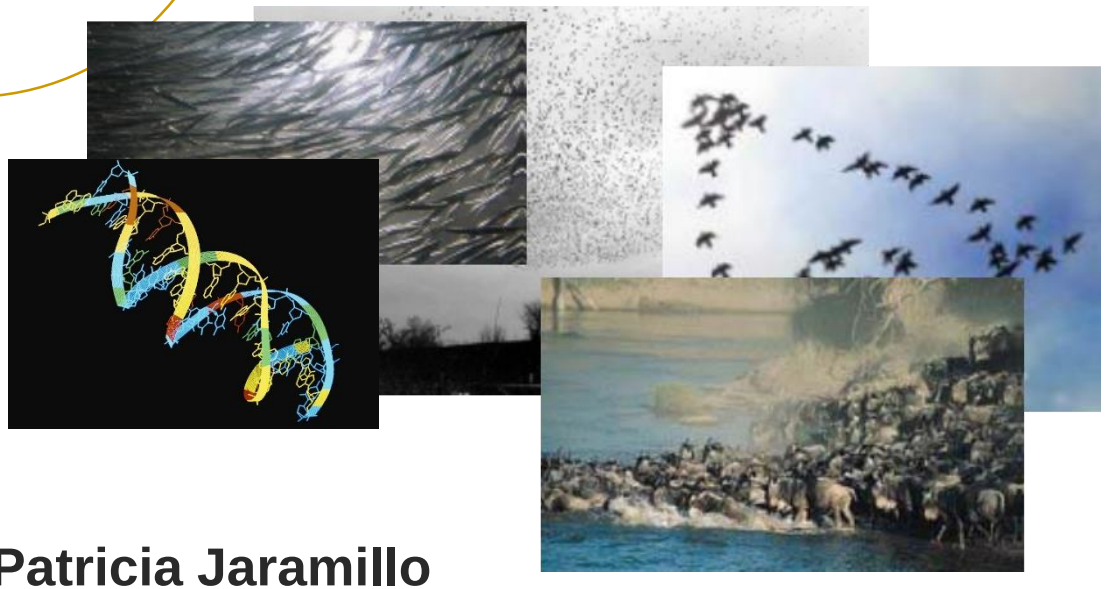
Universidad Nacional de Colombia
Sede Medellín

Computación Evolutiva



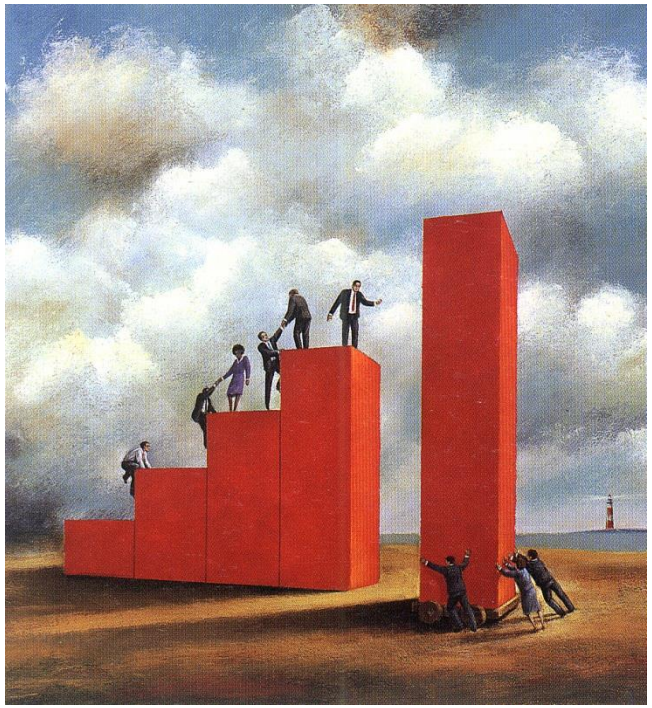
Profesora **Patricia Jaramillo**
gpjarami@unal.edu.co
Departamento de Ciencias de la Computación y la Decisión
Universidad Nacional de Colombia

Agenda

- ***El Qué:*** Optimización
- ***El Cómo*:**
  - Dificultades en la resolución de Problemas de optimización en IA
  - Computación Evolutiva y algoritmos bioinspirados
    - GA Algoritmos Genéticos
    - PSO Particle Swarm Optimization
    - Otros
- ***El Para Qué:*** Casos de aplicación
- Bibliografía

1.Introducción

Qué es optimizar?



Optimizar

Encontrar la **mejor** solución a un problema en el que se desee maximizar o minimizar un objetivo.

Ejemplos:

Diseñar una infraestructura o artefacto para minimizar costos o maximizar rendimiento.

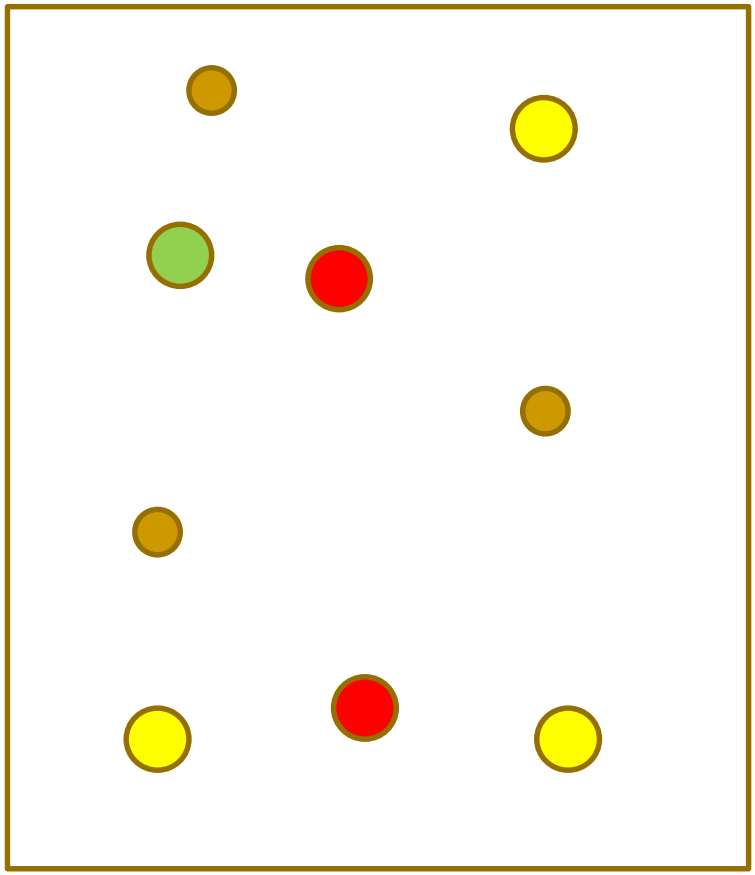
Elegir un portafolio de acciones para maximizar los retornos y minimizar el riesgo

Entrenar una red neuronal para minimizar el error.

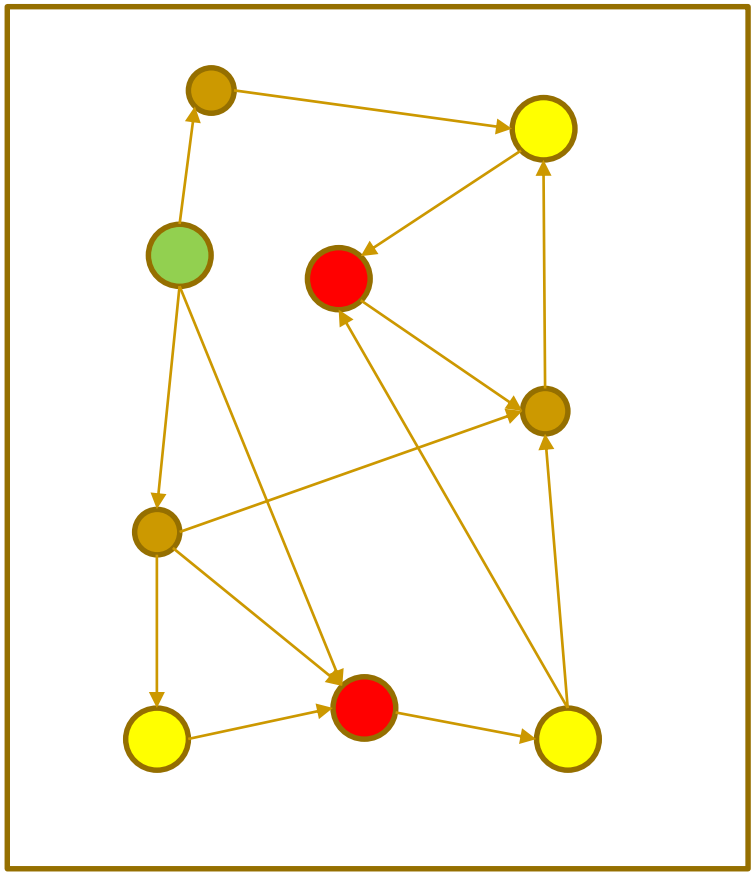
Diseñar la arquitectura de una red neuronal para minimizar el error

Fases de la decisión

Información

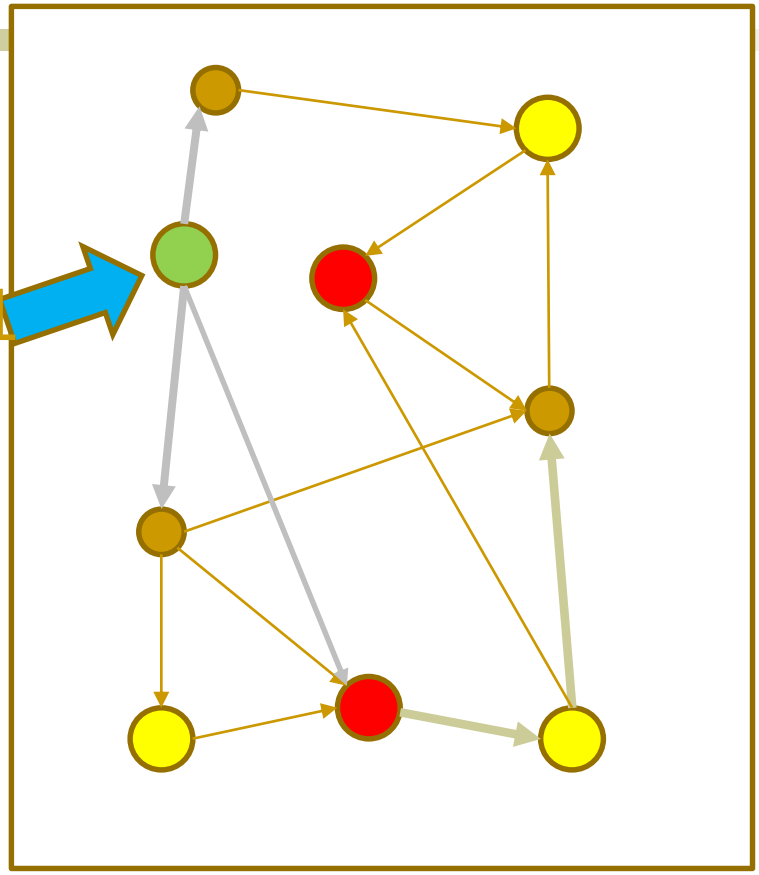


conocimiento



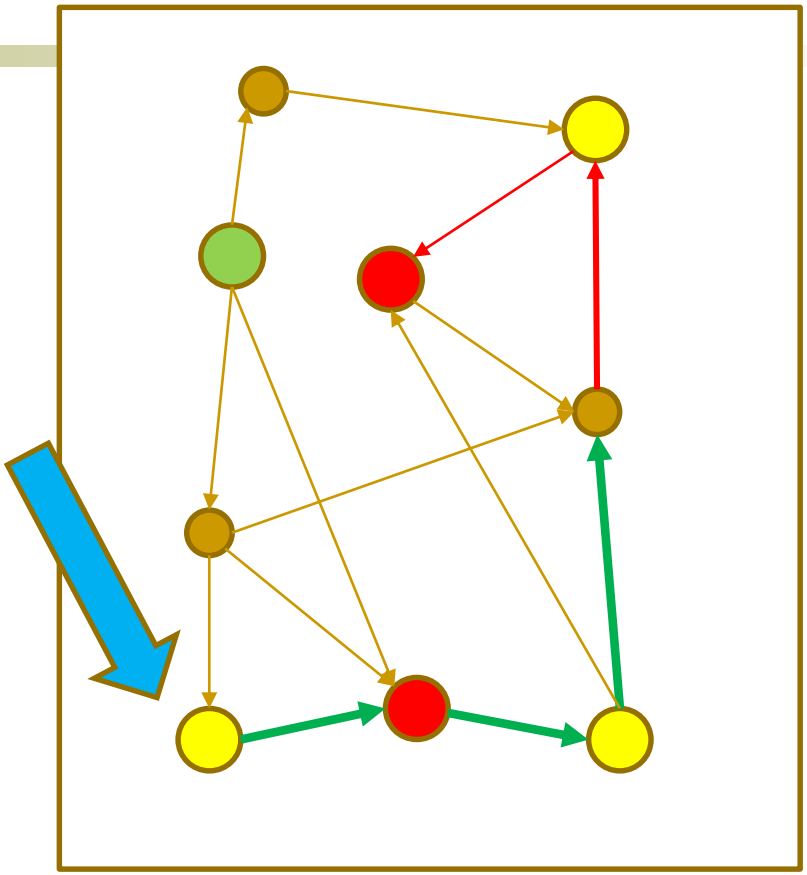
Decisión

Alternativa 1



Decisión

Alternativa 2



Todas las fases pueden apoyarse de la optimización, especialmente, la adquisición de conocimiento y la toma de decisiones.

Esquema general de un problema de optimización

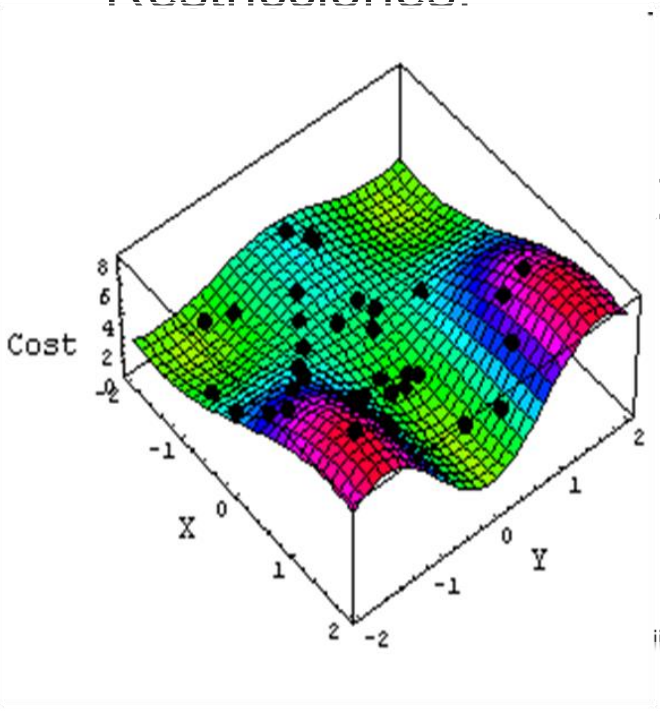
El primer paso para optimizar es modelar computacionalmente el problema del sistema

Función objetivo: Optimizar (min o max) $f(x)$

Restricciones:

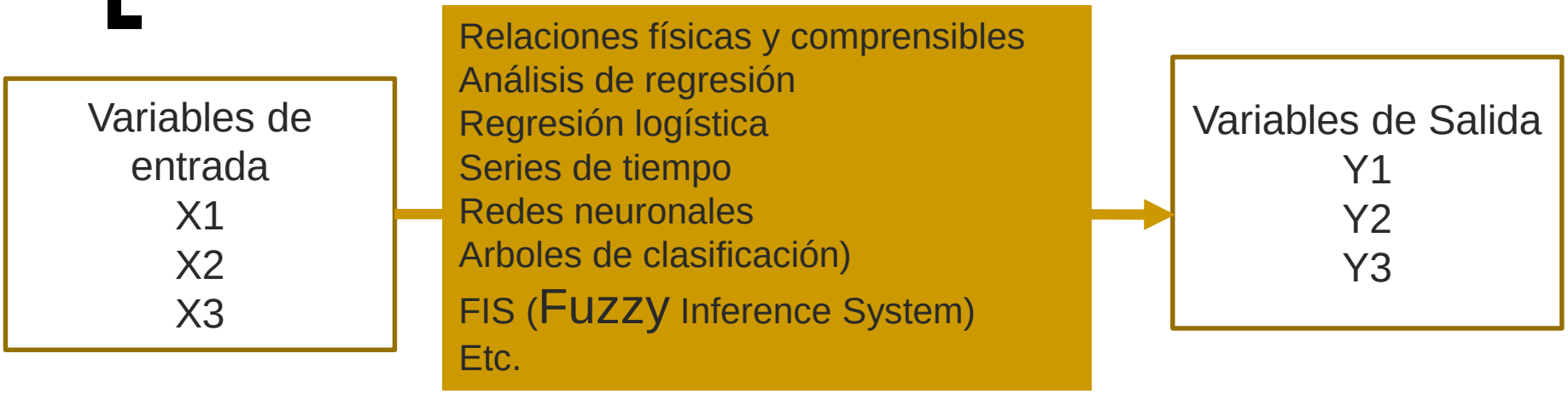
$LB \leq x \leq UB$

$G(x) \leq b$



X son variables de decisión

Ejemplo: Identificación de relaciones entre variables

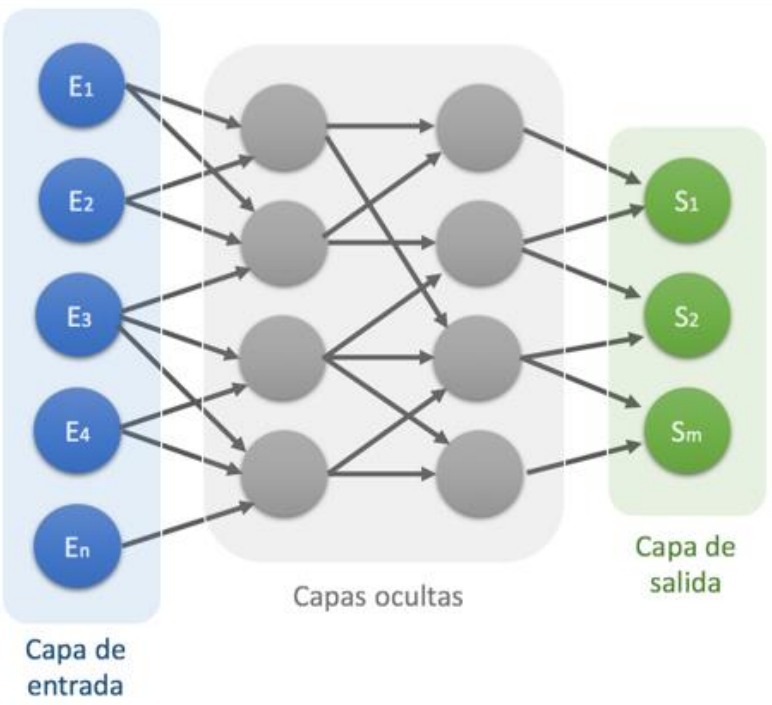


Elegir el modelo y calibrar el modelo que minimice el error entre la salida del modelo y la salida real

Permiten:

- Predecir el valor de las variables de salida, ante un nuevo caso (X1,X2,X3)
- Ayudar al entendimiento de las relaciones para para toma de decisiones

Ejemplo: Entrenamiento de una red neuronal



- Minimizar sum(l, (sum(n, $(y_{nl}-S_{nl})^2$))
- Variables de decisión: pesos y sesgos, wjk, θj

Ejemplo: Diseño de redes

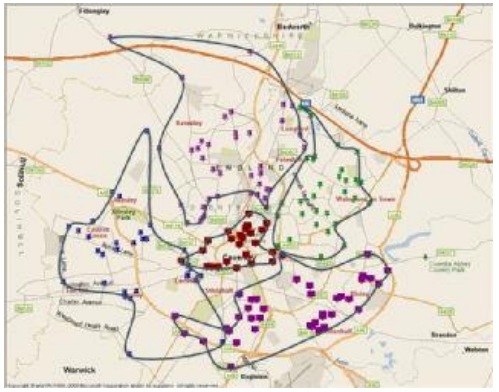
Ejemplo: determinar la manera mas económica de diseñar una línea de metro, sujeto a un presupuesto maximizando la demanda atendida en las diferentes paradas

Max sum(i, D(i) Y(i))

$$X(ij) \leq Y(i)$$

$$\sum_j X(i,j) = \sum_j X(j,i) \leq 1$$

$$C(i,j)X(i,j) + CEST\ Y(i) <= \text{PRES.}$$



Número de nodos	Posibles soluciones
6	60
7	360
10	181.000
20	10.000.000.000.000.000
100	???????

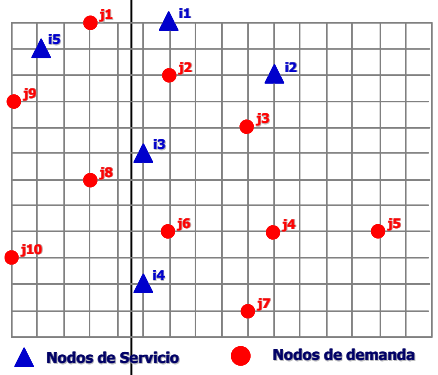
Ejemplo: A cuáles hospitales abastecer de equipos UCI

El gobierno tiene presupuesto para dotar hospitales y tiene que decidir a cuáles de los hospitales reforzar con equipos de respiración y camas para atender las urgencias de coronavirus, durante 3 meses. La demanda se asocia con la población más vulnerable, que ya ha sido analizada por una NN: variables: lugar donde habita, condiciones socioeconómicas, edad y antecedentes de salud. Se desea maximizar las vidas salvadas.

Y_{it}= *1 si se mejora el hospital i en mes t*
X_{ij}= *Numero de camas para el hospital i en t que atenderían a enfermos de la zona j en mes t, simultáneamente.*

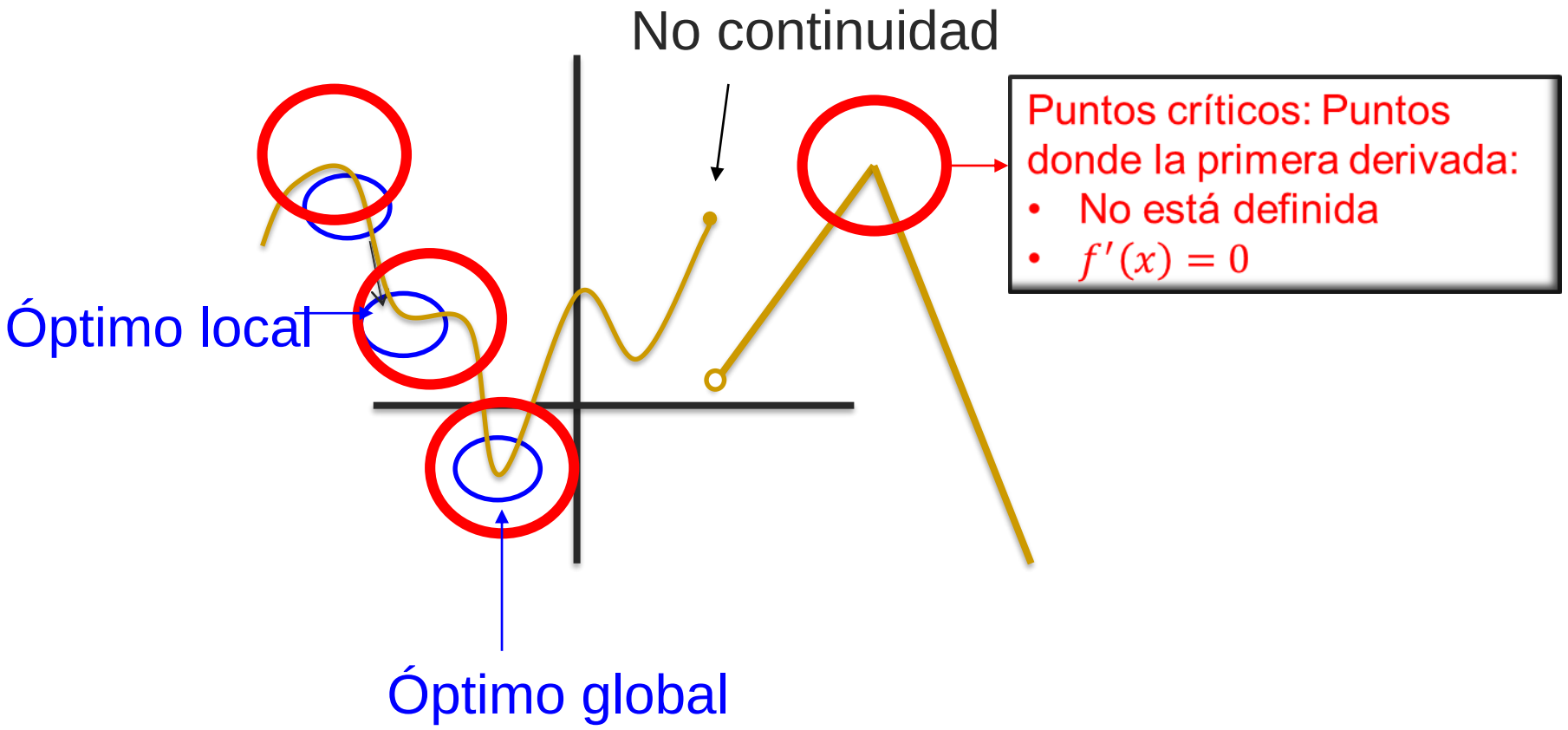
Maximizar $\sum_i \sum_i \sum_t$ *(4-t)*Prioridad(j) D(j,t)/dist(i,j) X(i,j,t))*

Sujeto a
Para cada demanda j ,t $\sum_i X(i,j,t) <= D(j,t)$
presupuesto limitado para cada mes t $\sum_i COpenY(i,t) + \sum_i \sum_j$ *COEQUIP X(i,j,t) <=PRES(t)*
cada hospital i en cada t $\sum_j X(i,j,t) <= Capacidad(i)(Y(i,t)+Y(i,t-1))$
X(i,j,t) enteras
Y(i,t) binarias

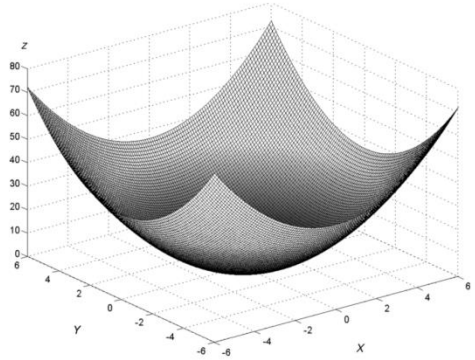


Y la incertidumbre…..

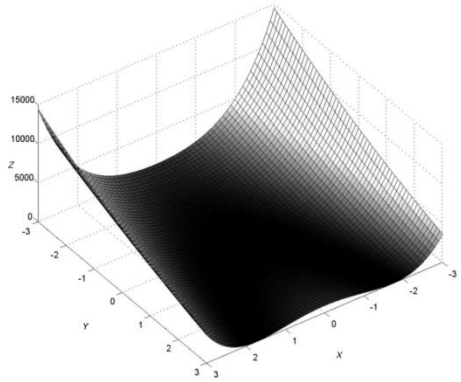
Conceptos básicos sobre funciones



Ejemplos:



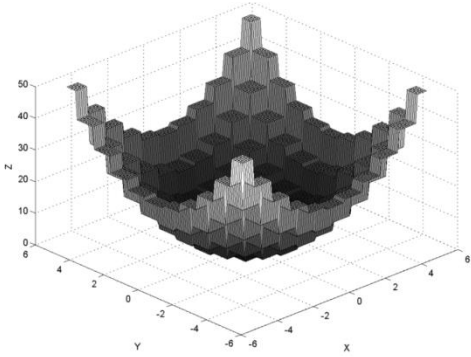
(a) f_1: Función esférica



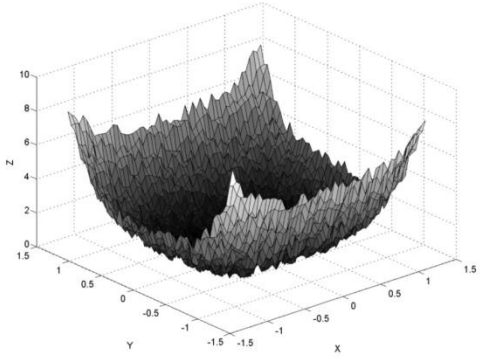
(b) f_2: Función Rosenbrock

Función de prueba	Dim(D)	Región	mín
$f_1(x) = \sum_{i=1}^{D} x_i^2$	2	$[-5,12, 5,12]^D$	0
$f_2(x) = \sum_{i=1}^{n} 100(x_i^2 - x_{i+1})^2 + (1 - x_i^2)^2$	2	$[-5,12, 5,12]^D$	0
$f_3(x) = -20 \cdot exp(-0,2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}) - exp(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(2\pi x_i)) + 20 + exp(1)$	2	$[-30, 30]^D$	0
$f_4(x) = \sum_{i=1}^{D} (\lfloor x_i + 0,5 \rfloor)^2$	2	$[-5,12, 5,12]^D$	0
$f_5(x) = \sum_{i=1}^{D} i(x_i^4) + random[0, 1)$	2	$[-1,28, 1,28]^D$	0
$\frac{1}{f_6(x)} = \frac{1}{500} \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}$	2	$[-65,536, 65,356]^D$	≈ 1

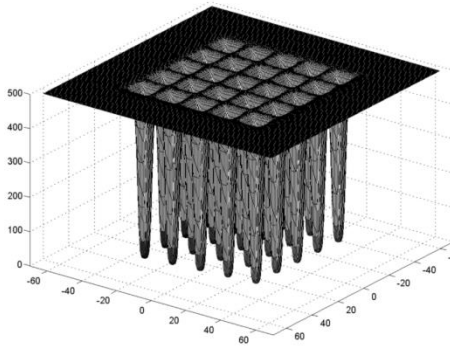
(c) f_3: Función Ackley



(d) f_4: Función Step



(e) f_5: Función Cuadrática ruidosa



(f) f_6: Función Foxholes

Condiciones de optimalidad

Sea el problema a Optimizar $f(\mathbf{x})=f(x_1,x_2,\ldots,x_n)$

La solución óptima $\mathbf{x}^*$ habrá de cumplir la condición de estacionaridad:

$$\nabla f(\mathbf{x}^*) = 0$$

que es una condición necesaria pero no suficiente.

Y que la matriz Hessiana sea definida positiva (para mínimo) o negativa (para máximo)

$$H \quad f(x_1, x_2, \ldots, x_n) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

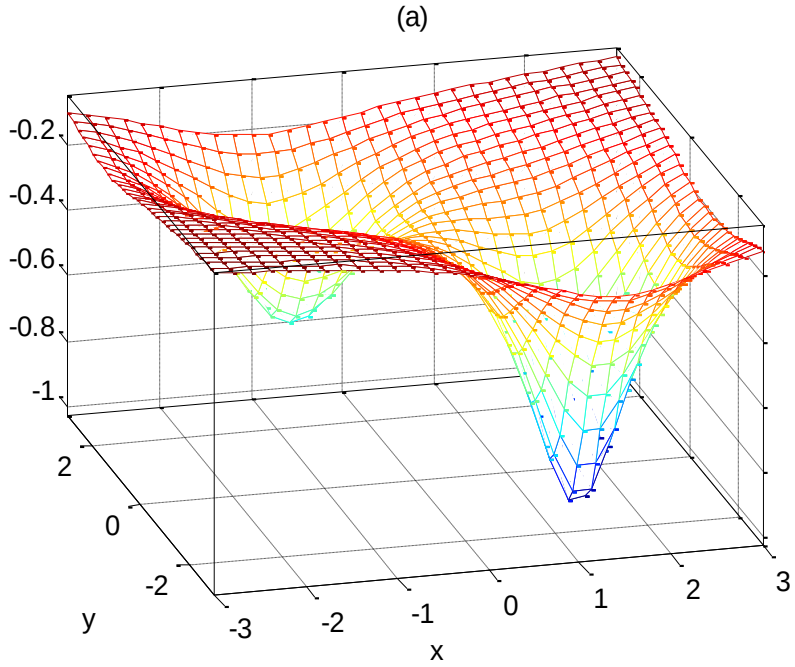
Métodos de optimización basados en gradiente

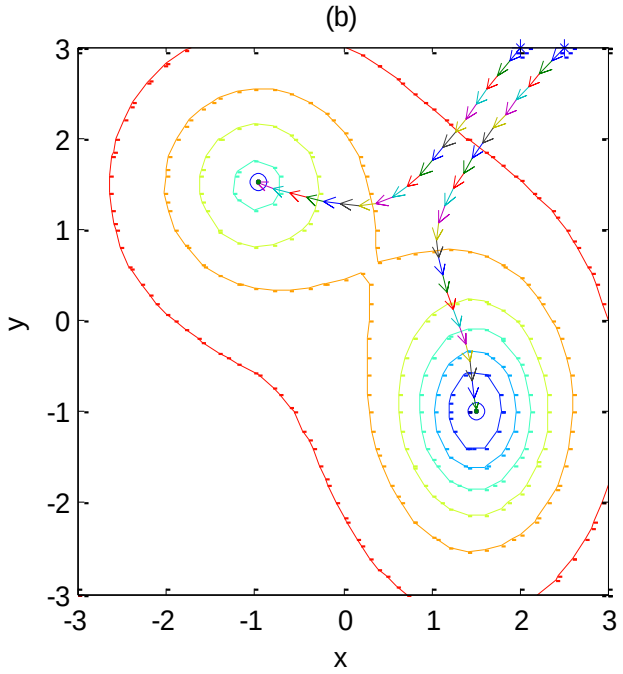
Se basan en la primera o en la segunda derivada de la función objetivo.

Comienzan en un punto cualquiera y se mueven de forma iterativa en direcciones que estiman convenientes para la minimización de f(**x**), según el gradiente en ese punto, hasta que se llegue a una solución que cumpla las condiciones de optimalidad.

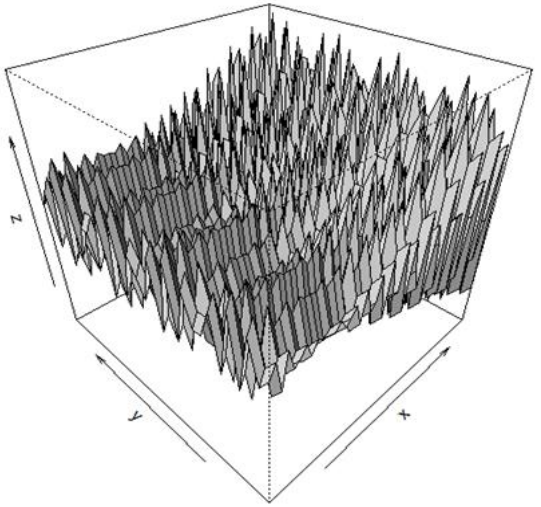
Falencias de los métodos basados en gradientes

- Sensibilidad al punto de inicio:



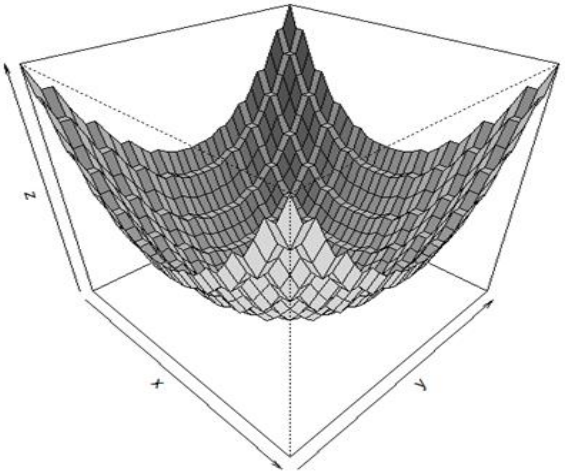


$f(x,y) = 21.5 + x\sin(4\pi x) + y\cos(20\pi y)$

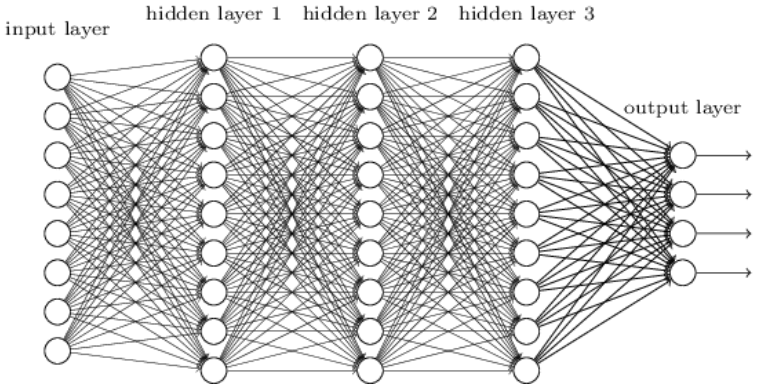


Multimodalidad

$f(x,y) = \lfloor x + 0.5 \rfloor^2 + \lfloor y + 0.5 \rfloor^2$



puntos no derivables.



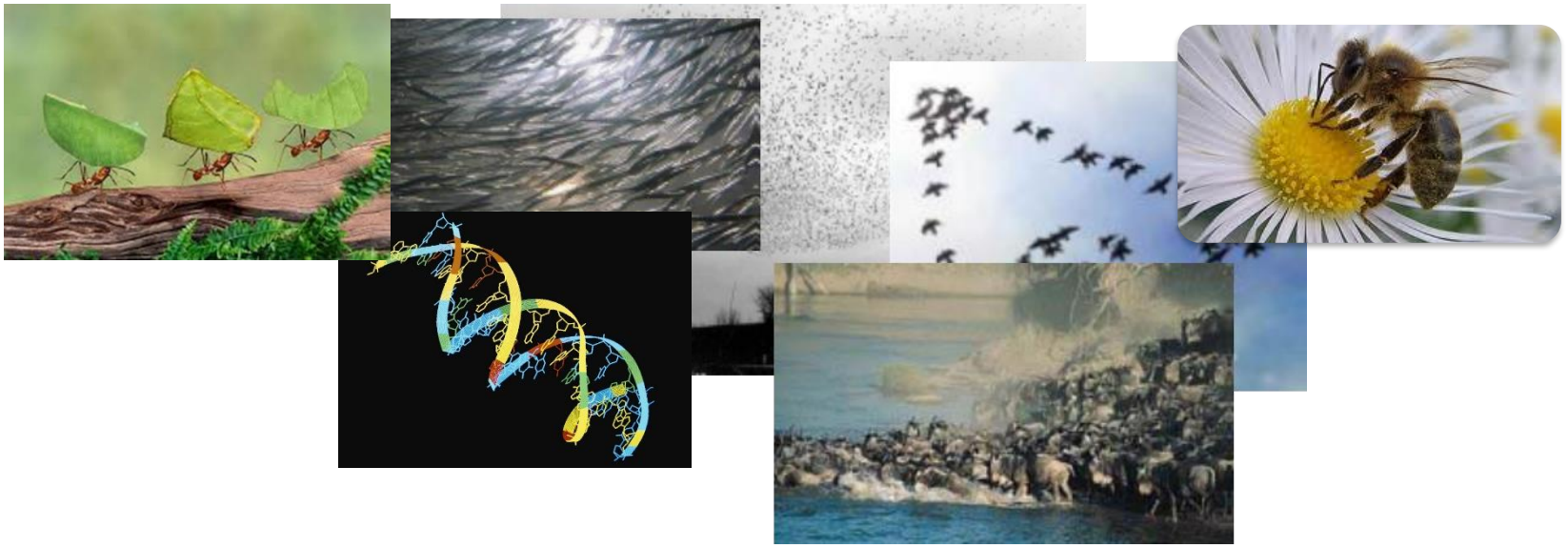
Alto número de variables

Por lo tanto, los métodos basados en gradientes no son viables para un gran número de problemas-

Otras posibles soluciones en caso de que estas fallen:

Metaheurísticas

Computación evolutiva y otros Algoritmos Bioinspirados



Pensemos: cómo encontrar el punto mas profundo en el fondo de un lago

Niveles del fondo = función desconocida

Los sistemas naturales son complejos:

- Tienen elementos que interactúan entre sí.
- Tienen propiedades emergentes: a partir de comportamientos simples de los elementos, se obtienen propiedades mucho más sofisticadas en el sistema.
- Se autoorganizan: a partir de una condición inicial desordenada, el sistema se ordena, espontáneamente.

Algoritmos bioinspirados

• Son metaheuristicas basados en principios generales de Inteligencia Artificial (IA).

•Se inspiran en procesos naturales.

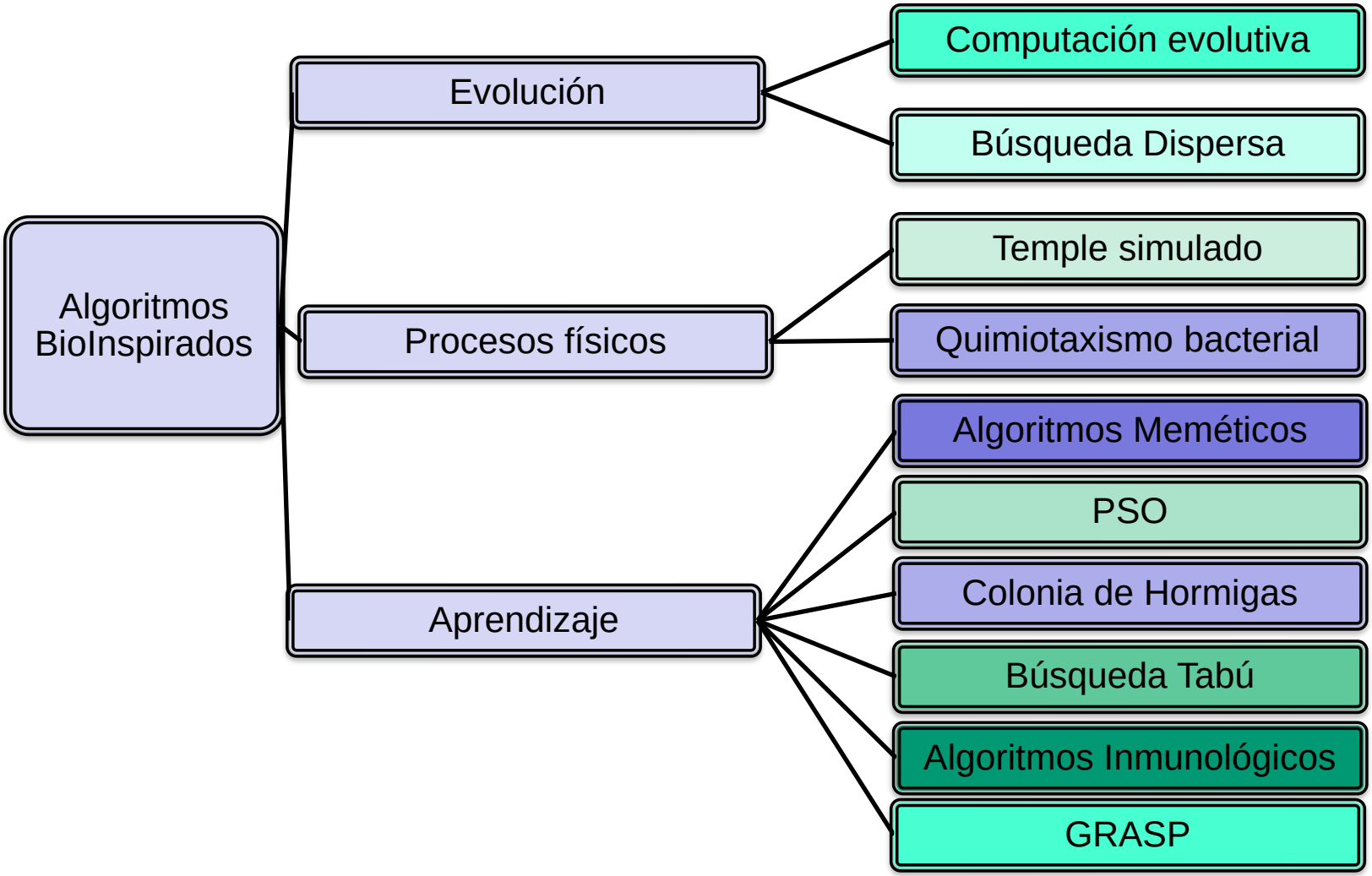
• Su propósito es encontrar una solución cercana al óptimo en un tiempo razonable.

•No garantizan la optimalidad de la solución encontrada

Principales Estrategias de los algoritmos

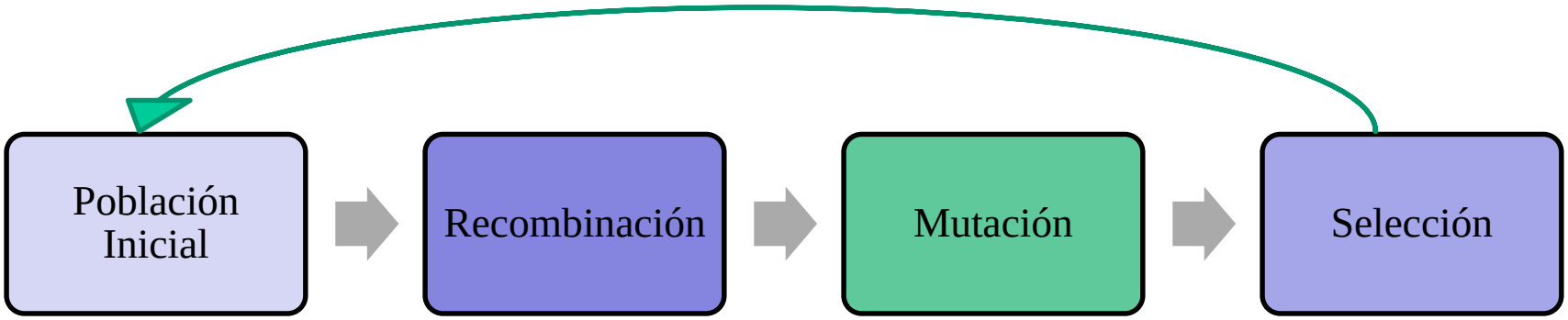
- Cooperación
- Uso de memoria
- Adaptabilidad y aprendizaje
- Balance entre explotación y exploración
- Búsqueda poblacional

Según su Inspiración pueden clasificarse en:

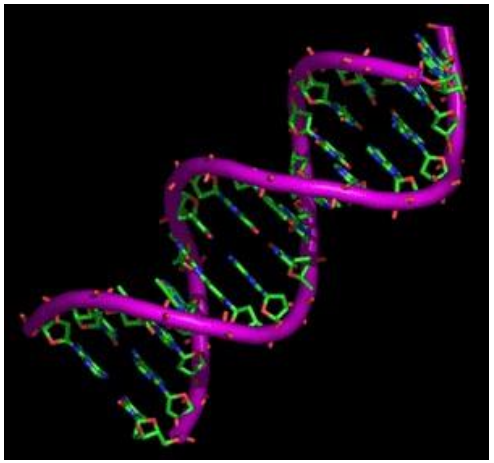


Algunos de ellos

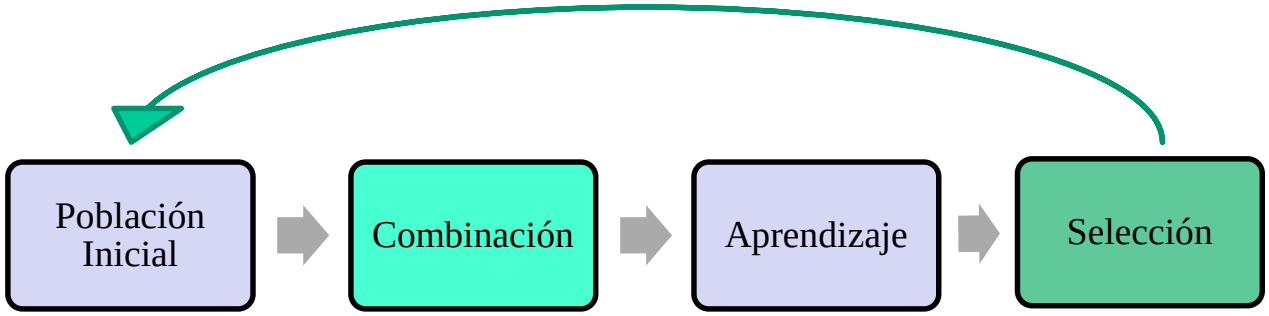
Inspirados en la evolución



- ***Computación evolutiva:*** algunos son:
  - Estrategias Evolutivas
  - Algoritmos genéticos
  - Evolución diferencial
- ***Búsqueda dispersa***



Inspirados en aprendizaje

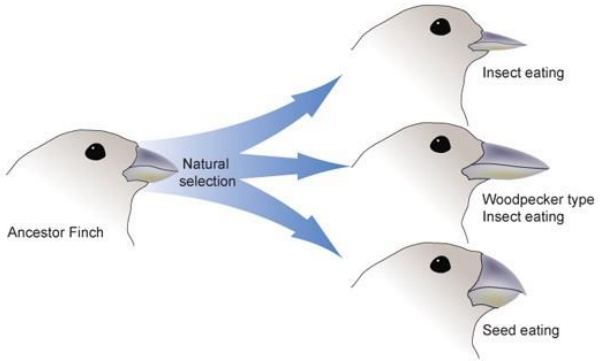


- PSO
- Colonia de Hormigas
- Búsqueda Tabú
- Algoritmos Inmunológicos
- GRASP

Algoritmos evolutivos

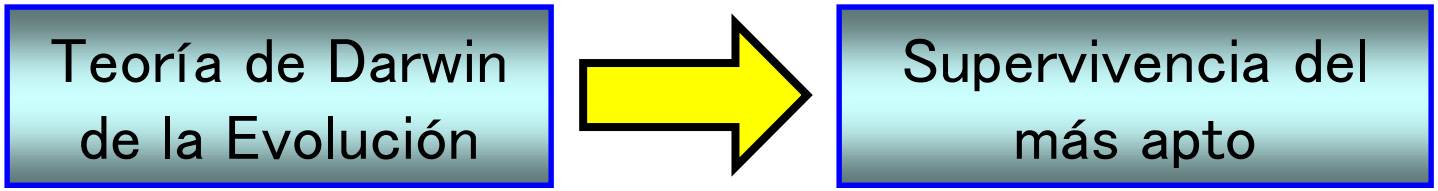
Emulan el mecanismo de la **selección natural en los seres vivos:**

Buenas características se **propagan** y se mejoran de generación en generación. Malas características desaparecen genéticamente.



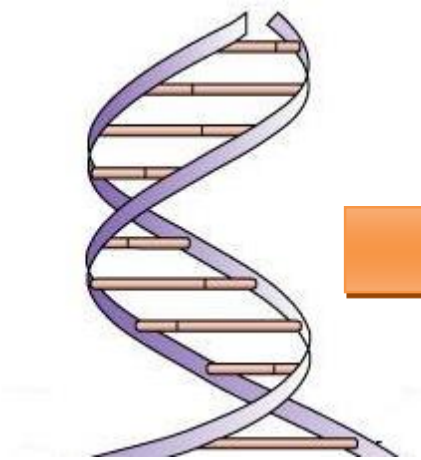
Antecedentes Históricos

✓**Darwin** enunció su teoría en 1858, que indica que la evolución se origina a través de **cambios** aleatorios de características hereditarias, combinados con un proceso de *selección natural*.



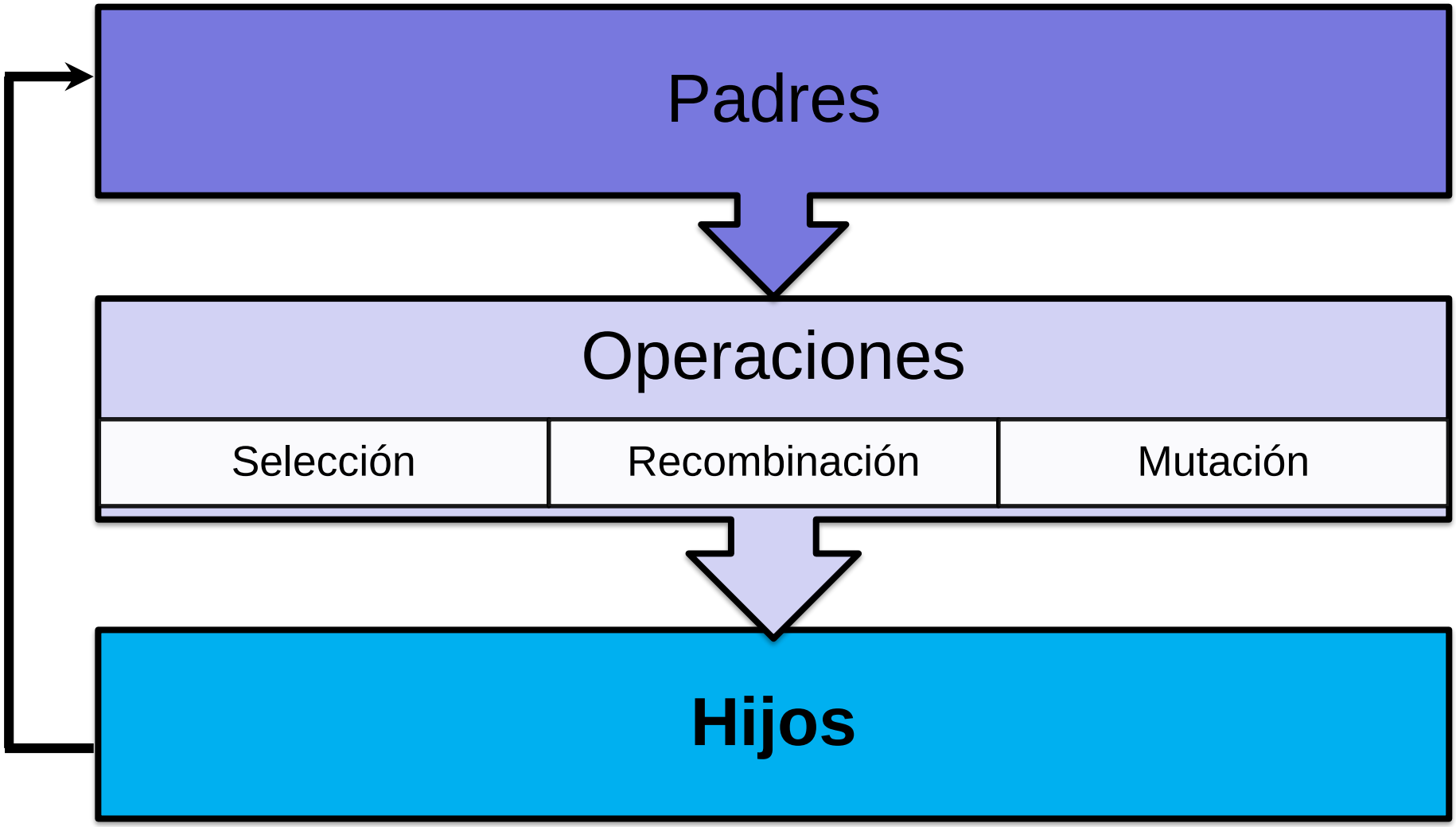
Genotipo

Fenotipo



Solución → **Valor de adaptación**

Mecanismo de evolución



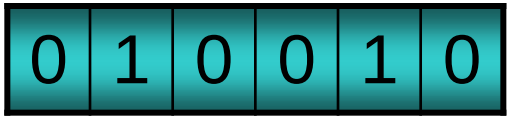
Algoritmos Genéticos

Desarrollados por John H. Holland como un proceso adaptativo

(Universidad de Michigan - 1975)

.

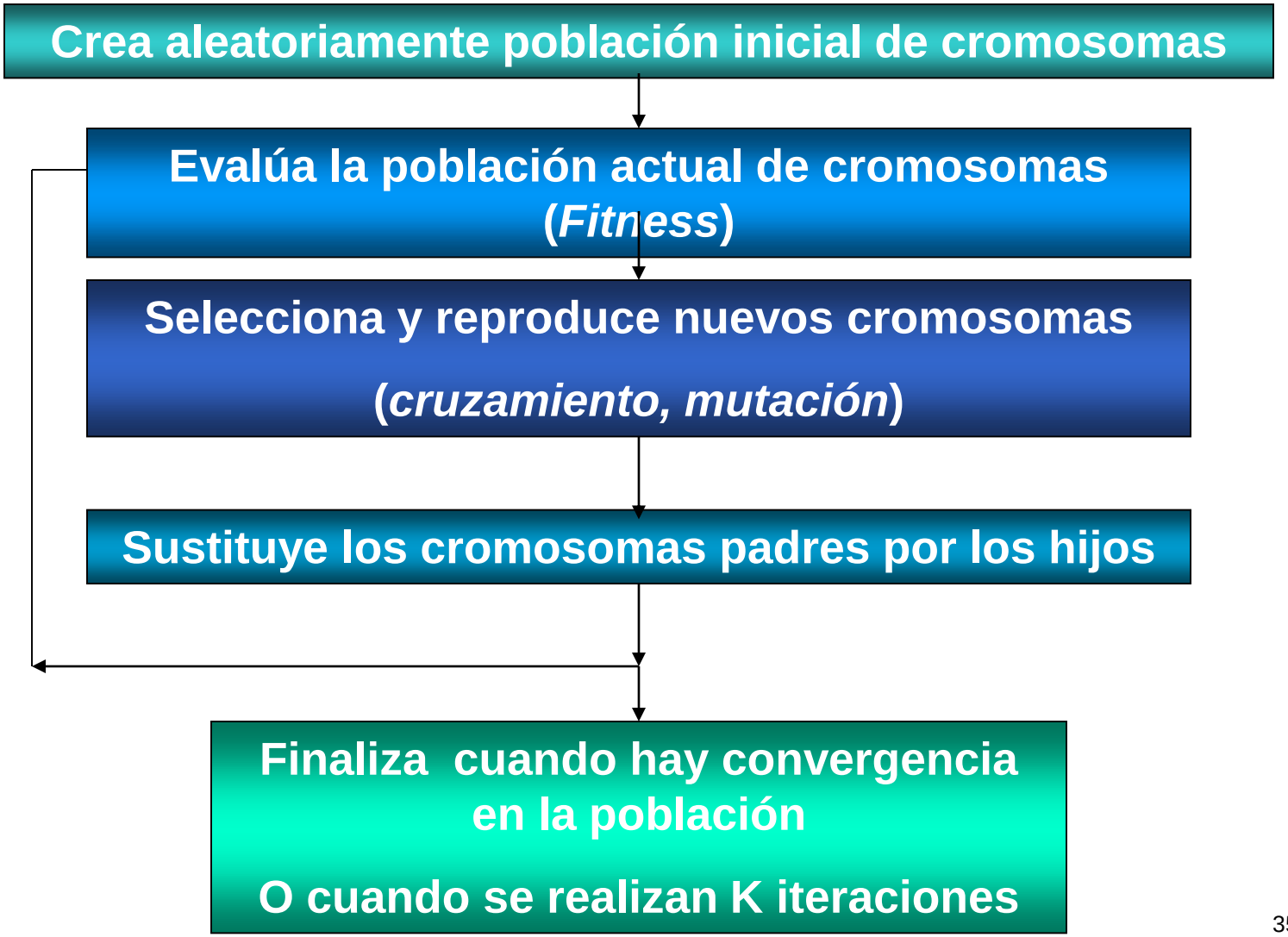
1. Genera una población de posibles soluciones: Cada solución se codifica como una serie de parámetros en un alfabeto dado, usualmente el binario



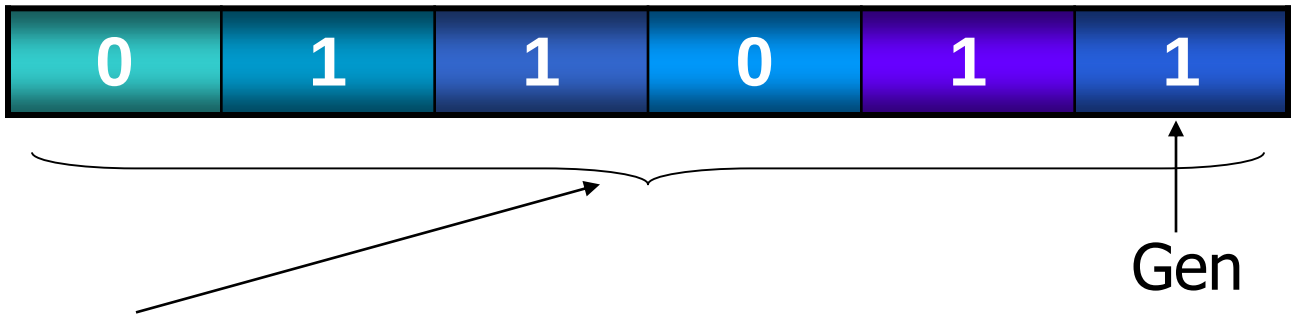
2. Identifica en las diferentes soluciones de la población actual las mejores soluciones (con el fin de capturar las mejores características).

3. Genera nuevas soluciones propagando, mediante cruzamiento y mutación de las actuales, las mejores características de generación en generación. Malas características desaparecen genéticamente.

Esquema Básico de los AG



A.¿Cómo representa cada solución?

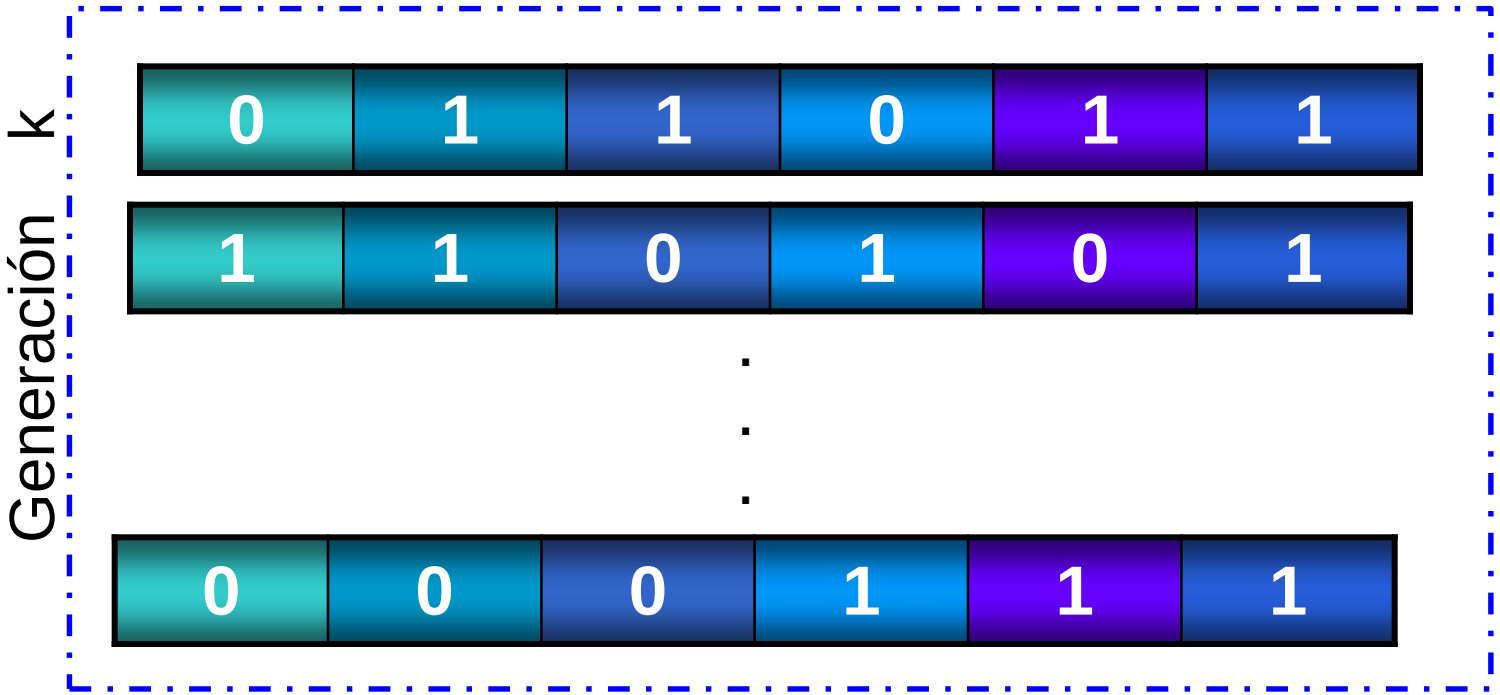


Gen: representa a cada variable de decisión.

Alelo: es el valor que toma el gen

Cadena de genes (individuo, cromosoma): representa una solución: un conjunto de posibles valores de las variables de decisión. Los genes controlan la herencia de las características de las cadenas

B.¿Cómo representa una generación?



Una generación es un población de individuos.

La inicial es generada aleatoriamente

Las demás surgen a partir de operaciones genéticas sobre la población anterior.

C.¿Cómo evalúa cada solución?

Mediante la función de <u>Aptitud</u> (*Fitness*):

Mide el desempeño de una solución respecto al Objetivo. Puede ser:

- la función objetivo o un escalamiento de la misma.
- Un modelo de simulación
- Un modelo externo al algoritmo
- El comportamiento de un dispositivo

Por ejemplo $F= 4x_1^2 + 6(x_2+x_3)^{1/3}+ x_4+ x_5x_6$

D. Cómo se hace la selección?

Individuos son seleccionados según su aptitud.

Las más aptas tienen mayor probabilidad de contribuir genéticamente en la siguiente generación.

Puede ser por:

- **Ruleta:** Simulación de montecarlo. Pr según aptitud
- **Torneo:** selecciona aleatoriamente varios individuos y escoge uno pora ptitud

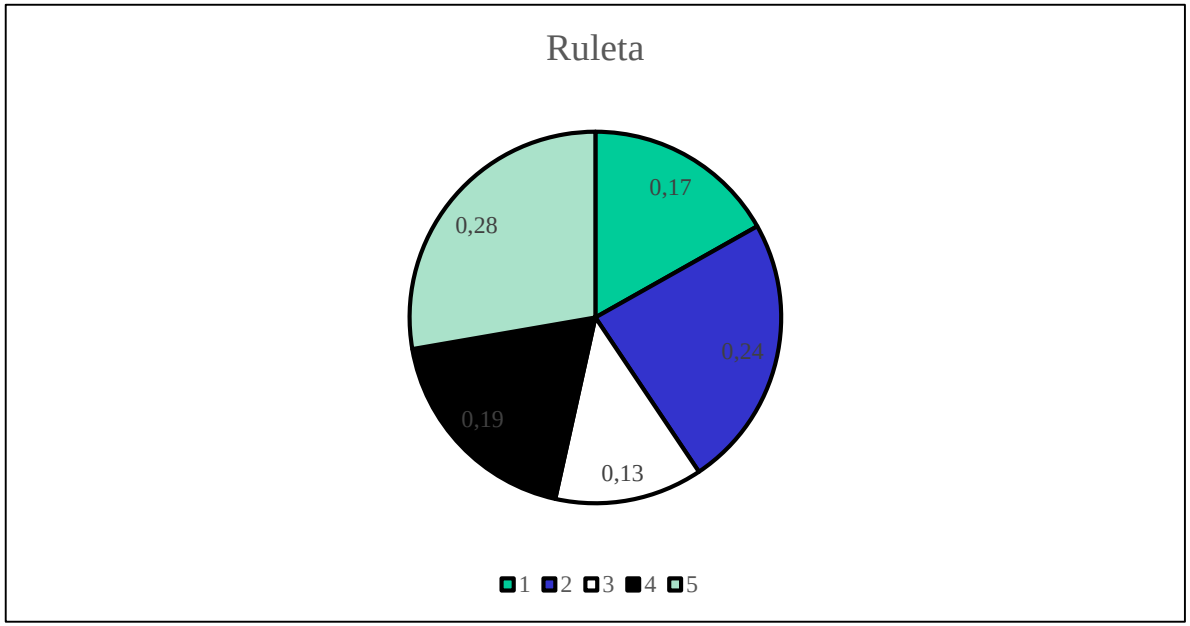
Ejemplo: Suponga la siguiente población inicial aleatoria y su medida de aptitud

0	1	1	0	1	0	F=7.55
1	0	1	0	1	1	F=11
0	1	0	0	1	0	F=6
0	1	1	0	1	1	F=8.56
1	1	1	0	1	1	F=12.56

Por ejemplo: selección Por Ruleta:

individuo	Aptitud	Prob. Ps	Prob. acumulada
1	7.55	0.17	0.16
2	11.00	0.24	0.40
3	6.00	0.13	0.53
4	8.56	0.19	0.72
5	12.56	0.28	1.00
Total	45.67	1.00	

$$P_S(a_i(t)) = \frac{F(a_i(t))}{\sum_{j=1}^{\mu} F(a_j(t))}$$



E. Cómo genera nuevos individuos para la próxima generación?

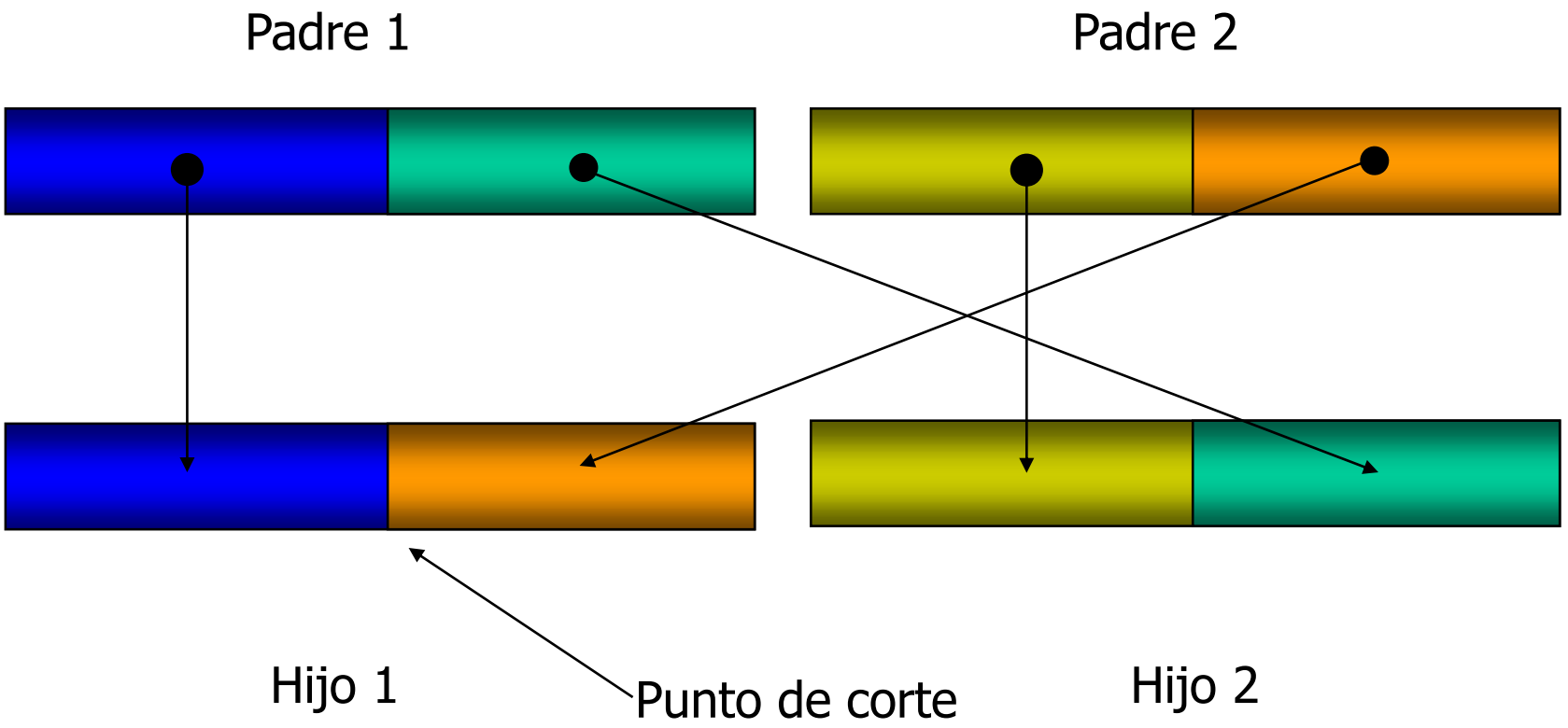
Mediante Operadores genéticos.

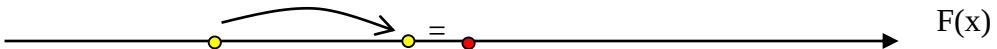
Los más usados son:

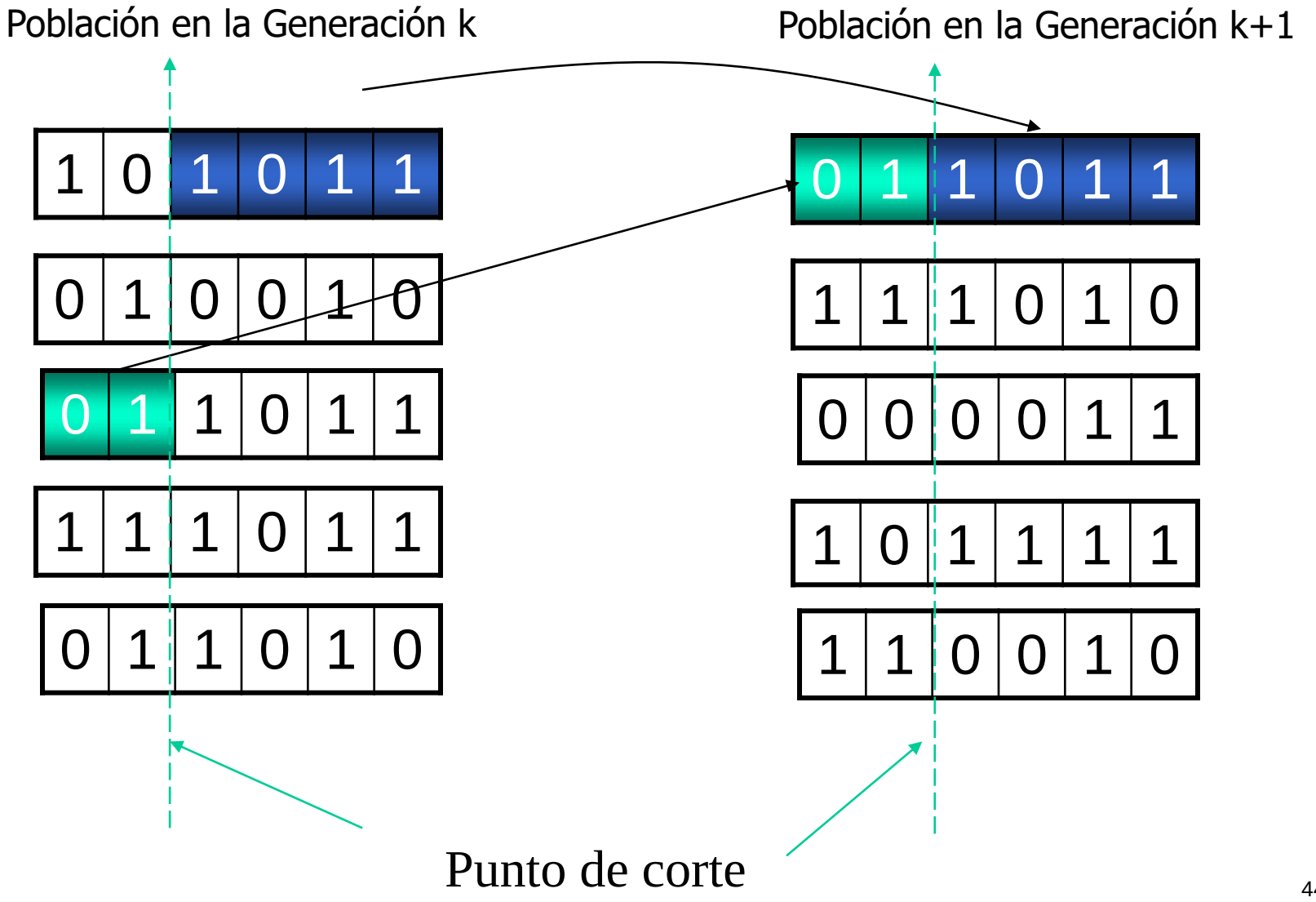
- cruzamiento
- mutación

Cruzamiento: a partir del cruce de dos individuos seleccionados de la generación anterior (padres), genera dos individuos para la nueva población:

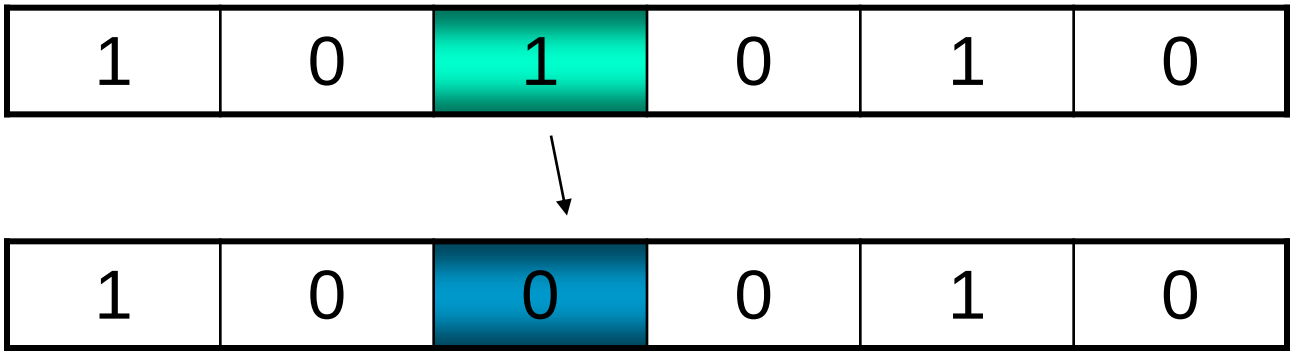
- Determina una posición de cruce aleatoria e intercambia las cadenas entre la posición inicial y el punto de cruce y el punto de cruce y la posición final






Población en la Generación k
Población en la Generación k+1
1 0 1 0 1 1
0 1 0 0 1 0
0 1 1 0 1 1
1 1 1 0 1 1
0 1 1 0 1 0
0 1 1 0 1 1
1 1 1 0 1 0
0 0 0 0 1 1
1 0 1 1 1 1
1 1 0 0 1 0
Punto de corte

Mutación: determina una posición de un gen aleatoriamente e invierte su valor.



•Permite introducir nueva información no presente en la población (ruido)

•Permite salir de máximos locales (exploración).

▪No se utiliza siempre, sino con una baja probabilidad (del orden de 5% o menor)

G. Se generan tantas cadenas genéticas como se requieran para completar una nueva población:

La nueva población supere estadísticamente a la anterior (aptitud promedio cada vez mejor)

H. Estos pasos se repiten **iterativamente** hasta que la población actual converja (todas las cadenas de la población tengan igual función de aptitud). Esa es la solución óptima.

En la práctica, se define un número fijo de iteraciones N y la solución óptima (en realidad seudo-óptima) es la correspondiente a la cadena con mejor aptitud.

Ejemplo sencillo:

Se consideran *6* productos para ofrecer en una estantería, con un volumen limitado V, cuya demanda depende del conjunto que se ofrezca, detectado por un análisis de canasta de compra.

Cada producto *j* se caracteriza por:

- G_j beneficio neto del producto i
- D_{ji} demanda del producto j si se ofrece en conjunto con el producto i
- Vj Volumen de producto j

Se desea Maximizar las ganancias

Max $Z(\mathbf{x})$= sum(j, $G_j \times$ sum(i,$D_j(X_j\ X_i)$))

Restricción

sum(j, $V_j\ X_j$) <= V

$$x_j = (0,1)$$

Elección de Parámetros

Por ejemplo:

No de Iteraciones 20

Población: 20

Probabilidad de cruce: 0.98

Probabilidad de mutación: 0.01

Población inicial

i	x_1	x_2	x_3	x_4	x_5	x_6	Ganancias	Prob.	Prob. Acumulada
1	0	0	1	1	1	0	6083	0.065	0.065
2	1	0	0	1	0	1	4417	0.045	0.110
3	1	0	0	1	0	1	4417	0.045	0.156
4	1	1	0	1	0	0	4135	0.044	0.200
5	0	1	0	0	1	1	5793	0.060	0.260
6	1	0	1	0	0	0	2409	0.026	0.286
7	0	0	0	0	1	0	2075	0.022	0.308
8	0	0	1	1	0	1	6008	0.062	0.370
9	1	1	1	1	0	0	6135	0.066	0.436
10	1	1	1	0	0	0	4127	0.044	0.480
11	0	0	1	1	0	0	4008	0.043	0.523
12	0	1	1	0	1	1	7793	0.081	0.605
13	0	0	0	0	0	1	2000	0.019	0.624
14	0	1	0	1	0	1	5726	0.059	0.683
15	0	0	0	1	0	1	4008	0.041	0.724
16	1	1	1	0	0	1	6127	0.064	0.788
17	1	0	1	1	0	1	6417	0.067	0.854
18	0	0	1	0	1	0	4075	0.044	0.898
19	0	1	0	1	1	0	5801	0.062	0.960
20	0	1	1	0	0	0	3718	0.040	1.000

Selección de padres:

aleatorio: 0.56, Alternativa 12

0	1	1	0	1	1

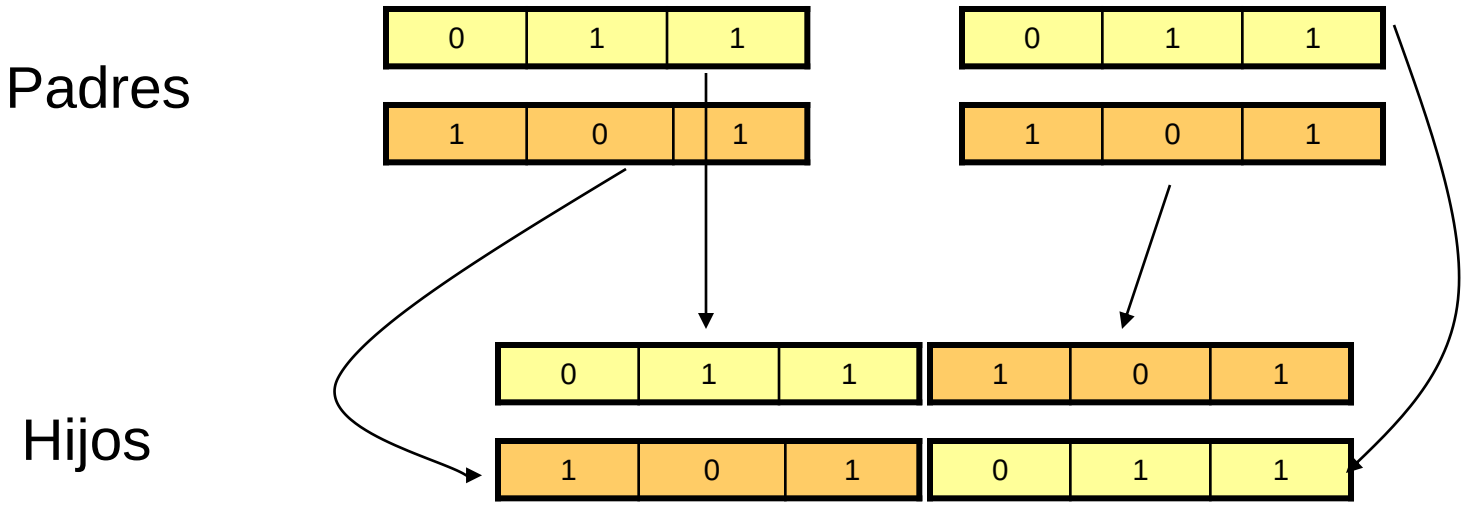
aleatorio: 0.81, Alternativa 17

1	0	1	1	0	1

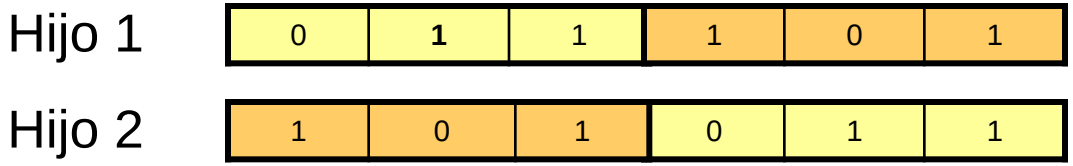
aleatorio: 0.86 < 0.98= ¡¡¡se cruzan!!!

Cada posible corte tiene igual probabilidad, 1/5

aleatorio: 0.51. $\Rightarrow$ Punto de cruce: 3



Mutación:

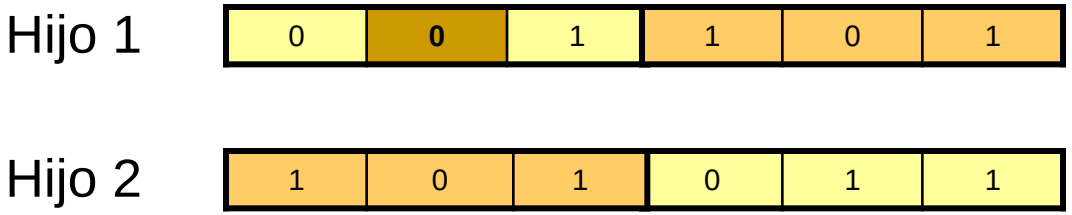


Para el Hijo 1

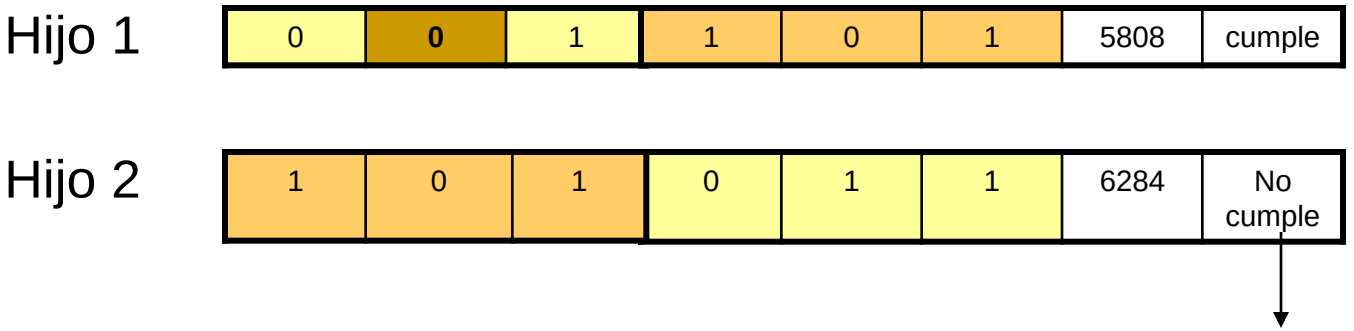
Gen 1: # aleatorio 0.58>0.01, no hay mutación

Gen 2: # aleatorio 0.005<0.01, ¡¡¡ hay mutación!!!.....

Supóngase que no se definieron más mutaciones:



Verifica si cumplen las restricciones:



Igualmente, a partir de la selección de padres y creación de hijos (cruzamiento y mutación) se genera una nueva población de 20 cadenas o individuos

Se repite el proceso durante 20 iteraciones (o cuando la solución converja). En la última iteración la mejor solución fue:

0	0	1	1	1	1	7883	Cumple

Es decir, se ofrecen los productos 3, 4, 5, 6

Ventajas de AG

- Gran habilidad para tratar con problemas muy complejos. Permite funciones objetivo altamente no lineales e incluso modelos de simulación o un modelo externo.

Busca en una población y no en un único punto. Esto incrementa la probabilidad de que no quede atrapado en un óptimo local.

- Usa reglas probabilísticas en vez de determinísticas como la mayoría de métodos tradicionales

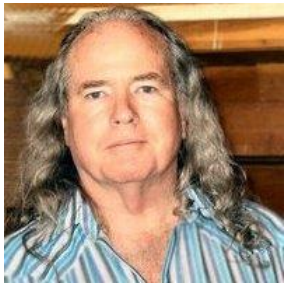
Otras técnicas

Optimización por Nubes de partículas

(Particle Swarm optimización PSO)

Optimización por Nubes de partículas PSO

Es una simple pero poderosa Metaheurística desarrollada por James Kennedy (psicólogo social) y Russ Eberhart (ingeniero eléctrico) en 1995



[Kennedy, J. and Eberhart, R. (1995). "Particle Swarm Optimization", *Proceedings of the 1995 IEEE International Conference on Neural Networks*, pp. 1942-1948, IEEE Press.]

Inspiración

interacción social en las manadas de animales: pájaros, peses insectos, búfalos, los cuáles, intercambiando información, buscan lo mejor para el grupo social.

Las sociedades de animales simples tienen un comportamiento emergente…

- La estructura global (nido) emerge desde las acciones independientes de los animales.

- actuaciones de nivel complejo de forma inteligente, flexible y fiable- Estas actuaciones no serían factibles si tuviesen que ser realizadas por un animal de forma individual (simples y no inteligentes).

- Siguen reglas y comunicación local simple. El resultado es una inteligencia de manada: Swarm Intelligence (**SI**)

Teoría de la Disonancia cognitiva

Existen dos mayores fuentes de cognición: la propia experiencia y la comunicación con otros.

Leon Festinger, 1999, Social Communication and cognition

Algoritmo PSO

•PSO asume que cada posible solución (x_1,x_2,…,X_n) es una partícula que tiene una posición en el espacio de búsqueda .

•Las Partículas son agentes que se mueven a través del espacio y almacenan (y posiblemente comunican) la mejor solución que han descubierto.

•La pregunta es, “¿Cómo se mueve una partícula desde una localización a otra del espacio de búsqueda?

Videos interesantes

https://www.youtube.com/watch?v=gkGa6WZpcQg

https://www.youtube.com/watch?v=_bzRHqmpwvo

Aprendizaje

En una manada, el espacio socio cognitivo puede contener algunos individuos que influyen unos a otros.

Así, cada individuo sigue los pasos de:

- Su propios éxitos (Self best)
- Del líder (el más exitoso) de la manada (Global Best)

Representación de una partícula

4 vectores:

- **x(t)** posición actual de la partícula
- **p(t)** localización de la mejor localización encontrada hasta el momento por la partícula.
- $\mathbf{p_g}$ localización de la mejor partícula de todo el grupo encontrada hasta el momento.
- **v(t)** gradiente (dirección) por el cual la partícula viajaría si no fuera perturbada, depende de la su posición en t-1.

– **3 valores de la función objetivo (aptitud):**

- **x-fitness** aptitud del vector x(t)
- **p-fitness** aptitud del vector p(t)
- **pg-fitness aptitud del vector pg**

inercia

nostalgia

$$\vec{v}_i(t+1) \leftarrow \vec{v}_i(t) + \phi_1 rnd \otimes (\vec{p}_i(t) - \vec{x}_i(t)) + \phi_2 rnd \otimes (\vec{p}_g - \vec{x}_i(t))$$

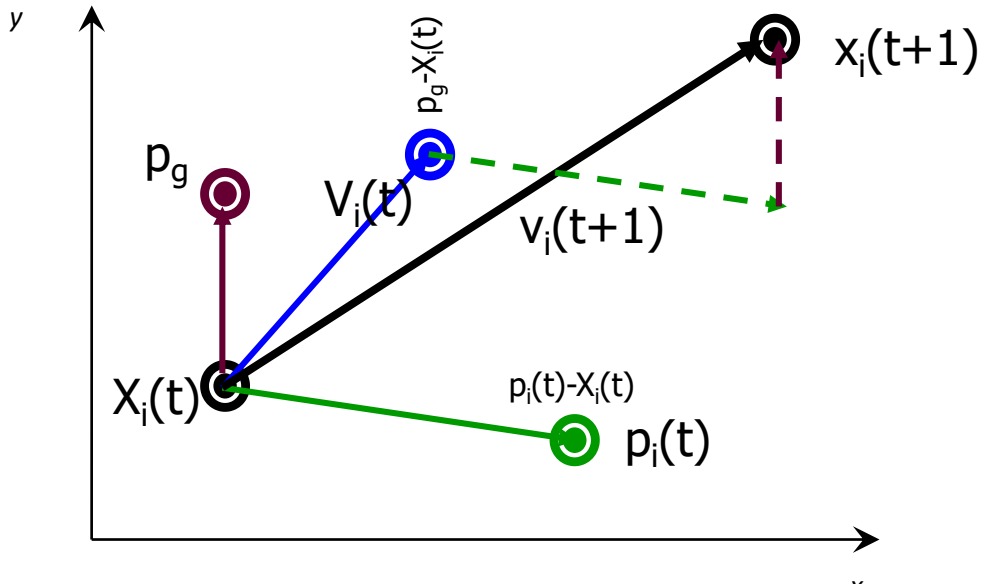
$$\vec{x}_i(t+1) \leftarrow \vec{x}_i(t) + \vec{v}_i(t+1)$$

imitación al lider

ϕ_1 y ϕ_2 : tasa de aprendizaje de los componentes de nostalgia y de imitación al líder

rnd es un número aleatorio uniforme en el rango [0,1].

representa tasa de aprendizaje que gobierna la componente social. (típicamente ϕ_1 y ϕ_2 =2)



Colonia de Hormigas

ACO (Ant Colony Optimization)

https://www.youtube.com/watch?v=eVKAIufSrHs

Colonia de Hormigas ACO

diseñada por Dorigo (1992) del Politécnico de Milan.

Se basa en la forma como las hormigas buscan el mejor recorrido desde su nido hasta el sitio en el que se encuentra su alimento.

Las hormigas reales

- Son capaces de seguir la ruta más corta en su camino de ida y vuelta entre la colonia y una fuente de abastecimiento.

- "Transmiten información": cada una, al desplazarse, deja un rastro de feromona.

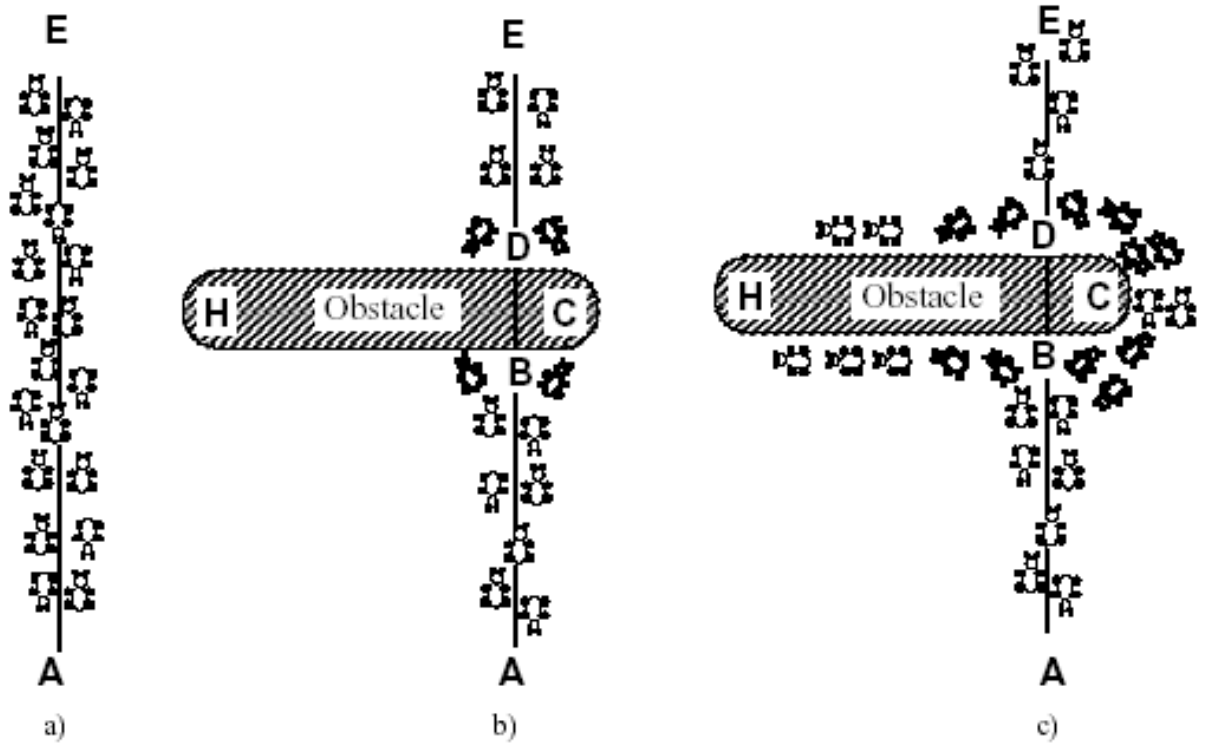
- Una hormiga detecta el rastro de feromona dejado por otras y tiende a seguirlo. Éstas, a su vez, van dejando su propia feromona a lo largo del camino recorrido y lo hacen más atractivo.

- Algunas seleccionan caminos con menor feromona, así se abre la posibilidad de encontrar caminos alternativos a la fuente de alimento (exploración).

- Esto conduce a un comportamiento emergente a partir de interaccione simples entre hormigas, sin siquiera contar con un líder.

- La feromona también se va evaporando con el paso del tiempo.

- La evaporación se nota más en los caminos más largos al tener menos feromona (reducción proporcional) y así se elimina progresivamente la aceptación de las hormigas en elegir estos caminos y aumentar la preferencia en otros de mejores características (exploración).



Un ejemplo con hormigas reales:

a) las hormigas siguen una ruta entre los puntos A y E

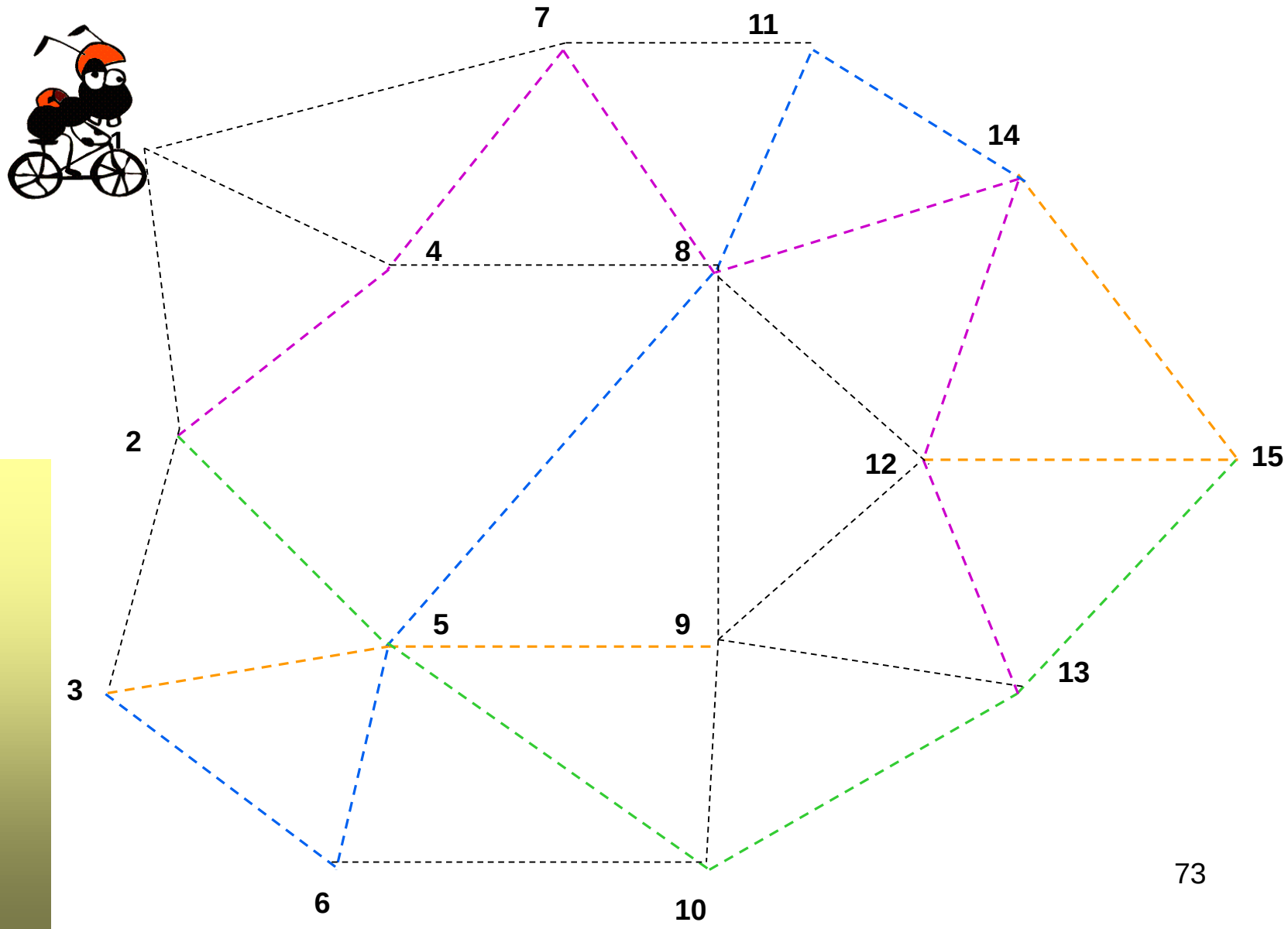
b) Se interpone un obstáculo. Las hormigas pueden elegir una ruta siguiendo uno de los dos caminos posibles con igual probabilidad.

c) Al cabo del tiempo en el camino mas corto habrá mas traza de feromona, por lo que más hormigas tienden a seguirlo.

Las hormigas virtuales

- Son “agentes” con capacidades básicas muy simples
- Tienen algo de memoria.
- Viven en un ambiente donde el tiempo es discreto.

Ejemplo: problema del agente viajero TSP



Cómo Funciona (1)

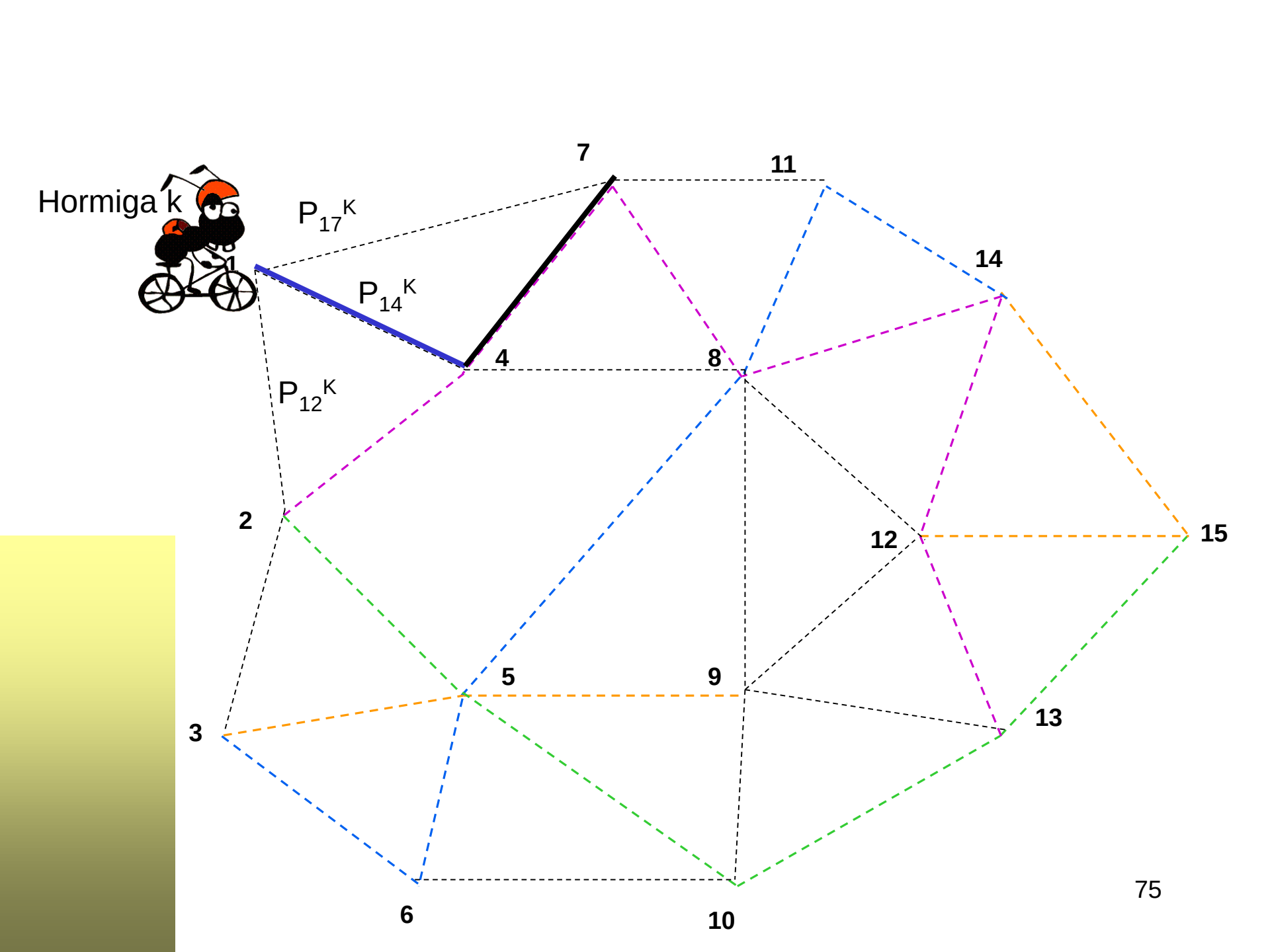
- Se lanzan m hormigas a recorrer la red saliendo del nodo origen. Cada hormiga busca su propio recorrido.

Así, cada hormiga k:

- cuando está en un nodo i, pasa a otro nodo j según una probabilidad definida P^k_{ij}, que depende de la distancia d_{ij} y de la cantidad de rastro de "feromona" presente en el arco ij.

- va dejando rastro de feromona por los arcos que pasa (si se considera actualización local)

- Para no hacer ciclos y hacer siempre opciones factibles, guarda en "memoria" los nodos que ha visitado y prohíbe volver a ellos (hasta que un viaje es completado -1 ciclo-). Esto es controlado por una lista tabú).

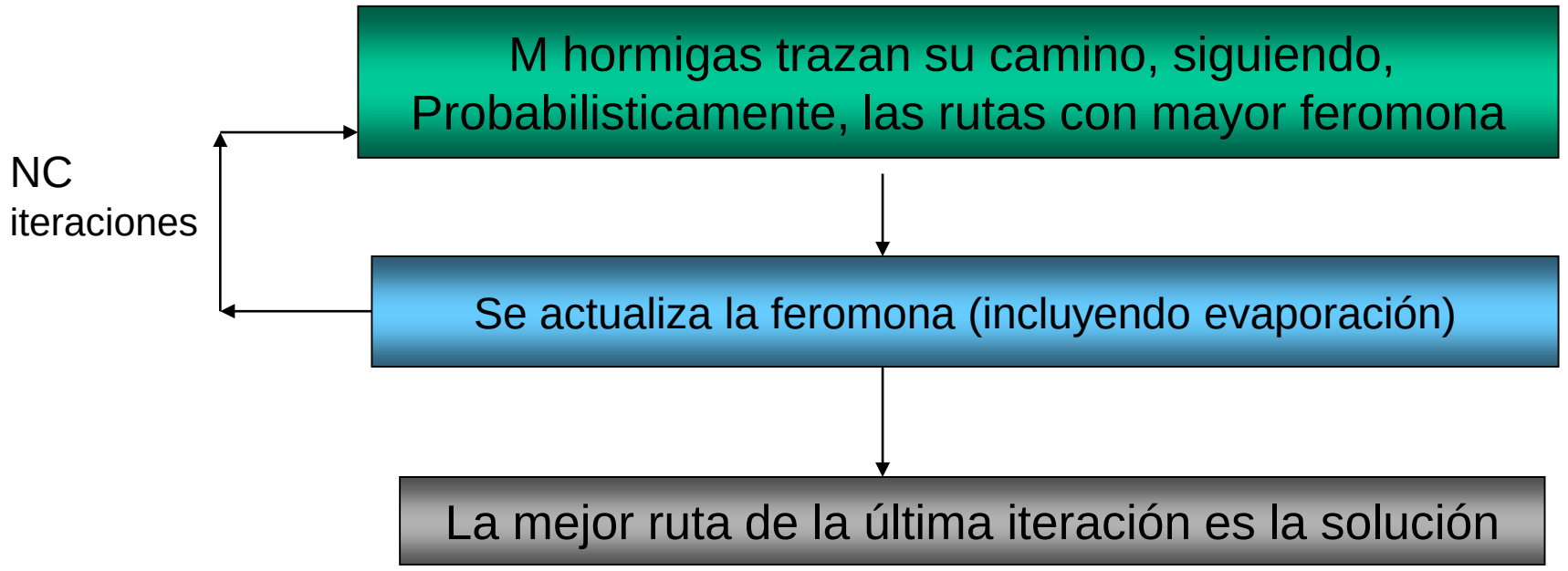

Hormiga k
P_{17}^{K}
P_{14}^{K}
P_{12}^{K}
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Cómo funciona (2)

- Una vez se termina un ciclo, se calcula la ruta más corta encontrada por alguna de las hormigas. Se agrega más feromona a un grupo de rutas encontradas entre todas las hormigas. (Puede ser a algunas o sólo a la mejor).

- Se repite el proceso en repetidas iteraciones hasta alcanzar el máximo de iteraciones o llegar a una “convergencia” del algoritmo

Proceso ACO


M hormigas trazan su camino, siguiendo,
Probabilisticamente, las rutas con mayor feromona
NC
iteraciones
Se actualiza la feromona (incluyendo evaporación)
La mejor ruta de la última iteración es la solución

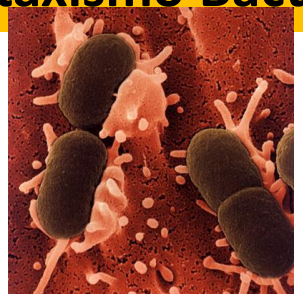
Ventajas de ACO

- Permite una retroalimentación positiva como mecanismo de búsqueda.
- Permite sinergia. la efectividad de la búsqueda es mayor por cooperación entre las hormigas.

Otros algoritmos

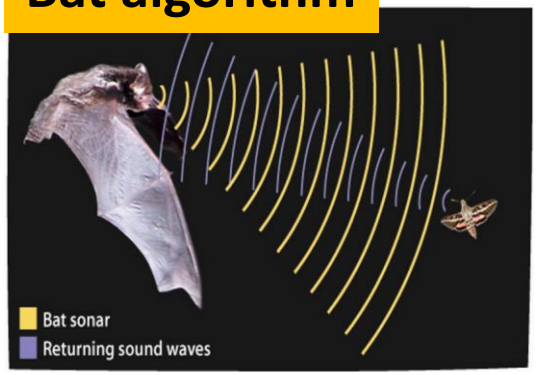
Harmony Search

Quimiotaxismo Bacterial QB

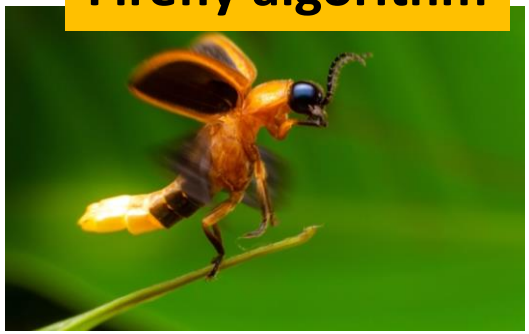


ABC
Artificia Bee Colony

Bat algorithm



Firefly algorithm



Tendencias Actuales de las MH

- Metaheuristicas Híbridas
- Coevolución
- Robótica evolutiva
- Nuevas metaheurísticas (evolución diferencial, harmony, bat algorithm, Bee algorithm etc.).
- Matheuristicas

Laboratorios

- Illinois Genetic Algorithms Laboratory –ILLiGAL (http://www-illigal.ge.uiuc.edu/about.html),

- Kanpur Genetic Algorithms Laboratory –KanGAL (http://www.iitk.ac.in/kangal/index.shtml)

- Evolutionary Computation Laboratory (George Mason University) –Eclab (http://cs.gmu.edu/ ~eclab/)

- The MathWorks-Genetic Algorithm and Direct Search Toolbox (http://www.mathworks.com/products/gads/)

Sitios web

- International Society for Genetic and Evolutionary Computation –ISGEC (http://www.isgec.org/)

- Special Interest Group for Genetic and Evolutionary Computation –SIGEVO (http://www.sigevo.org/)

- Genetic Algorithms Research at Colorado State University - The GENITOR Group (http://www.cs.colostate.edu/~genitor/)

- Machine Learning and Evolutionary Computation Group - M&EC University of Torino (http://www.di.unito.it/~mluser/)

Bibliografía

BRABAZON A., M. O'NEILL S. MC. GARRAGHY. (2015) Natural Computing Algorithms. - Springer. (ALLITEBOOKS)

EIBEN A.E. AND J.E. SMITH (2015) Introduction to Evolutionary Computing. 2 ed. Springer (ALLITEBOOKS)

SIMON., D. 2013. Evolutionary Optimization Algorithms. Wiley (ALLITEBOOKS)

ZÄPFEL G., R. BRAUNE AND M. BÖGL. (2010) Metaheuristic Search Concepts to Production and Logistics. A Tutorial with Applications to Production and Logistics. Ed. Springer.

Bibliografía

YANG X.S. (2010). Engineering Optimization An Introduction with Metaheuristics Applications, Hoboken: Wiley.

BACK T., D. B. FOGEL AND Z. MICHALEWICZ. (2000). Evolutionary Computation vol I Basic Algorithms and Operators. Institute of Physics Publishing, Bristol and Philadelphia.

BACK T., D. B. FOGEL AND Z. MICHALEWICZ. (2000). Evolutionary Computation Vol II Advanced algorithms and operators. Institute of Physics Publishing, Bristol and Philadelphia.

GENDREAU M. AND J. Y. POTVIN (2010). Handbook of metaheuristics, 2nd Edition, Ed. Springer.

BURKE E.K., G. KENDALL. (2005). Search methodologies: introductory tutorials in optimization and decision support techniques, Springer, New York.

SIVANANDAM S. N. AND S. N. DEEPA. (2008) Introduction to Genetic Algorithms. Ed. Springer.

YU X. AND G. MITSUO. (2010). Introduction to Evolutionary Algorithms. Ed. Springer

OLARIU S. AND ZOMAYA A. (eds) (2006). Handbook of inspired algorithms and applications. Chapman Hall/HCR.

MICHALEWICZ Z. AND D. B. FOGEL. (2000). How to solve it: Modern Heuristics. Ed. Springer

GLOVER F. AND KOCHENBERGER G. (2003). Handbook of Metaheurisctics. Ed. Kluwer.

TALBI E.G (2009). Metaheuristics: from design to implementation, Wiley.

Algunas revistas de interés:

- Computational Optimization and Applications (Kluwer)
- Journal of Heuristics (Kluwer).
- Journal of Combinatorial Optimization (Kluwer)
- INFORMS Journal on Computing (INFORMS)
- Evolutionary Computation (MIT press)
- European Journal of Operational Research (Elsevier)