoritmo Evolutivo

```
Generación de Población Inicial  $G(i=0)$   
Calculo de Fitness de  $G(i=0)$   
While (Criterios de Parada =FALSE)  
     $i=i+1$   
    Generación de  $G(i)$   
        Procedimiento de Selección.  
        Procedimiento de Cruce.  
        Procedimiento de Mutación  
    Calculo de Fitness de  $G(i)$ .  
End While
```

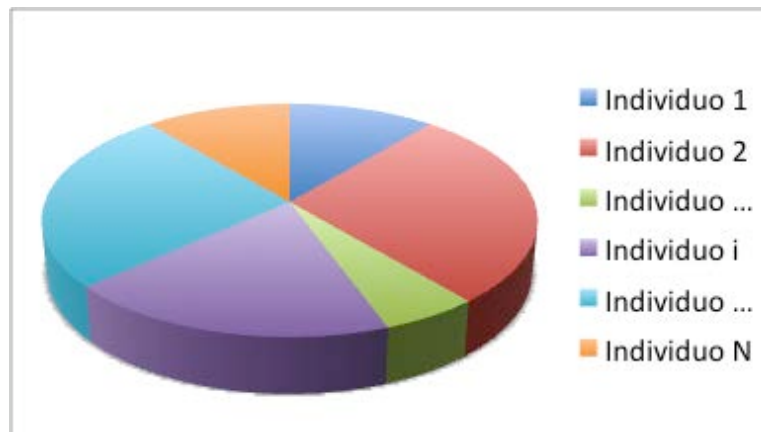
OPERADORES EVOLUTIVOS: SELECCIÓN

■ Procedimiento de Selección

- La selección se realiza considerando la función de Fitness.
- Muchos procedimientos: Ruleta, Elitista, Torneo, Escalafón

Población Original

Individuo 1
Individuo 2
.....
Individuo i
.....
Individuo N



Población Seleccionada

Individuo i
Individuo ...
Individuo 2
¿?
¿?
¿?

- Mediante la operación cruce se consiguen nuevos individuos para la próxima generación

■ Cruce en 1 Punto

Padre 1	1	4	1	3	2	2	4	4	1	1	3	3	3	2	1
Padre 2	2	1	4	4	3	1	2	4	2	1	3	1	3	2	4
Hijo 1	1	4	1	3	2	2	4	4	2	1	3	1	3	2	4
Hijo 2	2	1	4	4	3	1	2	4	1	1	3	3	3	2	1

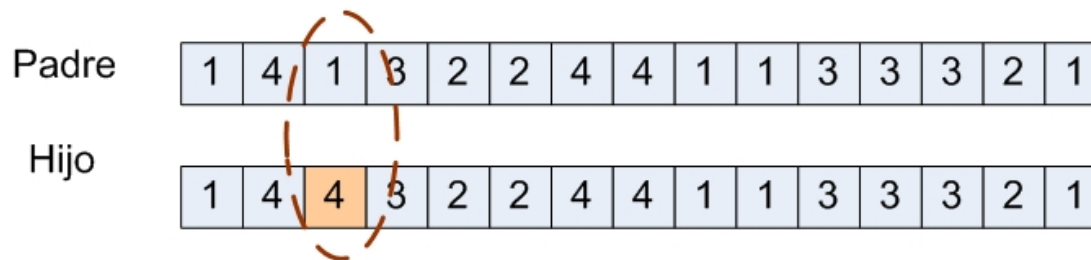
■ Cruce en 2 Puntos

Padre 1	1	4	1	3	2	2	4	4	1	1	3	3	3	2	1
Padre 2	2	1	4	4	3	1	2	4	2	1	3	1	3	2	4
Hijo 1	1	4	1	3	3	1	2	4	2	1	3	3	3	2	1
Hijo 2	2	1	4	4	2	2	4	4	1	1	3	1	3	2	4

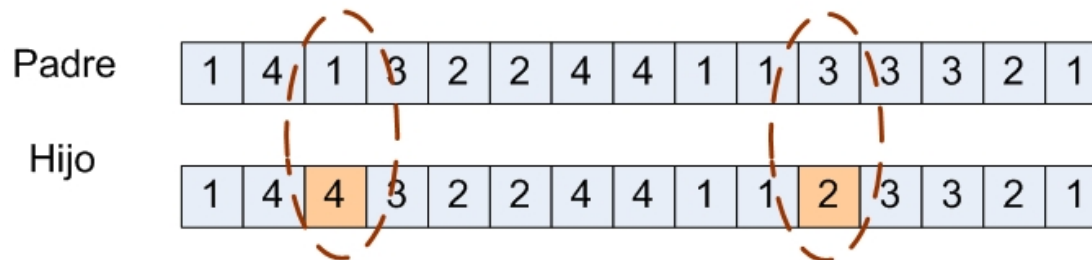
OPERADORES EVOLUTIVOS: MUTACIÓN

- El operador Mutación introduce la diversidad genética necesaria para explorar adecuadamente el espacio de búsqueda.
- Previene la caer en máximos/ mínimos locales

Mutación en 1 Punto



Mutación en Varios Puntos



ALGORITMOS EVOLUTIVOS: FINAL....?

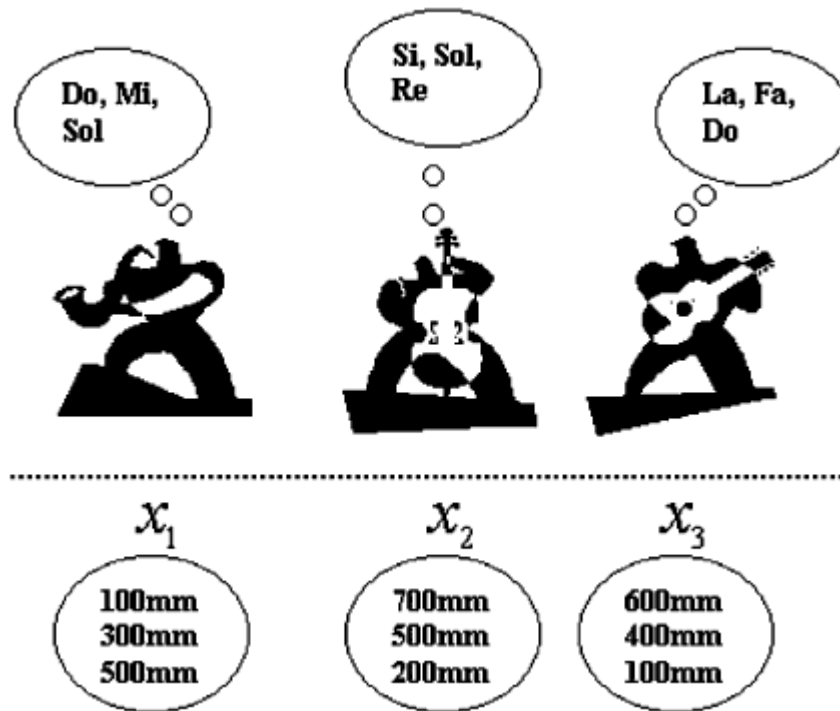


- Hay muchas cosas que estudiar y que tener en cuenta
 - Tipo y probabilidades de cruce y mutación
 - Tipo de selección.
 - Cuando se hace la selección
 - Aplicación o no de Elitismo.
 - Reparación de Soluciones
- Y además se pueden hibridizar con otras técnicas
 - Algoritmos Evolutivos + Búsqueda Local
 - Algoritmos Evolutivos + Redes de Hopfield
 - Algoritmos Evolutivos + Algoritmos Evolutivos.

METAHEURÍSTICAS BIOINSPIRADAS MODERNAS:

HARMONY SEARCH

- El algoritmo de Harmony Search (Búsqueda en Armonía) es una actualización moderna de las estrategias evolutivas.
- Basado en como una orquesta de jazz improvisa.





■ Como funciona?

- Conjunto de soluciones (Harmony Memory)
- Cada solución al problema es una armonía, formada por notas.
- En cada iteración del algoritmo se genera una nueva armonía por improvisación.
- Improvisación:
 - Harmony Memory Considering Rate.
 - Pitch Adjusting Rate.
 - Random Selection Rate.



- Improvisación 1: HMCR.
- Consiste en el primer paso de generación de una nueva armonía.
- Cada nota es generada de forma independiente, de manera que el HMCR es una probabilidad de que dicha nota sea obtenida de la Harmony Memory.
- En caso contrario, se genera aleatoria.

$$x'_i \leftarrow \begin{cases} x'_i \in \{x_i^1, x_i^2, \dots, x_i^{HMS}\} & \text{con probabilidad } HMCR \\ x'_i \in X_i & \text{con probabilidad } (1 - HMCR) \end{cases}$$



- Improvisación 2: PAR
- Es el segundo paso de la improvisación.
- Cada nota es ajustada en tono con una probabilidad PAR.
- En caso contrario, no se ajusta la nota.
- Se define un valor de ancho de banda bw , y un número aleatorio ϵ con una distribución $u(-1,1)$

$$x'_i \leftarrow x'_i + bw \cdot \epsilon$$



- Improvisación 3: RSR y aceptación/rechazo de armonía.
- Finalmente, algunas notas son remplazadas por otras de forma aleatoria, con una probabilidad RSR.
- La nueva armonía se compara con la peor del Harmony Memory, y si es mejor que ésta última la sustituye.
- El algoritmo funciona en bucle de improvisación y se detiene usualmente por número de iteraciones.

METAHEURÍSTICAS BIOINSPIRADAS MODERNAS:

EVOLUCIÓN DIFERENCIAL



- El algoritmo de Evolución diferencial (Differential Evolution) es un algoritmo de tipo evolutivo.
- Está basado en el procedimiento habitual de selección, y recombinación de individuos, pero con características especiales.
- Primero se *perturba* a un individuo:

$$X_c' = X_c + F(X_a - X_b)$$

Con F factor de recombinación [0.1, 2].



- Este vector *perturbado* se cruza con otro X_t , para formar un hijo X'_t . Normalmente este proceso se lleva a cabo con un cruce uniforme, en el que se define una probabilidad CR de que el hijo herede parte del vector X'_c .
- El mejor de los vectores formados entre X_t y X'_t sobrevive en la próxima generación.
- Se establecen una serie de estrategias, definidas como:

DE/X/Y/Z

DE: Differential Evolution.

X: individuo a perturbar.

Y: Número de individuos en la operación de perturbación.

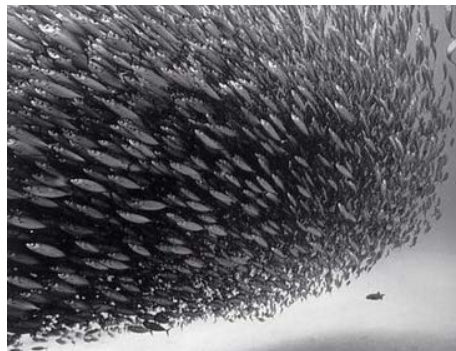
Z: tipo de cruce.

- Ejemplo: DE/best/2/bin.

METAHEURÍSTICAS BIOINSPIRADAS MODERNAS:

PARTICLE SWARM OPTIMIZATION

- PSO significa Particle Swarm Optimization.
- Fue desarrollado por Rusell C. Eberhart y James Kennedy en 1995
- Esta inspirado por el comportamiento social de bandadas de pájaros o bancos de peces
- Entra dentro de la categoría de optimización basada en poblaciones, similar a algoritmos genéticos





- **Características de un enjambre**
- Compuesto por agentes simples (auto-organizado)
- Descentralizado: No existe un único supervisor
- No hay un plan global (comportamiento emergente)
- Robusto: Las actuaciones se completan aunque el individuo falle
- Flexible
 - Puede responder a cambios externos
 - Percepción del entorno (sentidos)
 - No existe un modelo explícito de entorno ni habilidad para cambiarlo.



■ Inteligencia de Enjambres:

- propiedad de un sistema por la cual el comportamiento colectivo de agentes (no sofisticados), sin un control centralizado, interactúa localmente con el entorno proporcionando un patrón global de funcionamiento coherente conocido como “emergente”
- Partículas simples, dispuestas adecuadamente en un enjambre conducen a resultados adecuados



- Los pájaros y los peces **ajustan su movimiento** para evitar a los depredadores y **buscar comida** y compañeros
- Supongamos que una bandada de pájaros busca comida en una zona donde sólo hay una fuente de comida
- Los pájaros **no saben donde está** la comida pero **sí conocen su distancia** a la misma
- La estrategia más eficaz para hallar la comida es **seguir al ave que se encuentre más cerca de ella**
- La Optimización Basada en Nubes de Partículas (PSO) simula este comportamiento para diseñar algoritmos avanzados de optimización numérica



- Cada solución (**partícula**) es un “ave” que está siempre en continuo movimiento en el espacio de búsqueda (“vuela”)
- La **nube de partículas es un sistema multiagente**. Las partículas son agentes simples que guardan (y comunican) la mejor solución que han encontrado
- Cada partícula tiene un **fitness** (calidad de la solución), una **posición** y un **vector velocidad** que dirige su “vuelo”
- El movimiento de las partículas por el espacio está guiado por su mejor estado y por el de las partículas de su entorno (**interacción social**)



- Cada partícula está compuesta por
- Tres vectores
 - **X**: Almacena su posición actual.
 - **pBest**: almacena la mejor posición encontrada
 - **V**: Almacena el gradiente según el que se moverá
- Dos valores de fitness
 - **X_fitness**: Almacena la calidad de la solución actual
 - **pBest_fitness**: Almacena la calidad de la mejor solución/posición encontrada

p_i

$X_i = \langle x_{i1}, \dots, x_{in} \rangle$
 $pBest_i = \langle p_{i1}, \dots, p_{in} \rangle$
 $V_i = \langle v_{i1}, \dots, v_{in} \rangle$
 $x_fitness = ?$
 $pBest_fitness = ?$



- La nube se inicializa generando las posiciones y las velocidades iniciales de las partículas
- Las posiciones se pueden generar aleatoriamente en el espacio de búsqueda, de forma regular o con una combinación de ambas
- Las velocidades se generan aleatoriamente, con cada componente en el intervalo **$[-V_{max}, V_{max}]$**
- **V_{max}** será la velocidad máxima que pueda tomar una partícula en cada movimiento



- ¿ Cómo se mueve una partícula de una posición del espacio de búsqueda a otra?
- Se hace simplemente añadiendo el vector velocidad V_i al vector posición X_i para obtener un nuevo vector posición:

$$X_{i+1} \rightarrow X_i + V_i$$

- Una vez calculada la nueva posición de la partícula, se evalúa ésta. Si el nuevo fitness es mejor que el que la partícula tenía hasta ahora, **pBest_fitness**, entonces:

$$pBest_i \leftarrow X_i; pBest_fitness \leftarrow x_fitness$$



- El primer paso es ajustar el vector velocidad, para después sumárselo al vector posición
- Las fórmulas empleadas son las siguientes:

$$v_{id} = \omega \cdot v_{id} +$$

$$\phi_1 \cdot \text{rand}() \cdot (pBest_{id} - x_{id}) +$$

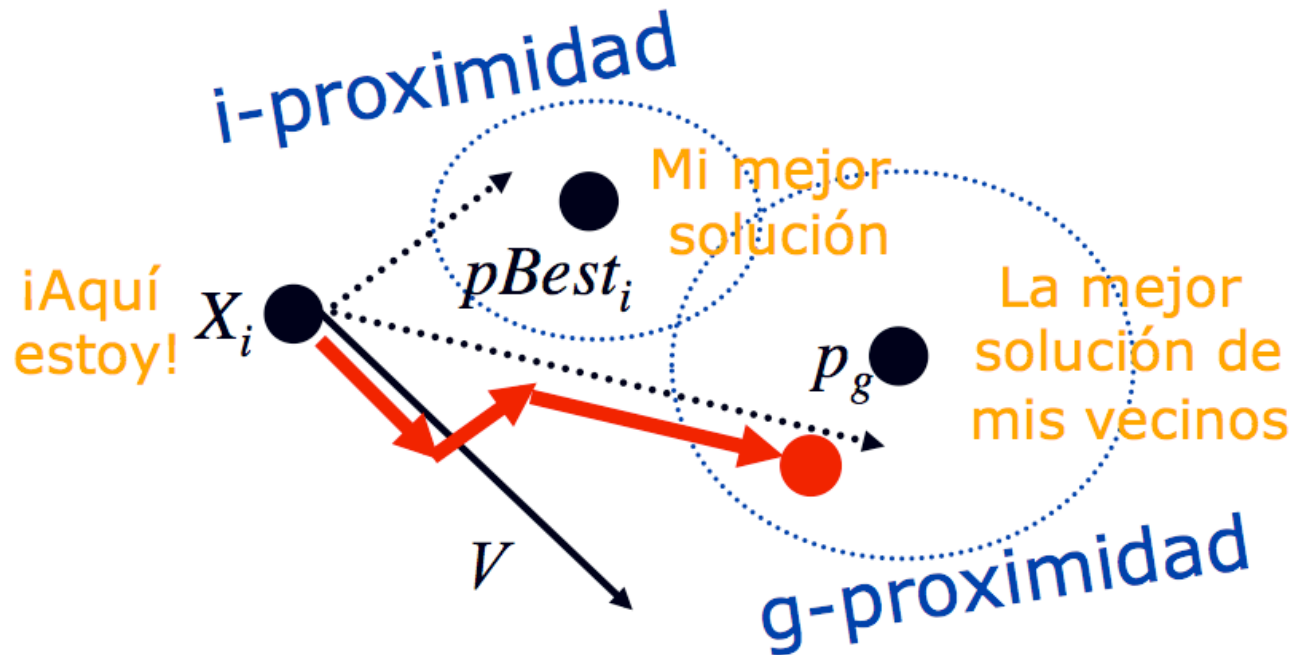
$$\phi_2 \cdot \text{rand}() \cdot (g_{id} - x_{id})$$

Cognitiva

Social

$$X_{(i+1),d} = X_{i,d} + v_{i,d}$$

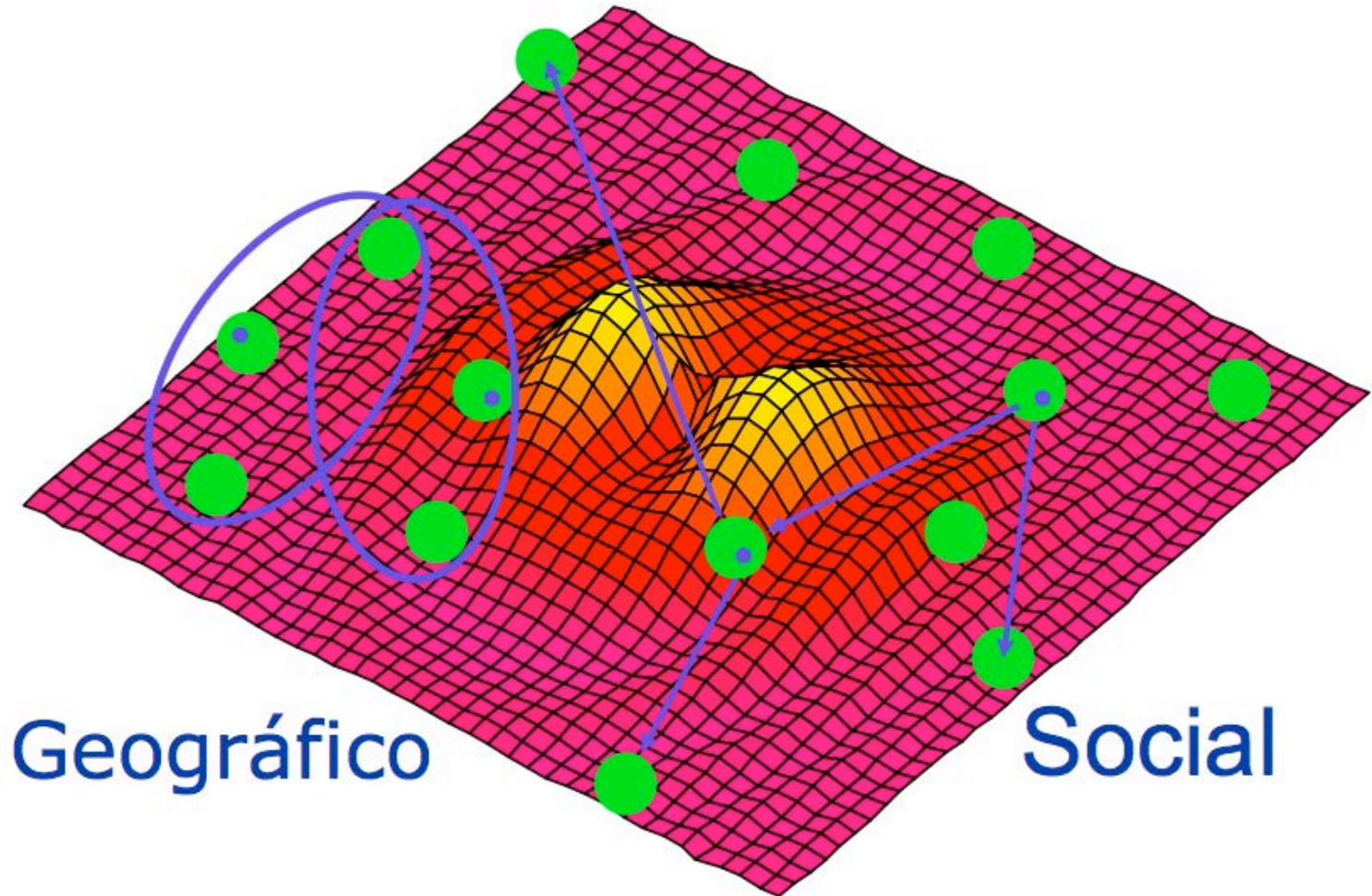
- P_i es la partícula en cuestión,
- ϕ_1, ϕ_2 son ratios de aprendizaje (pesos) que controlan los componentes cognitivo y social,
- **g** es la partícula con el mejor pBest_fitness del entorno de p_i (lBest) o de toda la nube (gBest),
- $\text{rand}()$ son números aleatorios generados en $[0,1]$
- d es la d -ésima dimensión del vector





- Las topologías definen el entorno de cada partícula individual. La propia partícula siempre pertenece a su entorno
- Los entornos pueden ser de dos tipos:
 - **Geográficos:** se calcula la distancia de la partícula actual al resto y se toman las más cercanas para componer su entorno
 - **Sociales:** se define a priori una lista de vecinas para cada partícula, independientemente de su posición en el espacio. Los entornos sociales son los más empleados
- Una vez decidido el entorno, es necesario definir su tamaño. El algoritmo no es muy sensible a este parámetro
- Cuando el tamaño es toda la nube de partículas, el entorno es a la vez geográfico y social, y tenemos **la PSO global**

OPTIMIZACIÓN POR ENJAMBRE



Geográfico

Social

METAHEURÍSTICAS BIOINSPIRADAS MODERNAS:

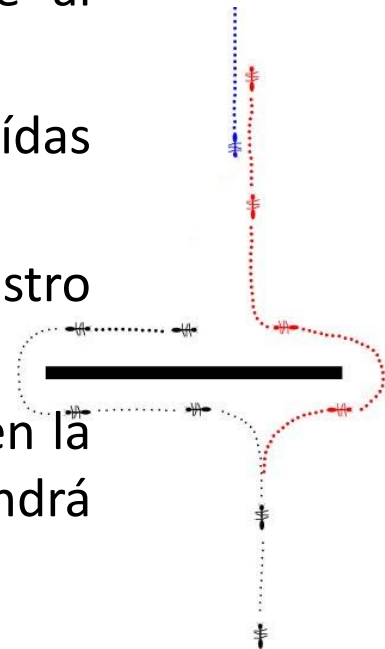
ALGORITMOS BASADOS EN COLONIAS DE HORMIGAS

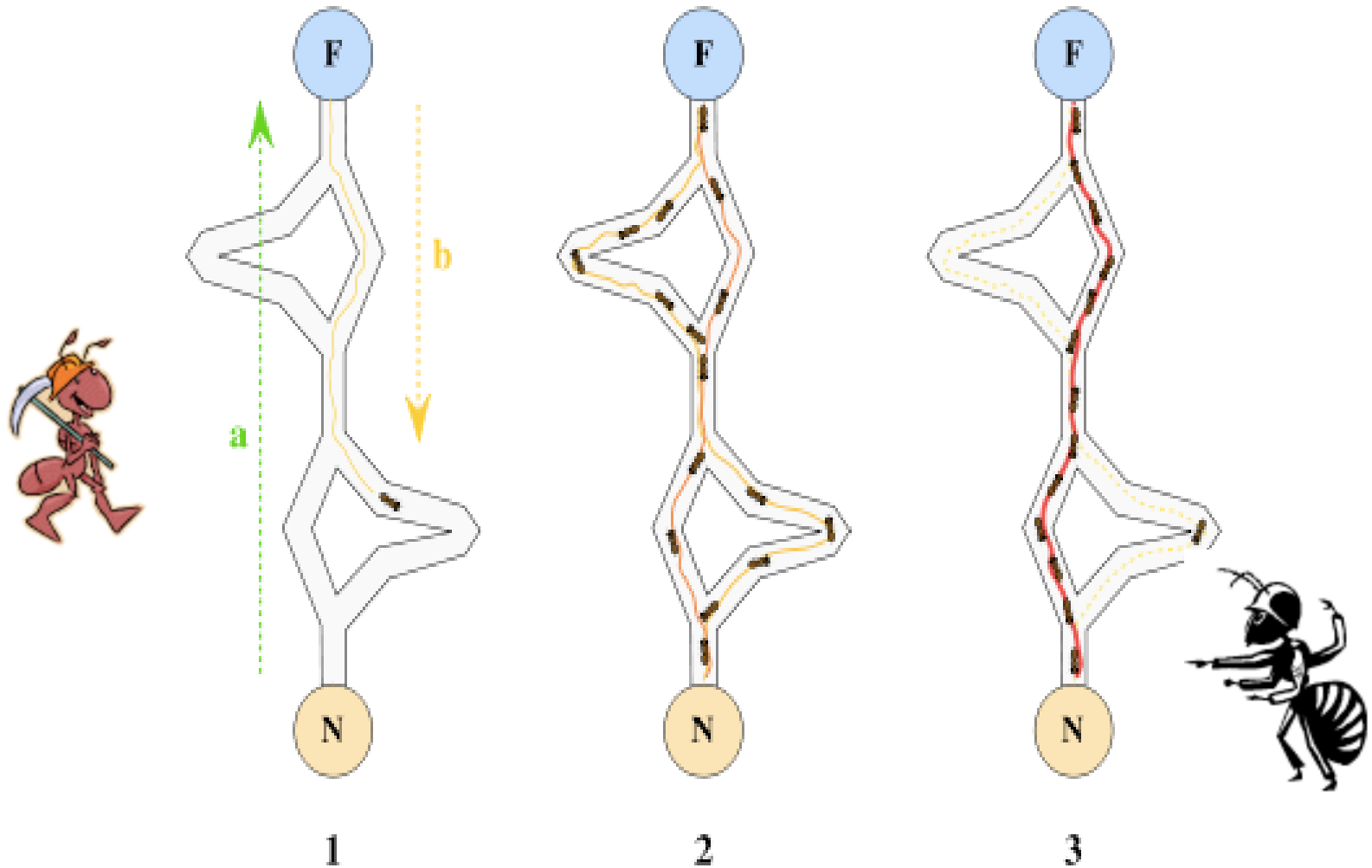


- Las hormigas son insectos sociales que viven en colonias y tienen un comportamiento dirigido al bienestar de la colonia en su conjunto.
- Una característica importante es que siempre encuentran el camino más corto entre el hormiguero y la comida
 - Peculiaridad : Las hormigas son ciegas.
- Sin embargo las hormigas segregan un elemento químico denominado feromona que es el que les permite averiguar por donde han ido las hormigas anteriores.
- Las feromonas son volátiles con el tiempo, si no son renovadas el “olor” se va y el camino desaparece



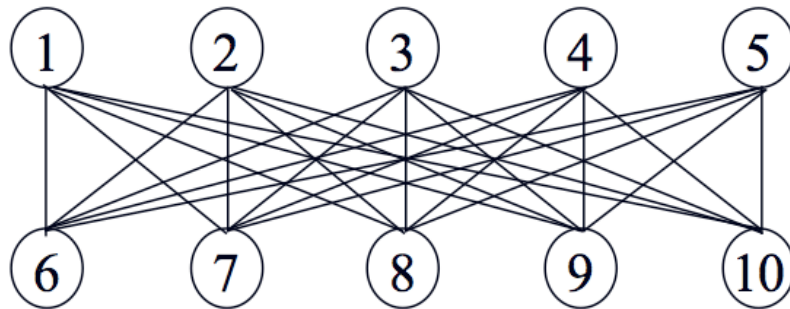
- Una **“hormiga”** sale del hormiguero en busca de comida de manera aleatoria
- Si la encuentra, vuelve lo más directamente posible al hormiguero dejando un rastro de feromonas
- Las hormigas cercanas se sentirán mas o menos atraídas por este rastro.
- En la vuelta a la colonia las hormigas incrementan el rastro de feromonas.
- Si hay dos rutas posibles, **la más corta** será recorrida, en la misma cantidad de tiempo **por más hormigas** Tendrá más feromonas será más atractiva.
- La otra ruta, al no ser transitada perderá sus feromonas
- Al final todas las hormigas irán por el camino más corto







- Para aplicar la optimización basada en colonia de hormigas a un determinado problema es necesario conseguir una representación en forma de grafo con pesos en los enlaces.



Pesos =

	1	2	...	10
1	-	-		p_{1-10}
2	-	-		p_{2-10}
...				
10	p_{1-10}	p_{2-10}		-



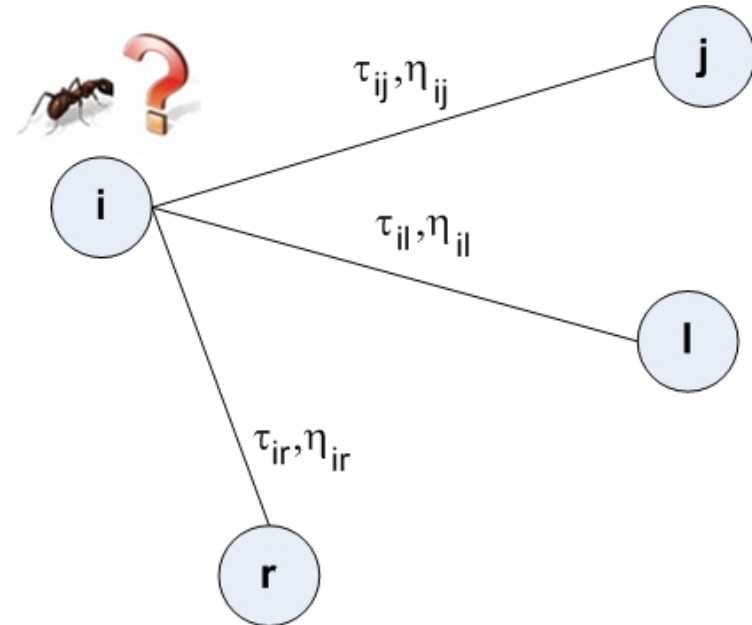
- Cada arco tiene dos tipos de información:
 - **Rastro de feromonas:** Medida de la deseabilidad de un arco representada por la **cantidad de feromona** depositada en el por las hormigas anteriores y que **es modificada** durante la ejecución del algoritmo
 - **Información heurística:** Preferencia del arco que depende del **problema concreto** y que **no es modificada** por las hormigas durante la ejecución del algoritmo.





- En cada iteración, cada hormiga artificial recorre **el grafo generando un camino completo (solución al problema)**.
- En cada paso, elige hacia qué nodo moverse según una regla probabilística de transición.
- La bondad de estas soluciones determina el aporte de feromona que realiza cada hormiga en el camino recorrido
- Se incorpora un mecanismo de evaporación de feromona más activo que el natural, lo que evita la perduración de los rastros de feromona y, por tanto, el estancamiento en óptimos locales

- El proceso constructivo de la hormiga se basa en una regla probabilística de transición sesgada por:
- La información heurística existente sobre el problema (grado de “deseabilidad” del arco)
- Las bondad de las decisiones que otras hormigas tomaron en el pasado (representada en los rastros de feromona)



$$P_{ij}(t) = \frac{[\tau_{ij}]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{q \in J} [\tau_{ij}]^{\alpha} \cdot [\eta_{ij}]^{\beta}}$$

METAHEURÍSTICAS BIOINSPIRADAS MODERNAS:

TEMPLE SIMULADO

TEMPLE SIMULADO



- Temple Simulado (Simulated Annealing)
- Es un algoritmo iterativo constructivo
- Pensemos en cómo se modela el metal en una fragua.
- Inicialmente se calienta a alta temperatura
 - Permite hacer cambios sustanciales en la forma.
- Se enfría poco a poco siguiendo una periodicidad.
- El principio físico del optimización:
 - *Si se calienta el metal a una temperatura suficientemente alta, para asegurar un estado aleatorio y se enfría suficientemente despacio para asegurar un equilibrio térmico, los átomos se colocarán en un estado de mínima energía.*

TEMPLE SIMULADO



- **Paso 1:** Inicializamos. Empezamos con una posición aleatorio. Se fija a la temperatura muy alta
- **Paso 2:** Se muta la posición conforme a una regla predefinida.
- **Paso 3:** Se calcula el nuevo fitness.
- **Paso 4:**
 - Si el fitness es mejor se acepta el cambio
 - Si no, puede aceptarse dependiendo de una probabilidad en función de la temperatura
- **Paso 5:** Se repite k veces para cada temperatura.
- **Paso 6:** Decrementar la temperatura
 - El proceso se repite hasta que se alcanza el punto de enfriamiento..

Algorithm SIMULATED-ANNEALING

Begin

temp = INIT-TEMP;

place = INIT-PLACEMENT;

while (*temp* > FINAL-TEMP) **do**

while (*inner_loop_criterion* = FALSE) **do**

new_place = PERTURB(*place*);

$\Delta C = \text{COST}(\text{new_place}) - \text{COST}(\text{place});$

if ($\Delta C < 0$) **then**

place = *new_place*;

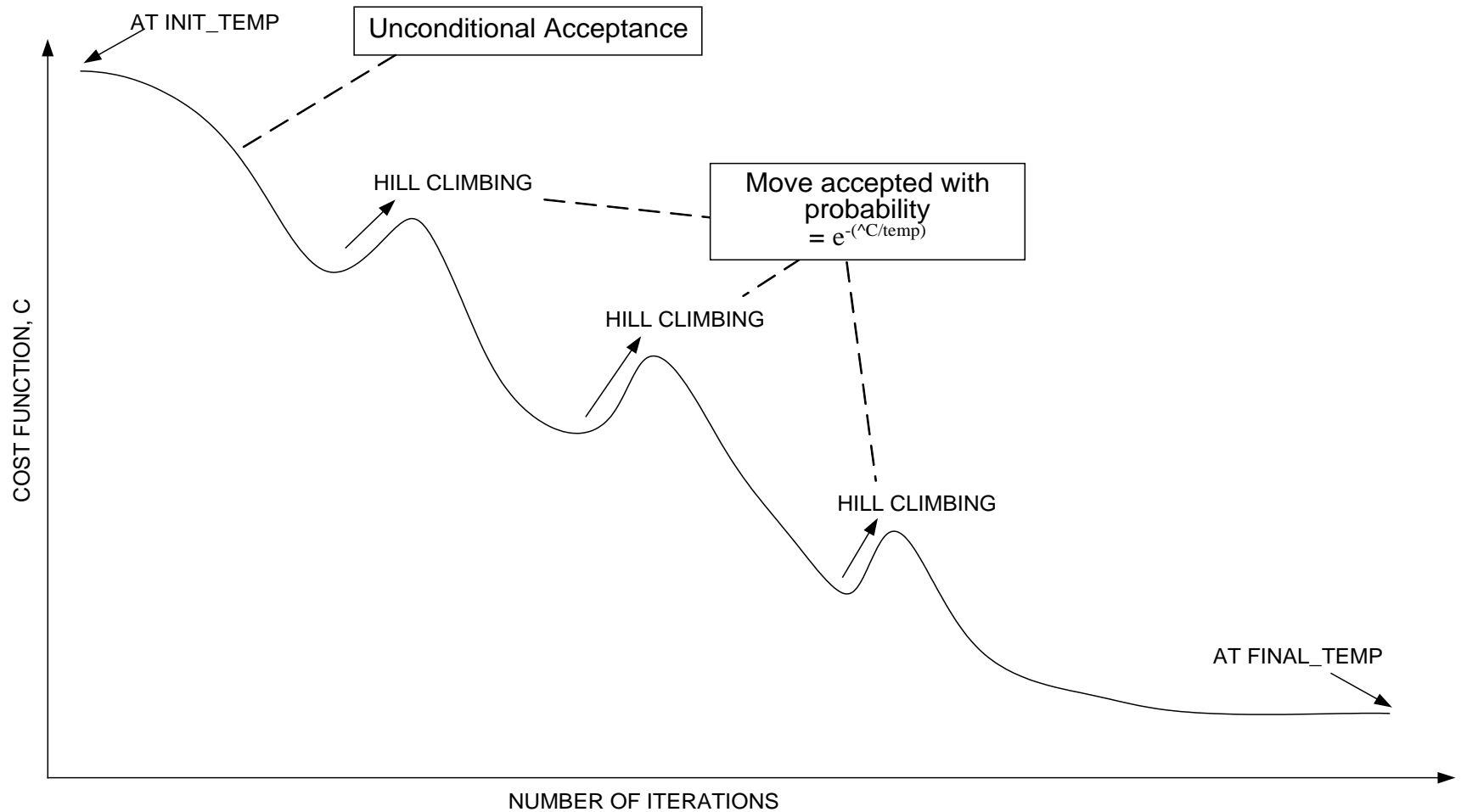
else if ($\text{RANDOM}(0,1) > e^{-(\Delta C/\text{temp})}$) **then**

place = *new_place*;

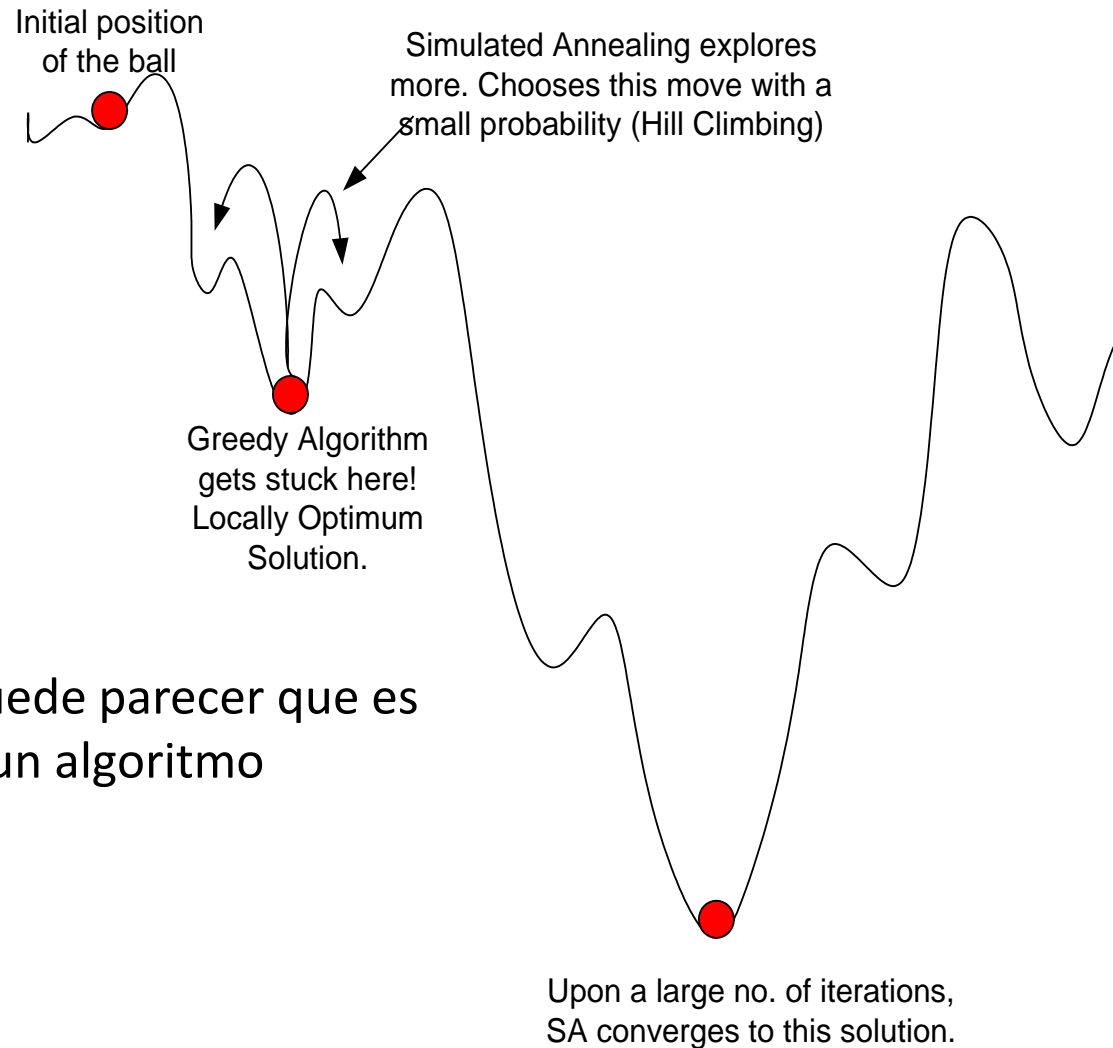
temp = SCHEDULE(*temp*);

End.

TEMPLE SIMULADO



TEMPLE SIMULADO



- A priori puede parecer que es igual que un algoritmo *greedy*.

METAHEURÍSTICAS BIOINSPIRADAS MODERNAS:

CORAL REEFS OPTIMIZATION

CORAL REEFS OPTIMIZATION



- Es un algoritmo basado en un grid de individuos.
- Es una mezcla entre algoritmos evolutivos y temple simulado, aunque está basado en el crecimiento de corales y arrecifes de coral.



- **Biología de los corales.**
- Los corales son animales invertebrados, pertenecientes a la familia phylum-cnidaria. A esta familia también pertenecen las anémonas, hidras o las medusas.
- Los corales viven como pólipos o colonias de pólipos, usualmente adheridos a un sustrato.
- Actualmente se conocen más de 2500 especies, tanto de aguas superficiales como de aguas profundas, y cada año se describen nuevas especies.
- Más de la mitad de las especies de coral son formadores de arrecifes, al formar un exoesqueleto calcáreo, que queda tras la muerte del pólipo.

- **Biología de los corales.**
- Los corales que forman arrecifes suelen vivir en aguas poco profundas, limpias y con una temperatura constante, sobre los 27°C.
- Los corales que forman arrecifes tienen dos tipos de reproducción sexual:
 - Broadcast Spawning.
 - Brooding.
- Todos los corales pueden reproducirse por reproducción asexual (Budding o Fragmentación).

CORAL REEFS OPTIMIZATION



- **Biología de los corales.**
- La mayoría de los corales de arrecife son broadcast spawners. El proceso se da a la vez en todo el arrecife, en un día concreto, y consiste en la liberación sincronizada de esperma y óvulos.
- Las larvas que se forman son arrastradas por las corrientes y caen, si encuentran un sustrato adecuado crecen.



CORAL REEFS OPTIMIZATION



- **Biología de los corales.**
- Algunos corales de arrecife son brooders.
- Esto implica que la reproducción se realiza dentro del propio coral, liberándose a la corriente directamente las larvas. La mayoría de los corales brooders son hermafroditas.



CORAL REEFS OPTIMIZATION



- **Biología de los corales.**
- Todos los corales pueden reproducirse asexualmente mediante budding o fragmentación.
- Budding consiste en la reproducción con el mismo genoma de pólipos en una colonia. La fragmentación involucra trozos de corales (cientos de pólipos), que se separan del arrecife y si encuentran el sustrato adecuado, pueden crecer.



CORAL REEFS OPTIMIZATION



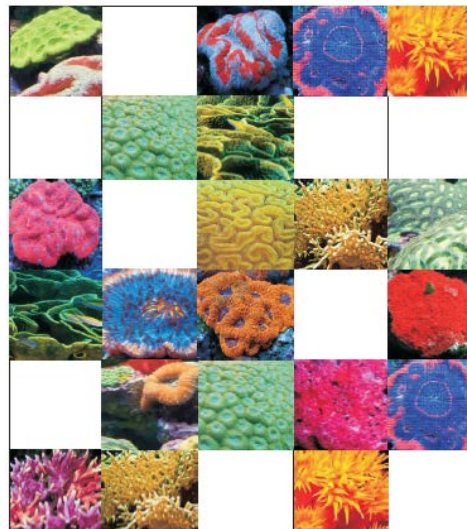
- **Biología de los corales.**
- En un arrecife formado, el espacio es vital para el nuevo crecimiento de corales. Se ha demostrado que los corales tienen estrategias agresivas para eliminar competencia y tener más espacio.
- Además los corales tienen enemigos naturales (depredadores como la estrella de mar corona de espinas), el aumento de la temperatura por el cambio climático, etc.



CORAL REEFS OPTIMIZATION



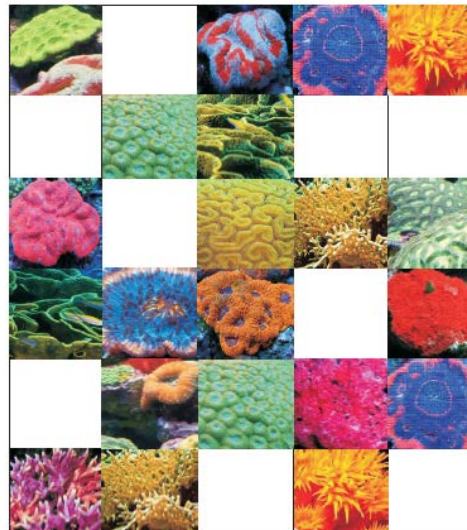
- El algoritmo de coral reefs optimization (CRO).
- El CRO simula los procesos de crecimiento, reproducción y predación de los corales en un arrecife.
- Primero se define el arrecife, donde cada coral representa una posible solución al problema.



CORAL REEFS OPTIMIZATION



- El algoritmo de coral reefs optimization (CRO).
- Posteriormente se lleva a cabo la simulación del proceso de broadcast spawning (mayoría del coral) y brooding (una pequeña parte del mismo).
- Las larvas generadas tratan de buscar un sitio en el arrecife, para ello se produce una lucha por el espacio.



CORAL REEFS OPTIMIZATION

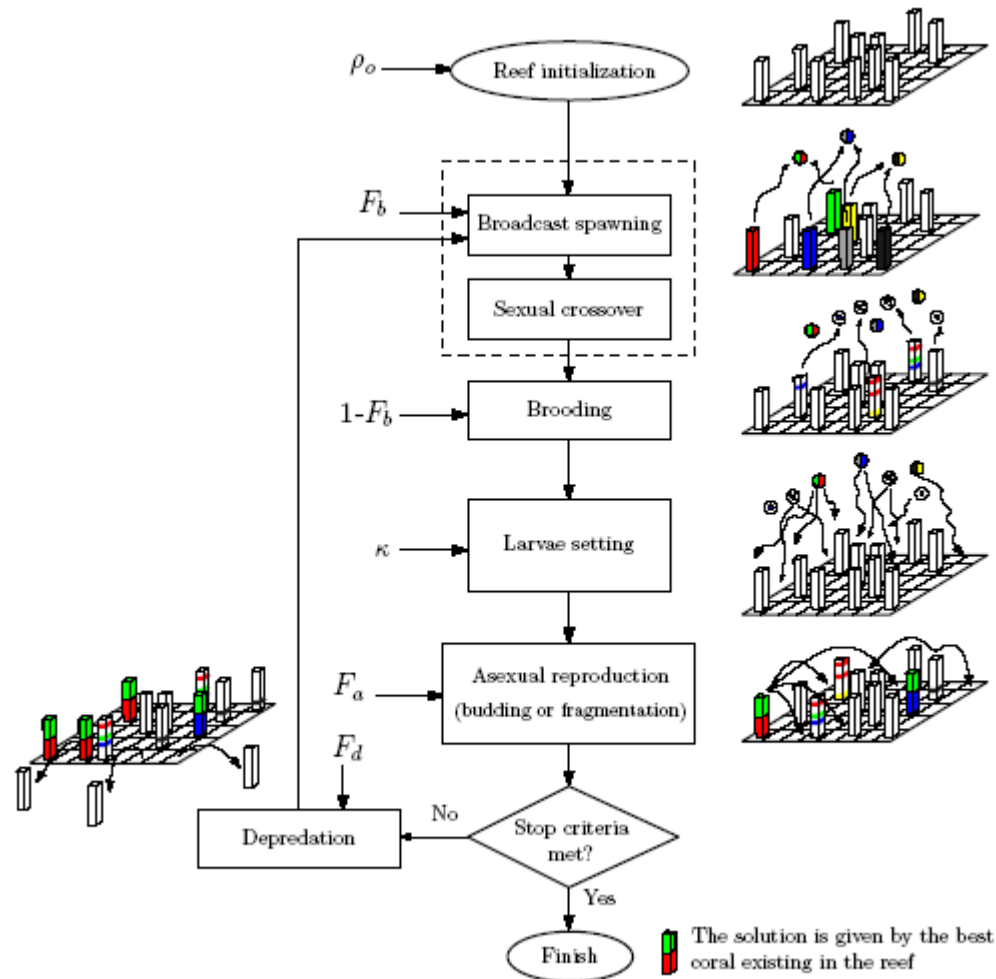


- El algoritmo de coral reefs optimization (CRO).
- Posteriormente se lleva a cabo la simulación del proceso de broadcast spawning (mayoría del coral) y brooding (una pequeña parte del mismo).
- Las larvas generadas tratan de buscar un sitio en el arrecife, para ello se produce una lucha por el espacio.
- Con una pequeña probabilidad, se simulan los procesos de fragmentación y depredación. La fragmentación se lleva a cabo con los corales mejores, y la depredación con un conjunto de los peores.
- El proceso se itera hasta alcanzar un número de generaciones prefijado.

CORAL REEFS OPTIMIZATION



- El algoritmo de coral reefs optimization (CRO).



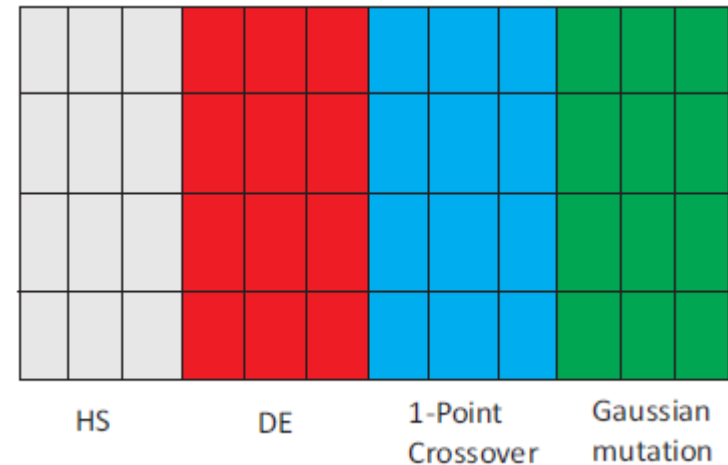
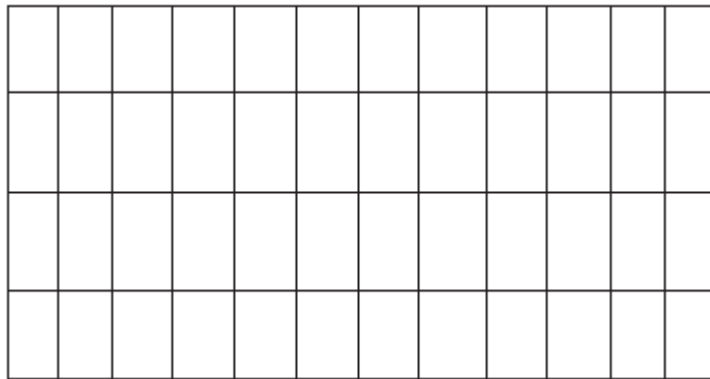
GENERALIZACIÓN DEL CRO

- La auténtica potencia del CRO es la generalización que es posible hacer del CRO con especies.
- El CRO con especies generaliza algoritmos de co-evolución con diferente modelización de codificación, operación, reparación, etc.
- Por ejemplo, con el CRO con especies se puede llevar a cabo co-evolución competitiva de diferentes codificaciones para un problema.
- Específicametine, es posible generalizar cualquier problema con codificación de longitud variable. Cada especie representaría individuos de una determinada longitud, que compiten con otros (ya que la función de fitness es común).



- Una estrategia similar se puede llevar a cabo con el **CRO con sustratos**.
- En este caso, el algoritmo estaría orientado a implementar versiones de co-evolución con diferentes operadores por sustrato.
- Sería posible también obtener **versiones de co-evolución competitiva para cualquier modelización diferente de determinadas características del problema** (diferentes operadores, cruce, mutación, reparación de individuos etc.) dependiendo del problema.
- Hemos empezado a explotar esta versión del CRO-SL.

CO-EVOLUCIÓN CON SUSTRATOS



- Supongamos que cada sustrato implementa una estructura de búsqueda diferente (HS, DE, cruce, mutación Gauss, etc).
- En este caso el CRO-SL evolucionará cada solución dependiendo del sustrato en el que caiga.

UNA INTRODUCCIÓN A LOS ALGORITMOS DE COMPUTACIÓN EVOLUTIVA

Sancho Salcedo Sanz

Departamento de Teoría de la Señal y Comunicaciones

Universidad de Alcalá