



# DECSAI

Departamento de Ciencias de la Computación e I.A.

Universidad de Granada



## Optimización

© Fernando Berzal, [berzal@acm.org](mailto:berzal@acm.org)

## Optimización



### Metaheurísticas

#### Técnicas de búsqueda local

- Ascensión de colinas

#### Optimización basada en modelos naturales

- Enfriamiento simulado
- Modelos evolutivos
  - Algoritmos genéticos
- Modelos de adaptación social ("swarm intelligence")
  - Colonias de hormigas  
[ACO: Ant Colony Optimization]
  - Nubes de partículas  
[PSO: Particle Swarm Optimization]



# Metaheurísticas



Técnicas computacionales que resuelven iterativamente problemas de optimización, intentando mejorar una solución candidata con respecto a una medida de calidad dada @ <http://en.wikipedia.org/wiki/Metaheuristic>

- Las metaheurísticas no suelen realizar suposiciones acerca del problema de optimización y pueden explorar espacios de búsqueda muy grandes.
- Las metaheurísticas no garantizan que se encuentre una solución óptima.
- Muchas metaheurísticas implementan alguna forma de optimización estocástica.



# Metaheurísticas



## **Enfoque tradicional**

Algoritmos deterministas de búsqueda exhaustiva.

- Garantizan encontrar una solución óptima.
- Pueden no funcionar en situaciones reales :-)

## **Enfoque basado en metaheurísticas**

Algoritmos estocásticos de búsqueda heurística [guiada].

- No garantizan encontrar una solución óptima.
- Siempre proporcionan una solución :-)

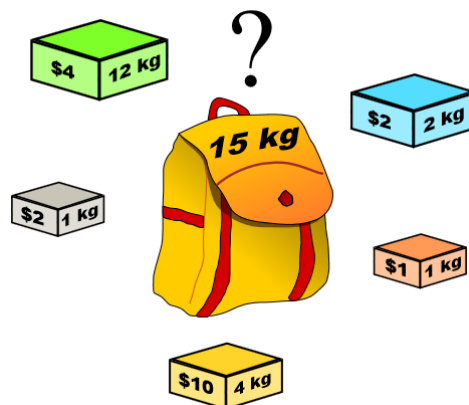




## Problemas de optimización combinatoria

(las metaheurísticas como alternativa al uso de técnicas exhaustivas del estilo de branch&bound)

- Problema de la mochila [knapsack problem],  $O(2^n)$   
[http://en.wikipedia.org/wiki/Knapsack\\_problem](http://en.wikipedia.org/wiki/Knapsack_problem)



## Problemas de optimización combinatoria

(las metaheurísticas como alternativa al uso de técnicas exhaustivas del estilo de branch&bound)

- Problema del viajante de comercio (TSP),  $O(n^2 2^n)$   
[http://en.wikipedia.org/wiki/Traveling\\_salesman\\_problem](http://en.wikipedia.org/wiki/Traveling_salesman_problem)

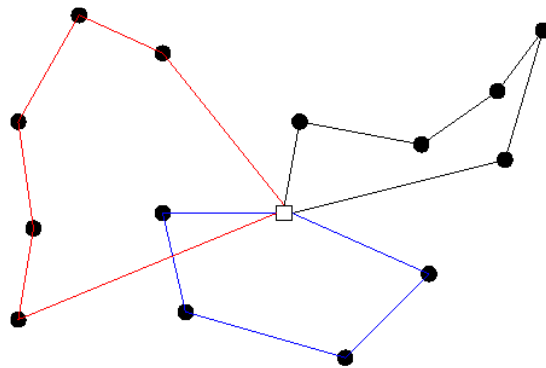




## Problemas de optimización combinatoria

(las metaheurísticas como alternativa al uso de técnicas exhaustivas del estilo de branch&bound)

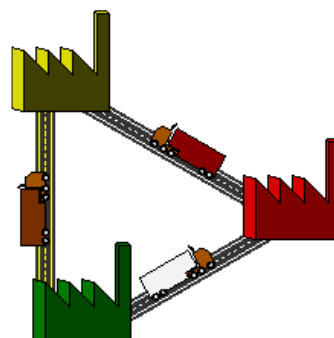
- Problema de la ruta de los vehículos (VRP),  $O(n!)$   
[http://en.wikipedia.org/wiki/Vehicle\\_routing\\_problem](http://en.wikipedia.org/wiki/Vehicle_routing_problem)



## Problemas de optimización combinatoria

(las metaheurísticas como alternativa al uso de técnicas exhaustivas del estilo de branch&bound)

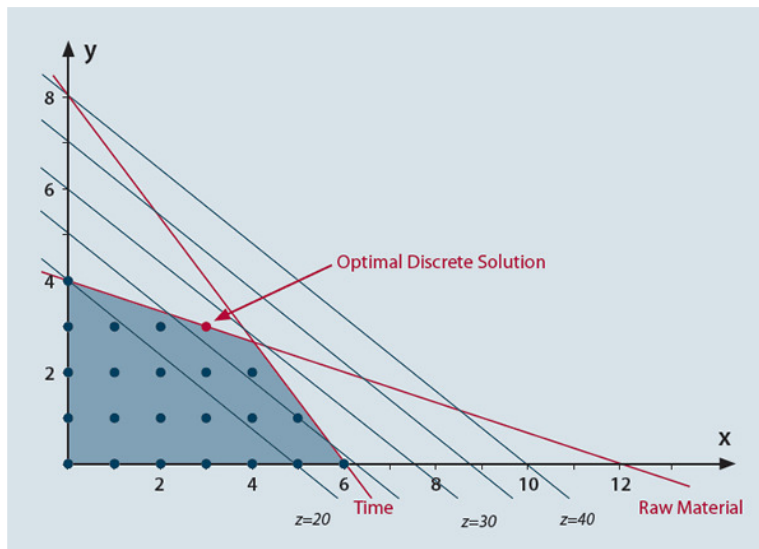
- Problema de la asignación cuadrática (QAP),  $O(n!)$   
[http://en.wikipedia.org/wiki/Quadratic\\_assignment\\_problem](http://en.wikipedia.org/wiki/Quadratic_assignment_problem)





## Problemas de optimización combinatoria

- Programación entera [integer programming]  
[http://en.wikipedia.org/wiki/Integer\\_programming](http://en.wikipedia.org/wiki/Integer_programming)



ILP =

Caso discreto de la programación lineal.



## Ejemplos

Espacios de búsqueda discretos (optimización combinatoria)

- Scatter search  
Fred Glover: "Heuristics for Integer programming Using Surrogate Constraints". Decision Sciences 8 (1): 156–166, 1977.
- Búsqueda tabú [tabu search]  
Fred Glover and C. McMillan: "The general employee scheduling problem: an integration of MS and AI". Computers and Operations Research, 1986.
- GRASP = Greedy Randomized Adaptive Search Procedure  
T.A. Feo and M.G.C. Resende: "A probabilistic heuristic for a computationally difficult set covering problem". Operations Research Letters, 8:67–71, 1989.





## Ejemplos

### Espacios de búsqueda discretos (optimización combinatoria)

- **Enfriamiento simulado**

Kirkpatrick, S.; Gelatt Jr., C.D.; Vecchi, M.P. (1983).

"Optimization by Simulated Annealing".

Science 220 (4598):671–680. DOI 10.1126/science.220.4598.671

- **Algoritmos genéticos**

John H. Holland: "Adaptation in Natural and Artificial Systems".

University of Michigan Press, 1975

- **Colonias de hormigas**

Marco Dorigo: "Optimization, Learning and Natural Algorithms"

PhD thesis, Politecnico di Milano, 1992.



## Ejemplos

### Espacios de búsqueda continuos

- **Nubes de partículas**

J. Kennedy & R. Eberhart: "Particle Swarm Optimization".

Proceedings of IEEE International Conference on Neural Networks, 1995.

- **Evolución diferencial**

R. Storn & K. Price: "Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces". Journal of Global Optimization 11 (4): 341–359, 1997.

- **Estrategias de evolución**

I. Rechenberg: *Evolutionstrategie – Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, PhD Thesis, 1971.



# Ascensión de colinas



## Ascensión de colinas simple

E: Estado activo

```
while (E no sea el objetivo
      y queden nodos por explorar a partir de E)
  Seleccionar operador R para aplicarlo a E
  Evaluar  $f(R(E))$ 
  if ( $f(R(E)) < f(E)$ )
    E = R(E)
```

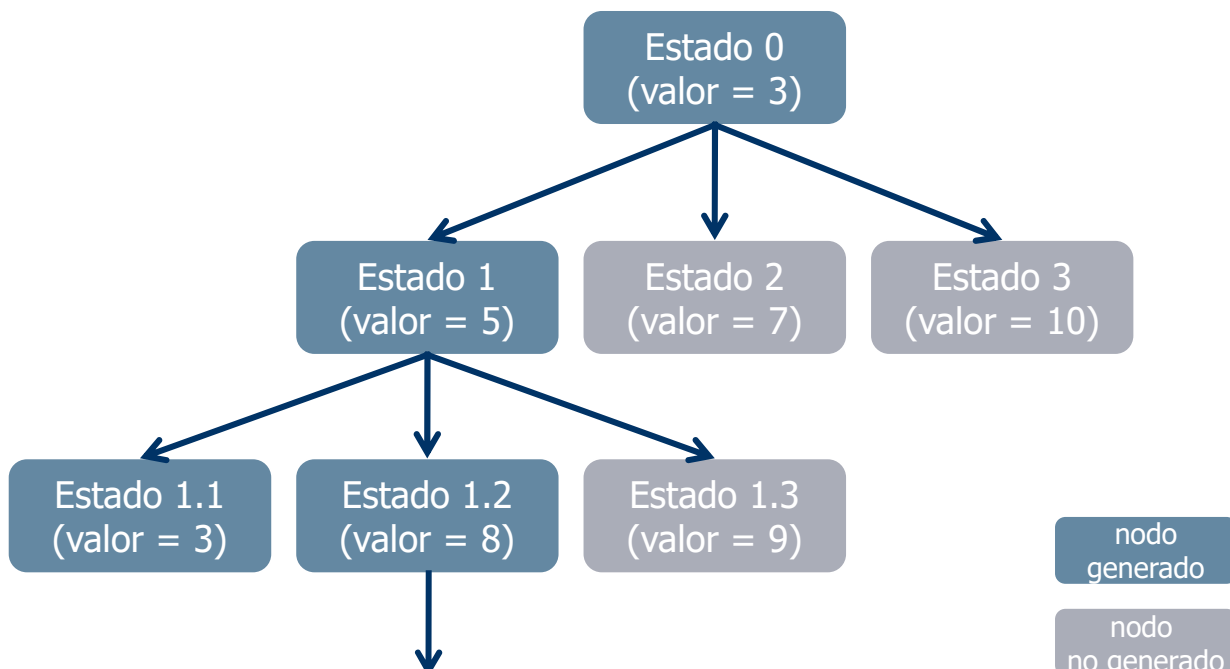


# Ascensión de colinas



## Ascensión de colinas simple

“Como subir al Everest con una niebla espesa y amnesia”



# Ascensión de colinas



## Ascensión de colinas por la máxima pendiente

E: Estado activo

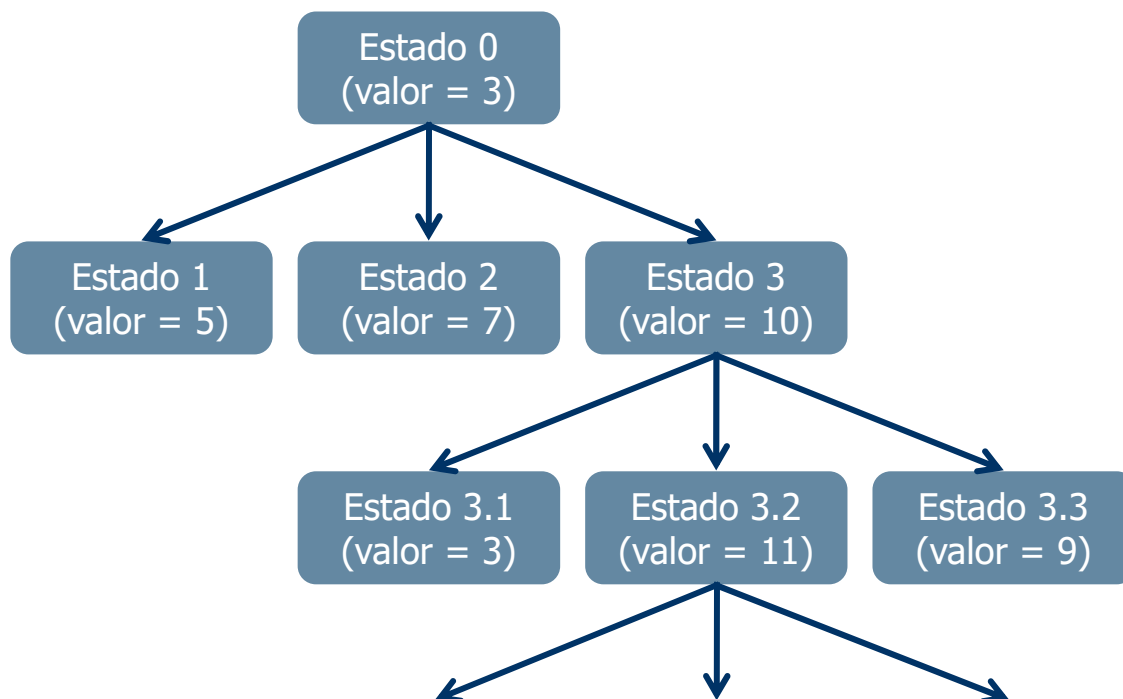
```
while (queden nodos por explorar a partir de E)
  Para todos los operadores  $R_i$ , obtener  $E_i=R_i(E)$ 
  Evaluar  $f(E_i)$  para todos los estados  $E_i=R_i(E)$ 
  Seleccionar  $E_{\max}$  tal que  $f(E_{\max}) = \max\{f(E_i)\}$ 
  if ( $f(E_{\max})>f(E)$ )
     $E = E_{\max}$ 
  else
    return E
```



# Ascensión de colinas



## Ascensión de colinas por la máxima pendiente





# Ascensión de colinas



## Ascensión de colinas por la máxima pendiente

### Observaciones

- Se realiza una búsqueda del mejor hijo del nodo actual  $E$  (para seleccionar el operador  $R$  más prometedor).
- Una vez elegido el operador  $R$  más prometedor, obtenemos el estado  $E' = R(E)$  y se repite el proceso a partir del estado  $E'$
- En memoria sólo tenemos que almacenar  $E$ , el hijo de  $E$  que estamos considerando en cada momento y el mejor hijo que hayamos encontrado.



# Ascensión de colinas



## Ascensión de colinas por la máxima pendiente

### Observaciones

- El proceso se repite hasta que se encuentre una solución, hasta no podamos avanzar más o hasta que todos los hijos sean peores que el padre del que provienen.
- Este último criterio puede dar lugar a varios problemas debido a la existencia de máximos locales y mesetas en la función de evaluación heurística.

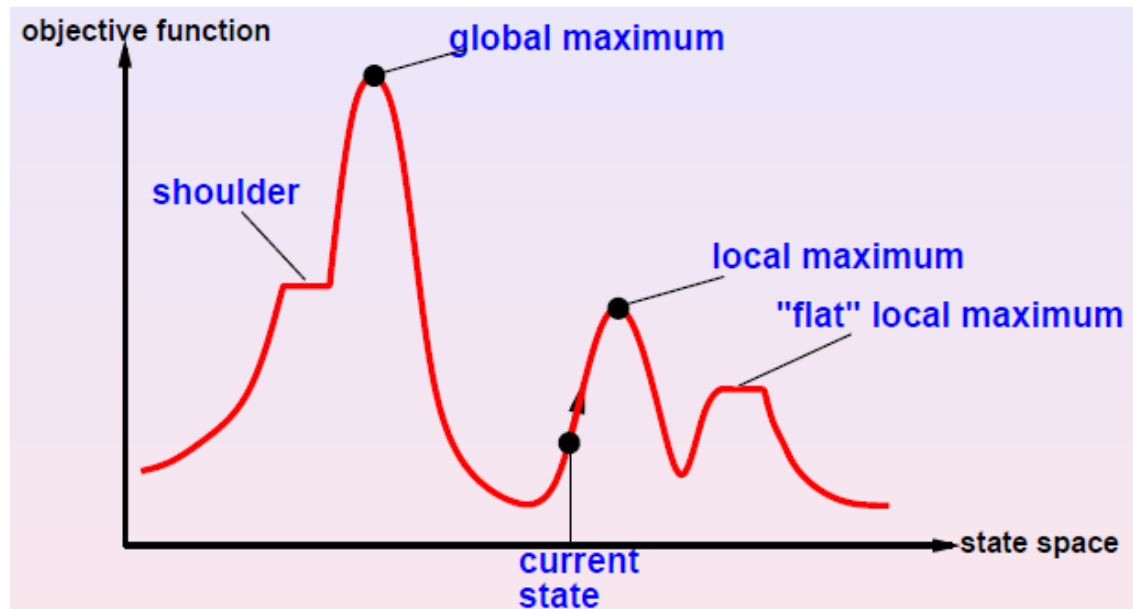


# Ascensión de colinas



## Ascensión de colinas por la máxima pendiente

### Limitaciones



# Ascensión de colinas



## Ascensión de colinas por la máxima pendiente

### Limitaciones

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 8 |   | 4 |
| 7 | 6 | 5 |

**objetivo**

|   |   |   |
|---|---|---|
| 3 | 2 | 1 |
| 8 |   | 4 |
| 7 | 6 | 5 |

**$E_{\text{máximo local}}$**

### Máximo local

Todos los movimientos empeoran el valor de la función heurística.

|   |   |   |
|---|---|---|
| 1 | 2 | 3 |
| 6 | 7 | 4 |
|   | 8 | 5 |

**$E_{\text{meseta}}$**

### Meseta

Todos los movimientos dejan igual el valor de la función heurística.





## Ascensión de colinas por la máxima pendiente

### Posibles soluciones

- Continuar la exploración de más niveles del árbol (no estaríamos ante una estrategia de búsqueda "local").
- Probar con distintos puntos de partida.
- "Dar saltos", aunque sólo sea de vez en cuando.

vg. enfriamiento simulado [simulated annealing]  
algoritmos genéticos



# Enfriamiento simulado



- Proceso de recocido del acero y cerámicas que consiste en calentar y luego enfriar lentamente el material para variar sus propiedades físicas.
- El calor causa que los átomos aumenten su energía y que puedan así desplazarse de sus posiciones iniciales (un mínimo local de energía).
- El enfriamiento lento les da mayores probabilidades de recristalizar en configuraciones con menor energía que la inicial (mínimo global).



# Enfriamiento simulado



## IDEA

Escapar de los máximos locales permitiendo movimientos malos. Gradualmente, tales movimientos decrecen en tamaño y frecuencia.

## MECANISMO: Temperatura $T(i)$

- Empieza siendo alta y decrece hasta aproximarse a 0.
- A una temperatura  $T(i)$ , la probabilidad de cada estado se puede calcular usando la distribución de Boltzmann:

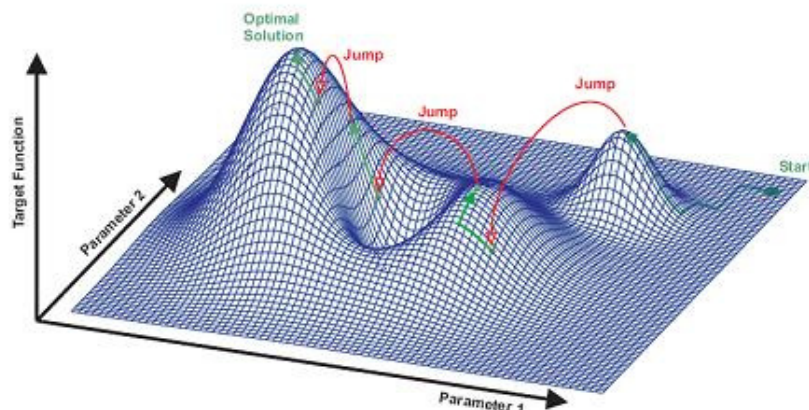
$$p(x) = \alpha e^{-\frac{E(x)}{kT}}$$



# Enfriamiento simulado



## Simulated Annealing



# Enfriamiento simulado



## Algoritmo

```
s ← s0; e ← E(s)           // Estado y energía inicial
best ← s; ebest ← e       // Mejor solución actual
k ← 0                     // Número de iteración
while (k < kmax) and (e > emin)
  T ← temperature(k/kmax) // Temperatura actual
  snew ← neighbour(s)     // Vecino aleatorio
  enew ← E(snew)         // Energía asociada al vecino
  if P(e, enew, T) > random()
    s ← snew; e ← enew   // Salto aleatorio al vecino
  if enew < ebest then
    sbest ← snew; ebest ← enew // Mejor solución encontrada
  k ← k + 1
return sbest              // Resultado final
```



# Algoritmos genéticos



- Se hace evolucionar una población de individuos (cada uno de los cuales representa una posible solución).
- La población se somete a acciones aleatorias semejantes a las de la evolución biológica (mutaciones y recombinaciones genéticas).
- Los individuos se seleccionan de acuerdo con una función de adaptación en función del cual se decide qué individuos sobreviven (los más adaptados) y cuáles son descartados (los menos aptos).



# Algoritmos genéticos

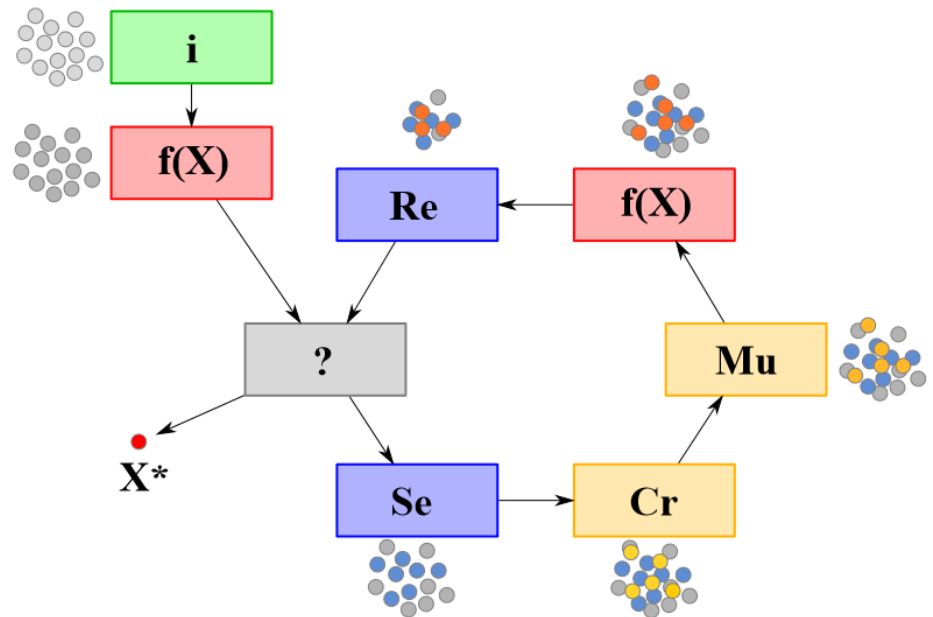


## Fases

- Inicialización
- Evaluación

Repetición...

- Selección
- Cruce
- Mutación
- Evaluación
- Reemplazo



# Algoritmos genéticos



## Algoritmo

$t \leftarrow 0$

población(t)  $\leftarrow$  poblaciónInicial

EVALUAR(población(t))

while not (condición de terminación)

$t \leftarrow t + 1$

población(t)  $\leftarrow$  SELECCIONAR(población(t-1))

población(t)  $\leftarrow$  CRUZAR(población(t))

población(t)  $\leftarrow$  MUTAR(población(t))

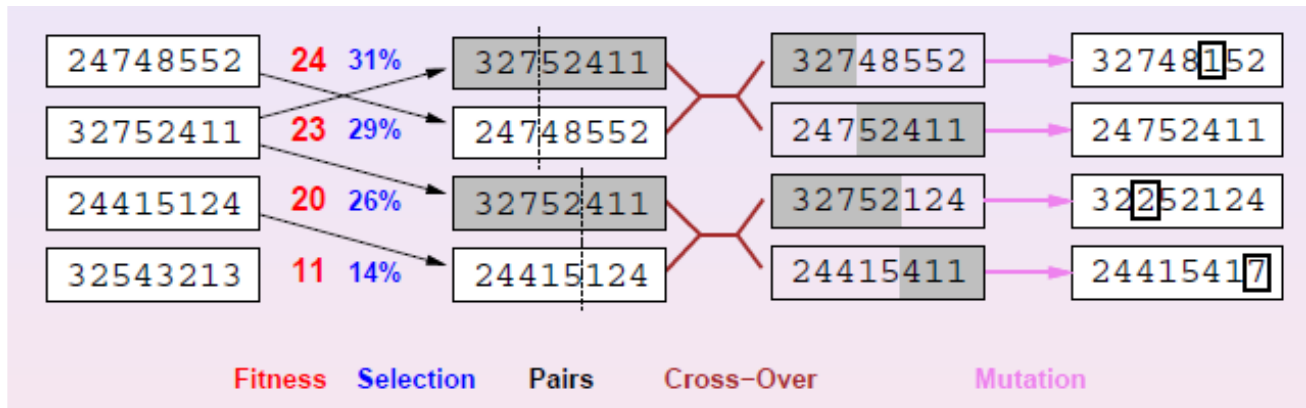
EVALUAR(población(t))

return población(t)



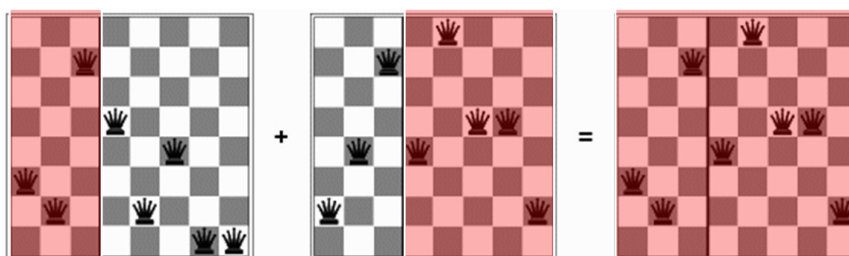


## Selección, cruce & mutación



## Ejemplo: El problema de las N reinas

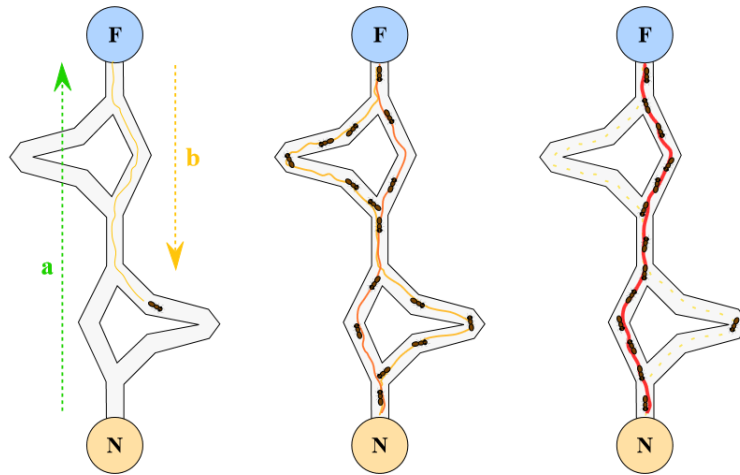
- Función de evaluación:  
Número de parejas de reinas que no se atacan.
- Operador de cruce:



# Colonias de hormigas



- Inicialmente, las hormigas se mueven aleatoriamente.
- Cuando encuentran comida, al volver a su colonia, van dejando un rastro de feromonas.
- Cuando otras hormigas encuentran feromonas, tienden a seguir el rastro dejado por sus predecesoras.

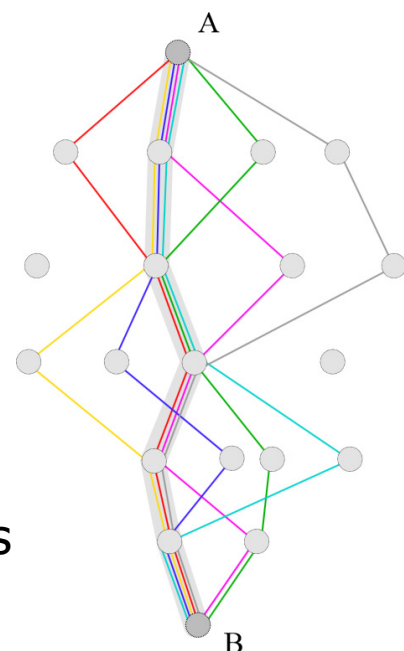


# Colonias de hormigas



Cuando encuentran comida, las hormigas vuelven de forma más o menos directa a la colonia (dejando un rastro de feromonas).

Como las feromonas se evaporan, el camino más corto acabará resultando más atractivo para otras hormigas y el más largo acabará desapareciendo.





# Colonias de hormigas



## Aplicación original: TSP

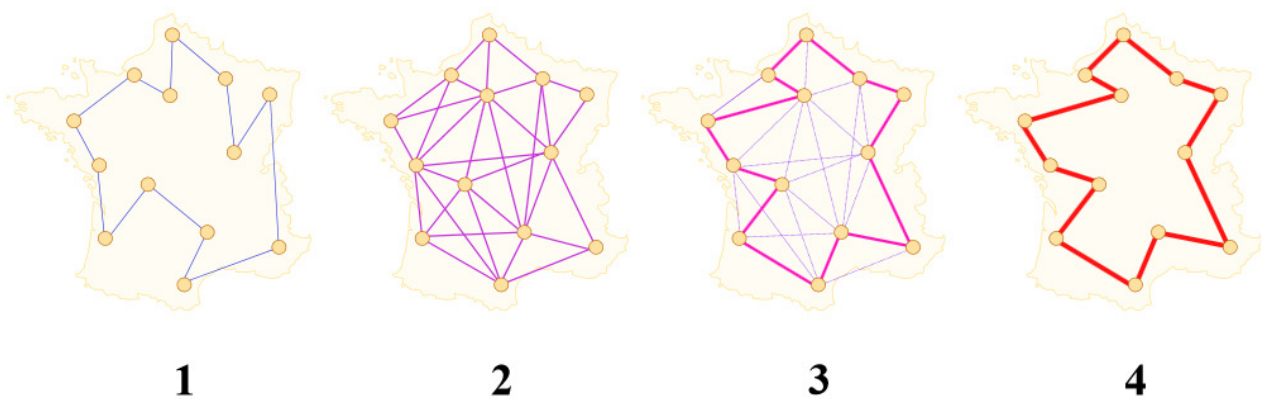
- La probabilidad de visitar una ciudad depende de su distancia (una ciudad lejana es menos visible).
- Cuanto mayor sea el nivel de feromonas asociado a un camino, mayor es la probabilidad de que se siga ese camino.
- Al completar un circuito, una hormiga deposita más feromonas en cada arista recorrida cuanto más corto es el camino que ha encontrado.
- Al final de cada iteración, las feromonas se evaporan.



# Colonias de hormigas



## Aplicación original: TSP



# Nubes de partículas



- Se mantiene una población [“enjambre” o nube] de soluciones candidatas [“partículas”].
- La posición de cada partícula se actualiza iterativamente de acuerdo a una fórmula definida sobre su posición y velocidad.

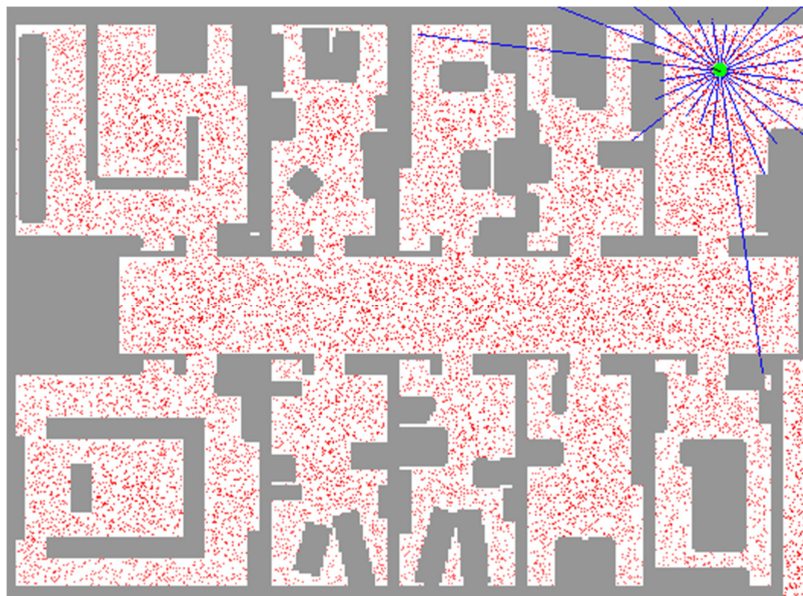


# Nubes de partículas



## Aplicación: Filtro de partículas

[Método secuencial de Monte Carlo]



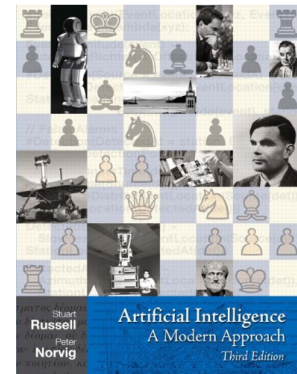
<http://www.cs.washington.edu/robotics/mcl/>



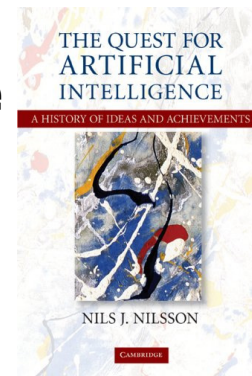
# Bibliografía



- Stuart Russell & Peter Norvig:  
**Artificial Intelligence:  
A Modern Approach**  
Prentice-Hall, 3<sup>rd</sup> edition, 2009  
ISBN 0136042597



- Nils J. Nilsson  
**The Quest for Artificial Intelligence**  
Cambridge University Press, 2009  
ISBN 0521122937



# Bibliografía



## Bibliografía complementaria

- Elaine Rich & Kevin Knight:  
**Artificial Intelligence.**  
McGraw-Hill, 1991.
- Patrick Henry Winston:  
**Artificial Intelligence.**  
Addison-Wesley, 1992.
- Nils J. Nilsson:  
**Principles of Artificial Intelligence.**  
Morgan Kaufmann, 1986.

