Recocido Simulado

Grado en Ingeniería Informática

Organización y Gestión de Empresas

José Manuel Galán
Luis R. Izquierdo

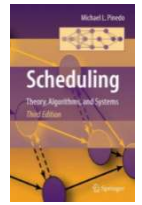
Universidad de Burgos

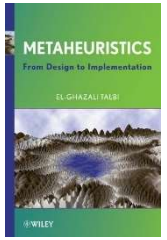
Estructura de la presentación

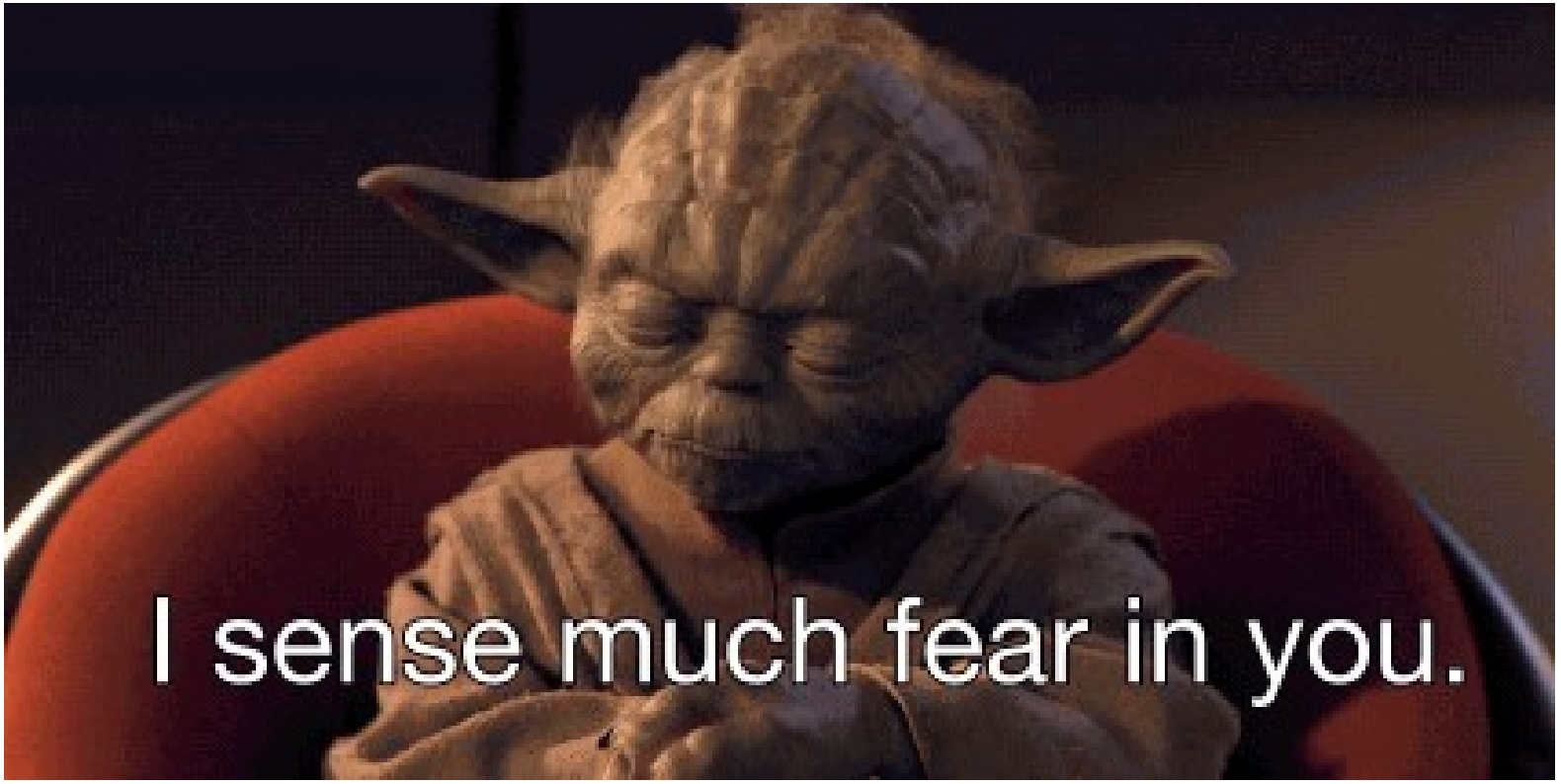
Bibliografía
Recocido
Recocido simulado
Algoritmo básico
Implementación
Ejemplos
Resumen

Bibliografía

- Pinedo, M.L. (2008) Scheduling: Theory, Algorithms, and Systems. Springer. Berlin.
- Hillier, F.S., Lieberman, G.S. (2008) Introducción a la Investigación de Operaciones. McGraw-Hill. México D.F.
- B. Melián, J.A. Moreno Pérez, J.M. Moreno Vega. Metaheurísticas: un visión global. Revista Iberoamericana de Inteligencia Artificial 19 (2003) 7-28
- Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.
- A. Duarte Muñoz et al (2007) Metaheurísticas. Dykinson. Madrid.
- S. Luke (2013) Essentials of Metaheuristics. Lulu. Available for free at http://cs.gmu.edu/$\sim$sean/book/metaheuristics/

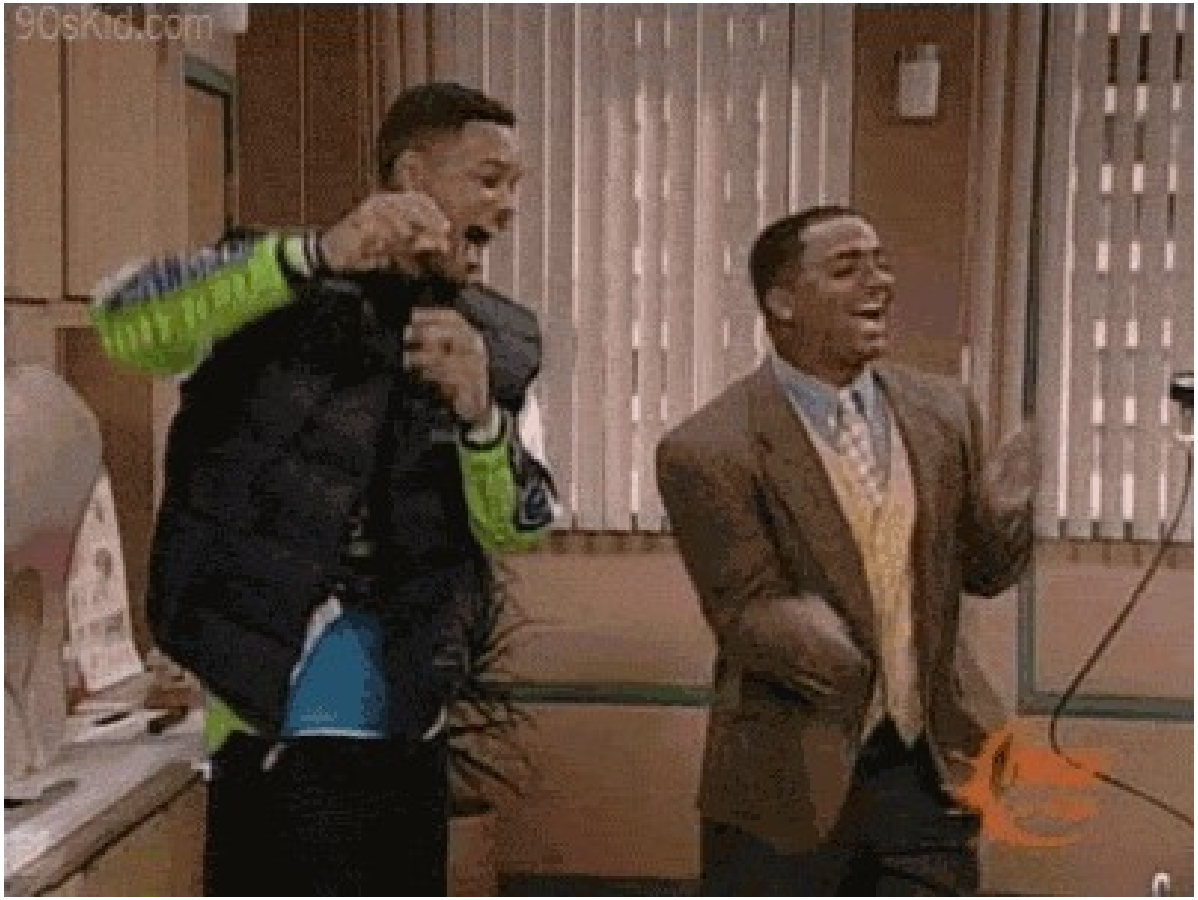






Source: https://giphy.com

Source: https://giphy.com



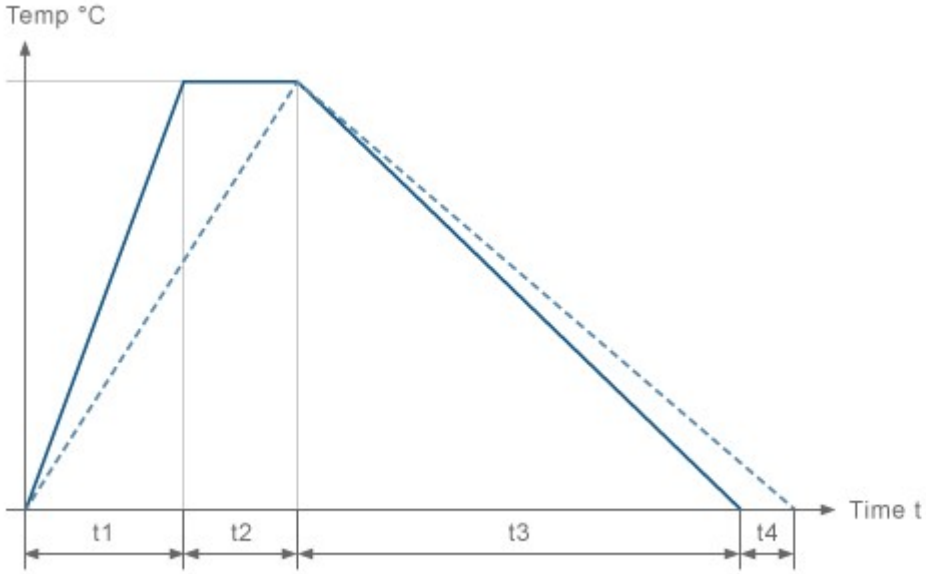
Source: https://giphy.com

Recocido

- **Recocido**, en metalurgia y ciencia de materiales, es un tratamiento térmico que altera las propiedades físicas y en algunos casos las químicas de un material aumentando su ductilidad y reduciendo su dureza, aumentando su trabajabilidad. Consiste en aumentar la temperatura por encima de la temperatura de recristalización, mantener una temperatura apropiada y posteriormente enfriar lentamente.
- Durante el recocido, los átomos migran a través de la retícula cristalina y el número de dislocaciones decrece reduciendo las tensiones del material. Esto conlleva al cambio en ductilidad y el ablandamiento.
- En la mayoría de los casos, el proceso se realiza calentando el material y posteriormente dejándolo enfriar lentamente al aire

Source: https://en.wikipedia.org/wiki/Recocido_(metallurgy)

Recocido



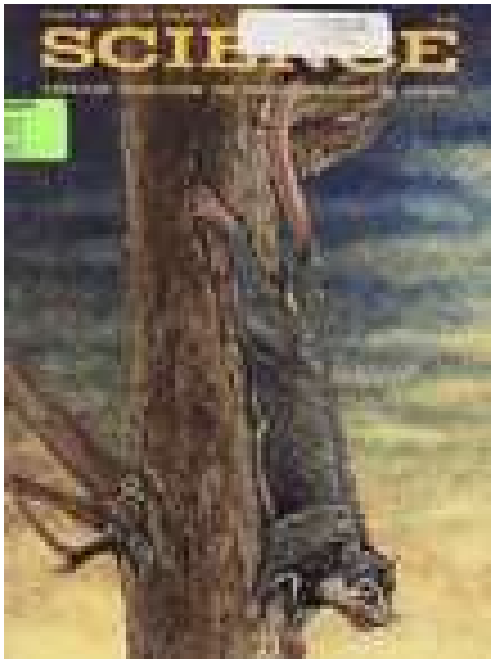
t1: Warm-up time
t2: Holding time
t3: Cooling time
t4: Additional time

The times are dependent upon the wall thickness and the material used for the semi-finished part or the finished part.

Source: http://www.ensinger-online.com/vn/products/stock-shapes/machining-of-semi-finished-parts/annealing/
http://www.keighleylabs.co.uk/heat-treatment-html/annealing/

Recocido simulado

- El **Recocido Simulado** es un algoritmo basado en trayectorias que utiliza una metáfora termodinámica como mecanismo de decisión en la aceptación de soluciones
- S. Kirkpatrick and C. D. Gelatt and M. P. Vecchi, Optimization by Simulated Annealing. *Science* 220 (4598) pages 671-680. 1983



Original paper: http://leonidzhukov.net/hse/2013/stochmod/papers/KirkpatrickGelattVecchi83.pdf

Optimización mediante recocido simulado

Science 13 May 1983: Vol. 220. no. 4598, pp. 671 – 680 DOI: 10.1126/science.220.4598.671

S. Kirkpatrick [1], C. D. Gelatt Jr. [1], and M. P. Vecchi [2]

[1] Research staff members at IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598
[2] Instituto Venezolano de Investigaciones Cientificas, Caracas 1010A, Venezuela

- *There is a deep and useful connection between **statistical mechanics** (the behavior of systems with many degrees of freedom in thermal equilibrium at a finite temperature) and **multivariate or combinatorial optimization** (finding the minimum of a given function depending on many parameters). A detailed analogy with annealing in solids provides a framework for optimization of the properties of very large and complex systems. This connection to statistical mechanics exposes new information and provides an unfamiliar perspective on traditional optimization problems and methods.*

Source: S. Kirkpatrick and C. D. Gelatt and M. P. Vecchi, Optimization by Recocido Simulado. *Science* 220 (4598) pages 671-680. 1983. Doi: 10.1126/science.220.4598.671

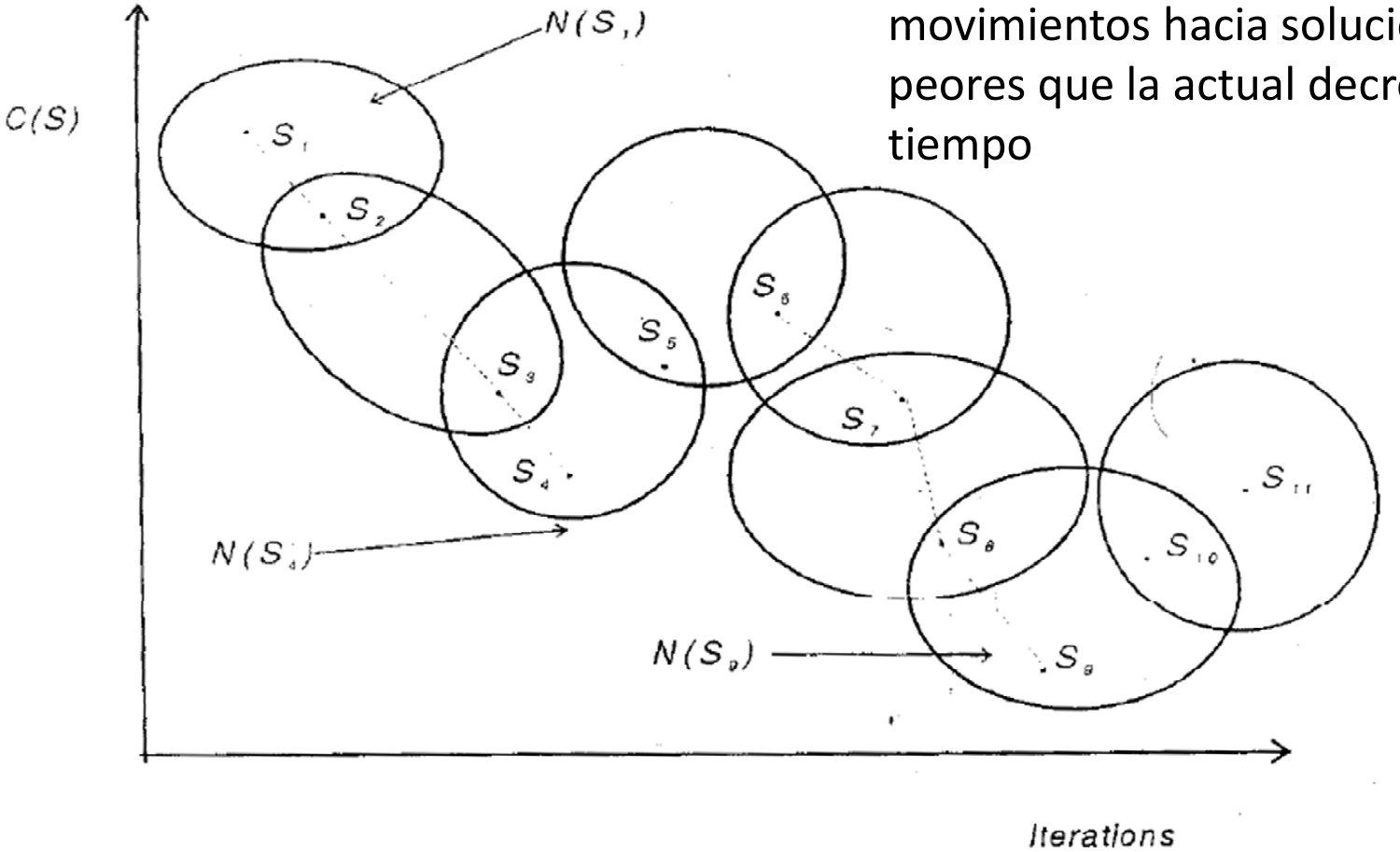
Fundamentos

- Un mecanismo para evitar atascarse en óptimos locales, lo que frecuentemente ocurre en la búsqueda local ingenua, es **aceptar soluciones peores**
- Este proceso –tratar de escapar de óptimos locales – se debe llevar a cabo de forma **controlada**
- En el caso del algoritmo de recocido simulado, el proceso se controla mediante el parámetro temperatura: el algoritmo interpreta el enfriamiento lento como un **lento descenso en la probabilidad de aceptar peores soluciones a medida que se explora el espacio de soluciones**
- De esta forma se aplica la filosofía de diversificar y explorar en las etapas iniciales del algoritmo e intensificar la búsqueda al final

Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Fundamentos

La probabilidad de aceptar movimientos hacia soluciones peores que la actual decrece con el tiempo



Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Fundamentos. Metropolis model

En 1953, Metrópolis y otros introdujeron un modelo simple para similar la evolución de un sólido sometido a un baño de calor para el equilibrio térmico

En su modelo la probabilidad a una temperatura dada T de un salto a un estado con una energía diferente viene dada por la distribución de probabilidad de Boltzmann

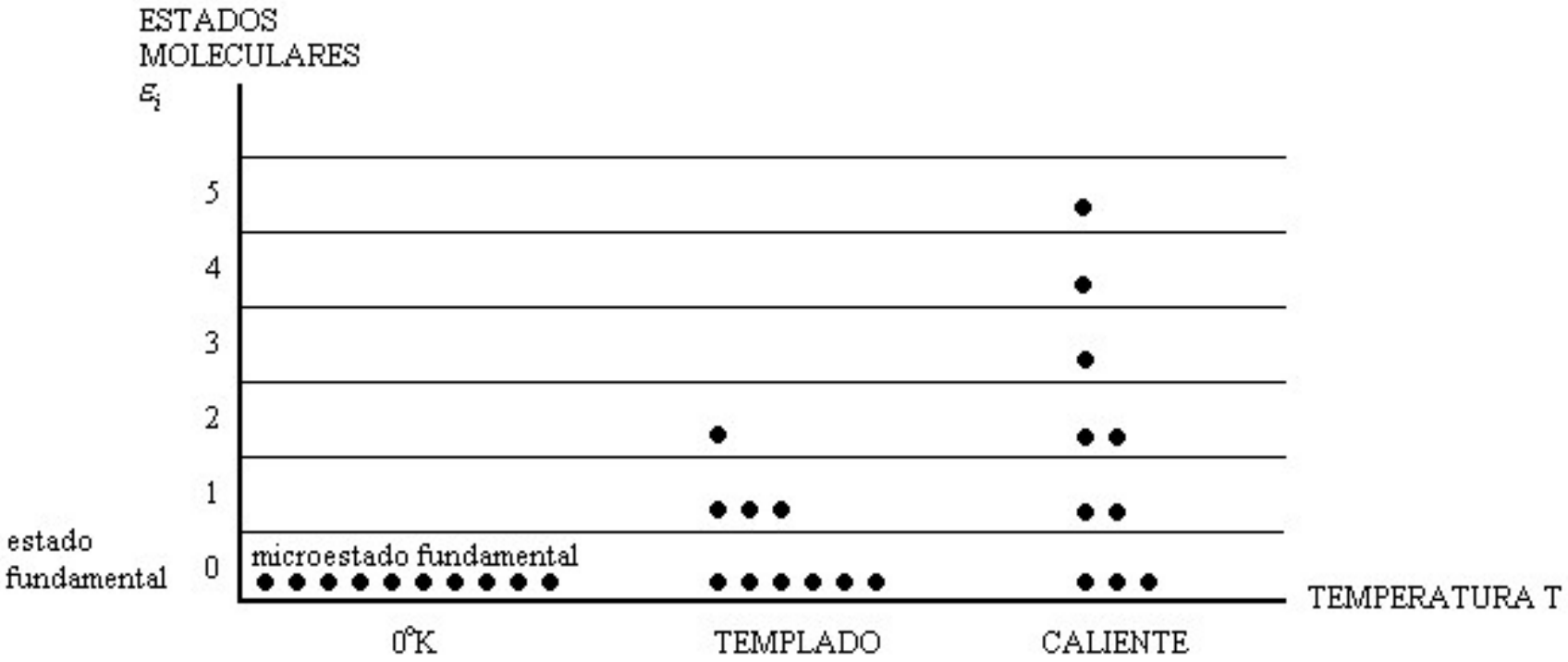
$$P[\delta E] = e^{\left(\frac{-\delta E}{kt}\right)}$$

Donde k es la constante de Boltzmann

Dado un estado actual *i* de un sólido con energía *E*, un posterior estado *j* se obtiene mediante un mecanismo de perturbación que transforma el estado actual en otro estado con una pequeña distorsión, por ejemplo, mediante el desplazamiento de una partícula individual. Si la diferencia de energía es igual o menor que 0, el estado es aceptado y pasa a ser el estado actual. Si la diferencia de energía es mayor que 0 entonces el estado se acepta de acuerdo a la distribución de probabilidad de la ecuación

Source: Gonzalez, T. F. (Ed.). (2007). Handbook of approximation algorithms and metaheuristics. CRC Press.
Zäpfel, G., & Braune, R. (2010). Metaheuristic search concepts: A tutorial with applications to production and logistics. Springer Science & Business Media.

Fundamentos



Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Analogía entre el sistema físico y el problema de optimización

Sistema físico	**Problema de optimización**
Estado del sistema	Solución
Posición molecular	Variables de decisión
Energía	Función objetivo
Estado fundamental	Solución óptima global
Estado metaestable	Óptimo local
Enfriamiento rapido (templado)	Búsqueda local
Temperatura	Parámetro de control T
Recocido cuidadoso	Recocido simulado

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Metáfora

- Cada configuración de una solución en el espacio de búsqueda representa una energía interna diferente del sistema. Calentar el sistema implica relajar el criterio de aceptación de las muestras obtenidas en la búsqueda por el espacio de soluciones. A medida que el sistema se enfría, el criterio de aceptación se endurece para irse centrando en movimientos de mejora. Una vez que el sistema se ha enfriado, la configuración resultante será idealmente una solución cercana al óptimo global

Source: Brownlee, J. (2011). Clever algorithms: nature-inspired programming recipes. Jason Brownlee.

Fundamentos

- Algoritmo estocástico que permite bajo ciertas condiciones la degradación de la solución actual. El objetivo es escapar de óptimos locales y retrasar la convergencia
- Es un algoritmo sin memoria (no utiliza la información recogida a lo largo de la búsqueda)

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Algoritmo básico

- Partiendo de una solución inicial el algoritmo se ejecuta en varias iteraciones
- En cada iteración se genera un vecino aleatorio
- Movimientos que mejoran la función objetivo se aceptan siempre

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Algoritmo básico

- Si la solución vecina candidata es peor, entonces se selecciona una probabilidad dada que depende de la temperatura actual (T) y de cuanto se degrade la función objetivo (δ)
- A medida que el algoritmo progresa, la probabilidad de estos movimientos decrece (ya que la temperatura decrece a medida que avanza el algoritmo)
- Normalmente la probabilidad se modela siguiendo la distribución de Boltzmann

$$P_{aceptación} = e^{(-\delta/T)}$$

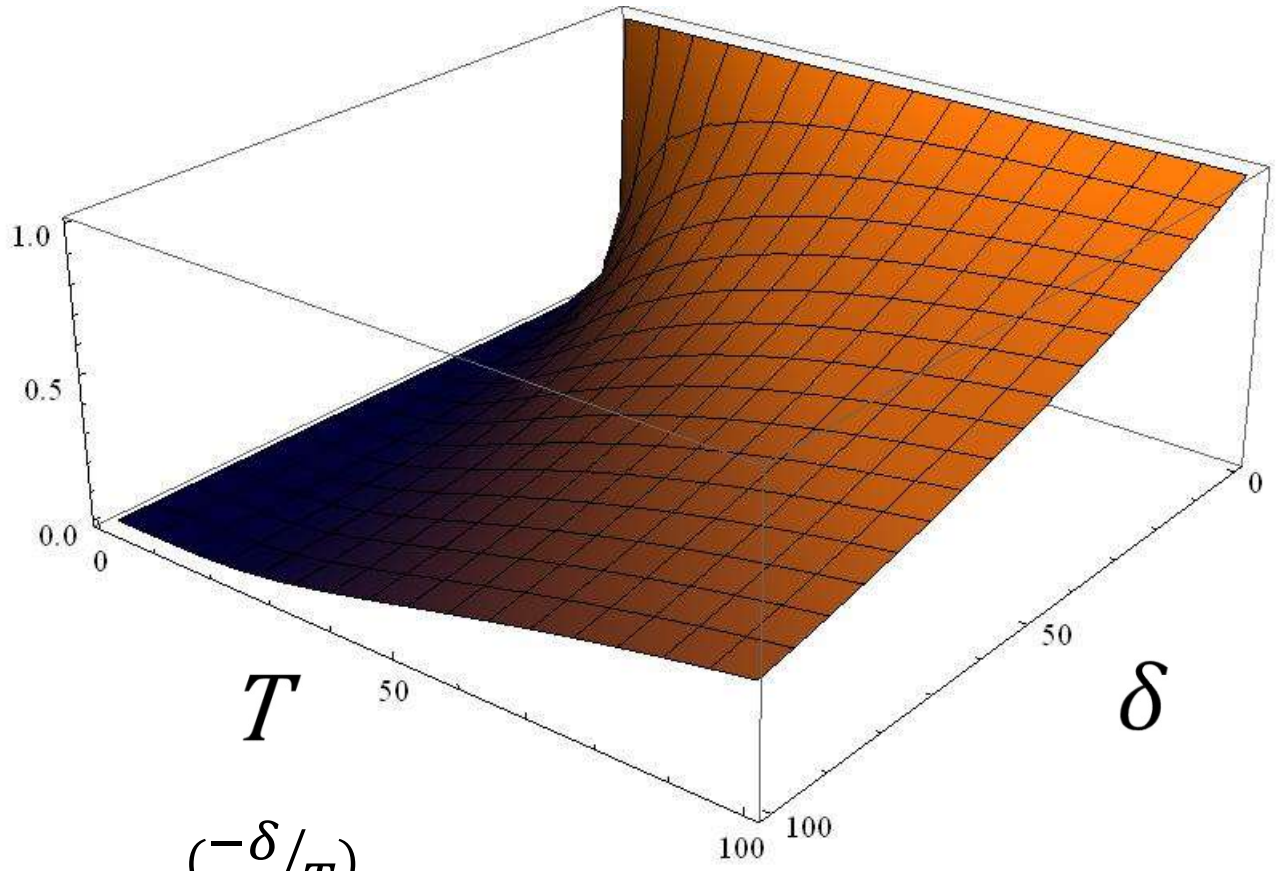
Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Algoritmo básico

- Se utiliza un parámetro de control, llamado temperatura (T), para determinar la probabilidad de aceptar soluciones peores
- En cada nivel de temperatura, se explora un número grande de intentos L(T)
- Una vez que se alcanza un estado de cierto equilibrio, gradualmente se baja la temperatura

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Algoritmo básico

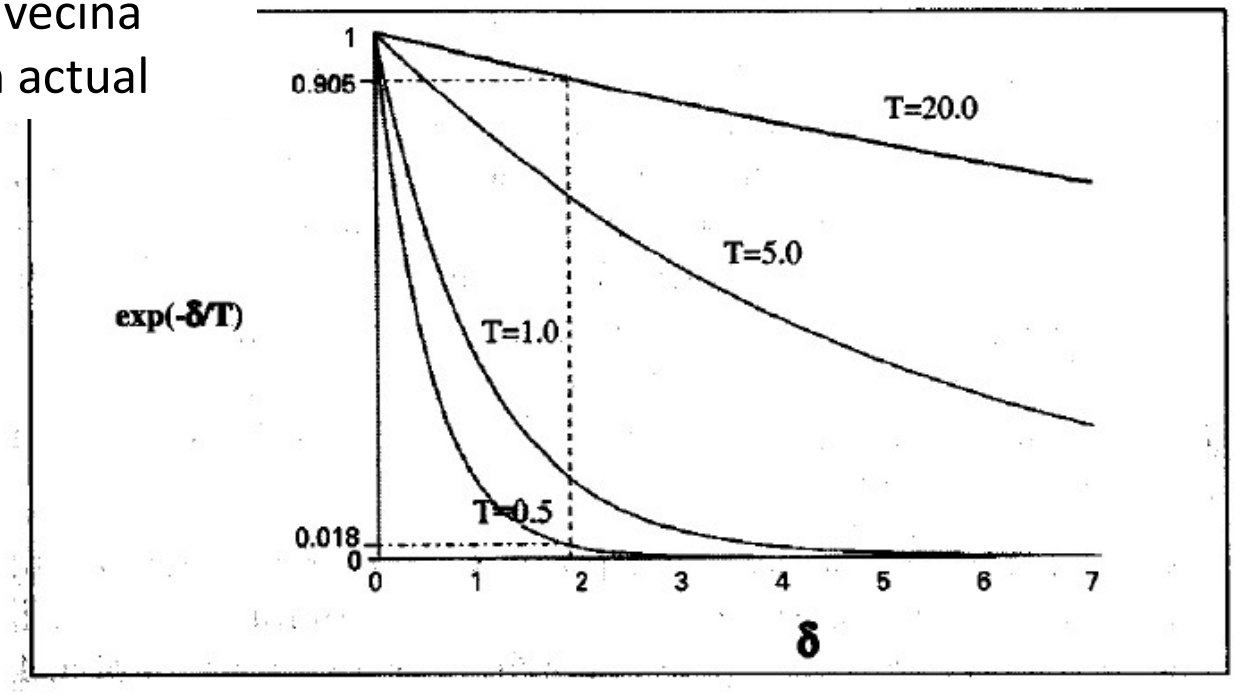


$$P_{aceptación} = e^{(-\delta/T)}$$

Based on: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
Figure developed with Mathematica Software

Algoritmo básico

δ = C(s')-C(s)
s= Solución vecina
s'= Solución actual



Para δ = 2, resulta 50 veces menos probable un movimiento si T=0.5 que si T=20

Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Algoritmo básico. Pseudocódigo

INPUT (T_0, α, L, T_f)
$T \leftarrow T_0$
$S_{act} \leftarrow$ Genera_Solución_Inicial
WHILE $T \geq T_f$ DO
    BEGIN
        FOR count ←1 TO L(T) DO
        BEGIN
        $S_{cand} \leftarrow$Genera_Vecino_Aleatorio(S_{act})
        $\delta \leftarrow$Cost(S_{cand})-Cost(S_{act})
        IF (U(0,1)<$e^{(-\delta/T)}$) OR (δ<0)
        THEN $S_{act} \leftarrow S_{cand}$
        END
    $T \leftarrow \alpha(T)$
    END

{Devuelve el mejor S_{act} visitado}

Algoritmo básico. Pseudocódigo

Algorithm 6.5: Simulated Annealing - control flow

```
Create initial solution s;
Set initial temperature T_0;
Set number of trials at each temperature level (level-length) 𝓛;
Set level count k ← 0;
while termination criterion not satisfied do
    for i=1 to 𝓛 do
        Create new neighbor s' by applying an arbitrary / random move to s;
        Calculate cost difference ΔC between s' and s: ΔC = C(s') − C(s);
        if ΔC ≤ 0 then
            Switch over to solution s' (current solution s is replaced by s');
        else
            Create random number r ∈ [0, 1];
            if r ≤ exp(−ΔC/T_k) then
                Switch over to solution s' (current solution s is replaced by s');
            end
        end
    end
    Update best found solution (if necessary);
    Set k ← k + 1;
    Set / Update temperature value T_k for next level k;
end
return Best found solution;
```


Source: Zäpfel, G., & Braune, R. (2010). Metaheuristic search concepts: A tutorial with applications to production and logistics. Springer Science & Business Media.

Algoritmo básico. Procesado de una solución

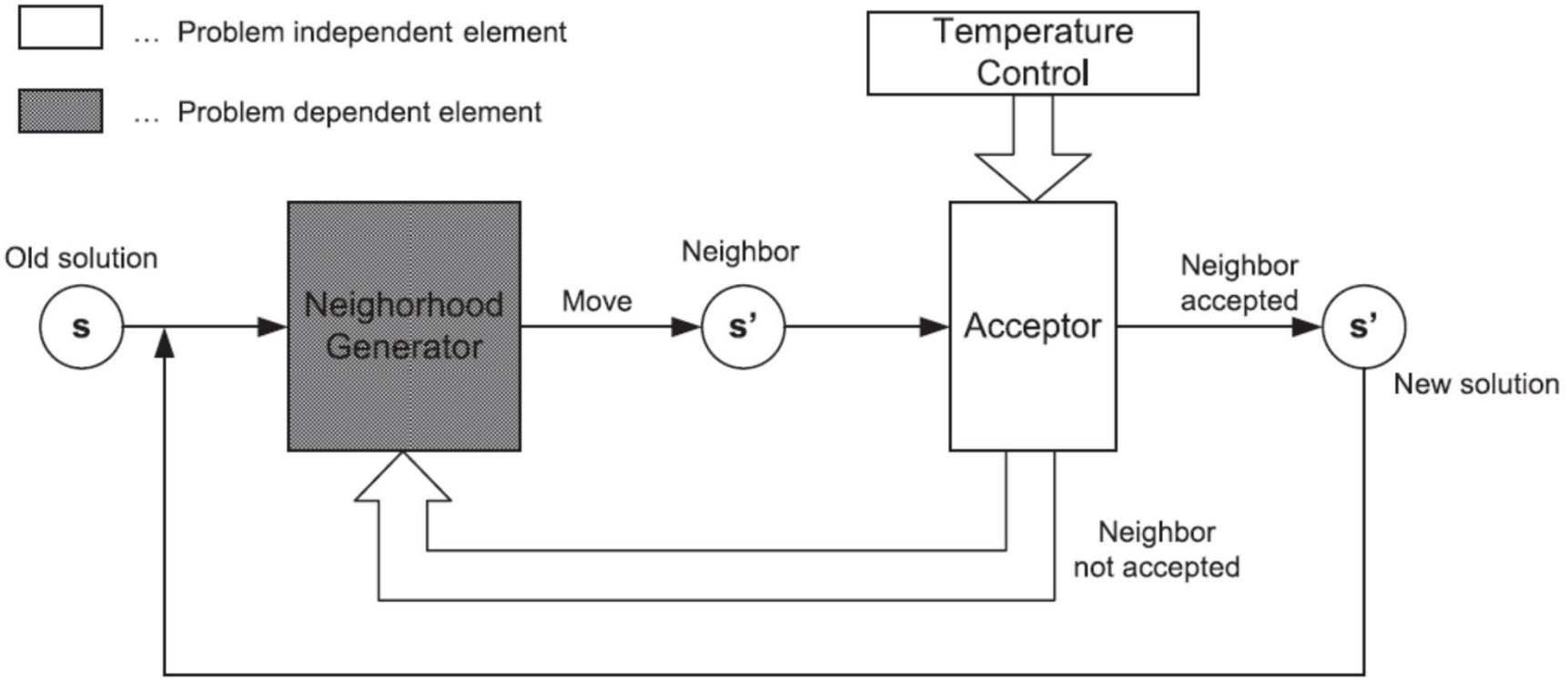


Fig. 6.14: Elements of Simulated Annealing - degree of problem dependency

Source: Zäpfel, G., & Braune, R. (2010). Metaheuristic search concepts: A tutorial with applications to production and logistics. Springer Science & Business Media.

Estructura de implementación

1. Representación
2. Solución inicial
3. Mecanismo de transición entre soluciones
4. Secuencia de enfriamiento

Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Representación

- Codificación permutacional
  - Ejemplo: problema del viajante
  - (7 6 3 1 5 4 2 8)
- Codificación binaria
  - Ejemplo: problema de la mochila
  - (0 1 0 0 1 1 1 0)
- Codificación real
  - Ejemplo: problema de optimización con parámetros reales
  - (2.7 3.5 4 6.27)

Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Solución inicial

- Aleatoria
- Solución previa
- Solución obtenida con otras heurísticas

Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Mecanismo de transición entre soluciones

1. Generación de una nueva solución
   i. Definición de vecindad (¿cuál es el conjunto?)
   ii. Selección de un elemento del conjunto vecindad
2. Cálculo de la diferencia en la función objetivo entre la solución actual y la candidata (del vecindario)
3. Criterio de aceptación

Source: Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica

Mecanismo de transición

INPUT (T_0, α, L, T_f)
T← T_0
S_{act} ← Genera_Solución_Inicial
WHILE T≥ T_f DO
    BEGIN
        FOR count ←1 TO L(T) DO
        BEGIN
        S_{cand} ←Genera_Vecino_Aleatorio(S_{act})
        δ ←Cost(S_{cand})-Cost(S_{act})
        IF (U(0,1)<$e^{(-\delta/T)}$) OR (δ<0)
        THEN S_{act} ← S_{cand}
        END
    T ← α(T)
    END

{Devuelve el mejor S_{act} visitado}

Generación de una nueva solución

Cálculo de la diferencia en la función objetivo

Criterio de aceptación

Secuencia de enfriamiento

INPUT (T_0, α, L, T_f)
T$\leftarrow T_0$
$S_{act} \leftarrow$ Genera_Solución_Inicial
WHILE T$\geq T_f$ DO
    BEGIN
        FOR count $\leftarrow$1 TO L(T) DO
        BEGIN
        $S_{cand} \leftarrow$Genera_Vecino_Aleatorio(S_{act})
        $\delta \leftarrow$Cost(S_{cand})-Cost(S_{act})
        IF (U(0,1)<$e^{(-\delta/T)}$) OR (δ<0)
        THEN $S_{act} \leftarrow S_{cand}$
        END
    T $\leftarrow \alpha$(T)
    END

{Devuelve el mejor S_{act} visitado}

Valor inicial del parámetro de control

Criterio de terminación

Tiempo en una temperatura dada

Secuencia de enfriamiento

Temperatura inicial

- Si la temperatura inicial es muy alta, la búsqueda (al comienzo) será similar a una búsqueda local aleatoria. Por otro lado, si la temperatura inicial es demasiado baja, la búsqueda se parecerá una búsqueda local del primer vecino. Por tanto, es necesario equilibrar entre ambos procedimientos extremos

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Temperatura inicial

- Estrategias básicas habituales:
  - Aceptar todas
  - Porcentaje de la solución inicial
  - Técnicas más avanzadas
    - *Acceptance deviation*
    - *Acceptance ratio*
    - *Ben-Ameur method*
    - …

Source: Ben-Ameur, W. (2004). Computing the initial temperature of simulated annealing. *Computational Optimization and Applications*, 29(3), 369-385.
Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Estado de equilibrio

- Para alcanzar un estado de equilibrio a cada temperatura, se debe aplicar un número de transiciones suficiente. La teoría sugiere que el número de iteraciones L(T) a cada temperatura debería ser exponencial al tamaño del problema, lo cual en la práctica dificulta la estrategia. El número de iteraciones debería ser acorde al tamaño del problema y particularmente proporcional al tamaño del vecindario |N(s)|
  - **Estático**: el número de vecinos generados de una solución s es $y \cdot |N(s)|$. Cuanto mayor sea y, mayor será el coste computacional pero mejores los resultados
  - **[Tema Avanzado] Adaptativo:** el número de vecinos es generado dependiendo de las características de la búsqueda. Por ejemplo, no es necesario alcanzar el equilibrio a cada temperatura

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Estado de equilibrio

- Un método adaptativo simple consiste en reducir la temperatura cuando una de las siguientes situaciones se cumpla:
  - El algoritmo alcanza un número máximo de vecinos (max_neighbors)
  - El algoritmo acepta un número de soluciones candidatas máximo (max_successes)
- Típicamente max_successes =0.1* max_neighbors

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Secuencia de enfriamiento

- Existe un compromiso entre la calidad de la solución obtenida y la velocidad de la secuencia de enfriamiento. Si la temperatura se reduce lentamente, las soluciones obtenidas son generalmente mejores pero obtenerlas conlleva un tiempo computacional significativamente mayor

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Secuencia de enfriamiento

- Secuencia ad hoc determinada por el diseñador
- Lineal (i denota iteración)

$$T = T - \beta, \qquad T_i = T_0 - i \times \beta$$

- Geométrica o exponencial. La función más popular de enfriamiento considera un $\alpha \in [0.8, 0.99]$

$$T = \alpha T$$

- Logarítmica o de Boltzmann. Extremadamente lenta pero existe una prueba de convergencia al óptimo global

$$T(t) \geq \frac{T_0}{\ln(1+t)}, \qquad t = 1, 2, \ldots$$

- Cauchy: $T_i = T_0 / (1 + i)$

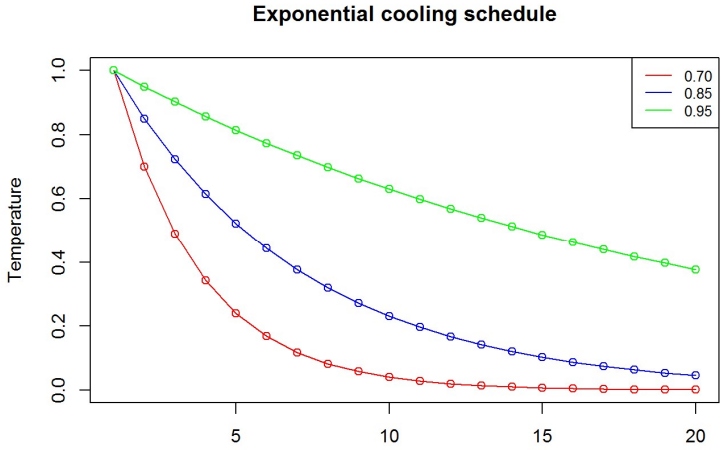
- Cauchy modificado utilizando balanceo o un número fijo de iteraciones

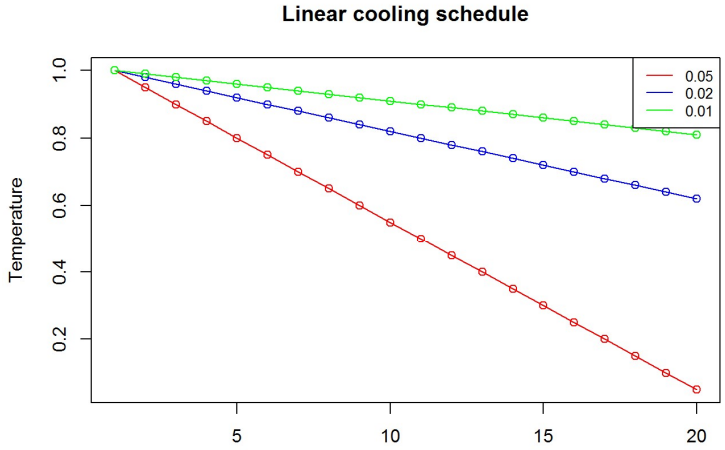
$$T_{i+1} = \frac{T_i}{1 + \beta T_i}$$

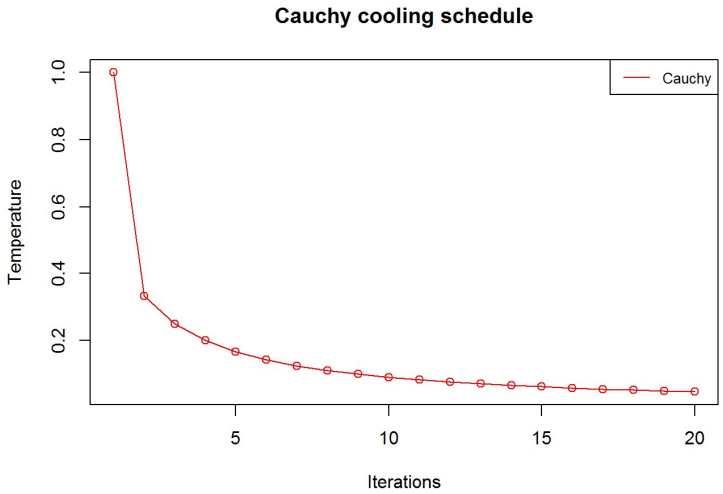
$$\beta = T_0 - T_F/(L-1)T_0T_F$$

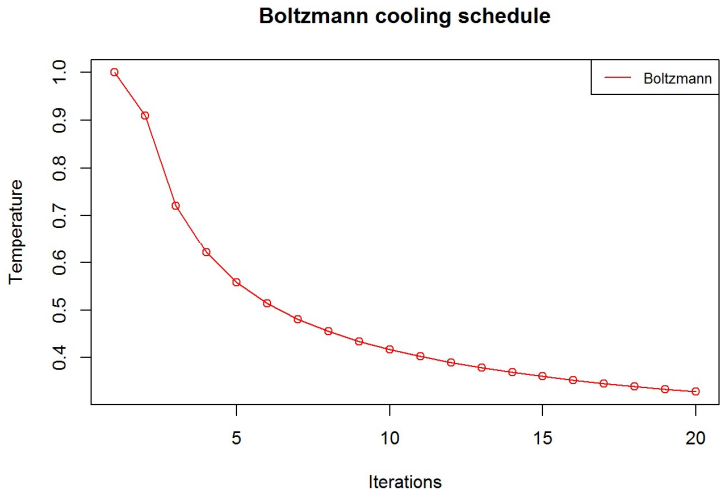
Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.
Apuntes Algorítmica. Softcomputing and Soft Computing and Intelligent Information Systems. Universidad de Granada
http://sci2s.ugr.es/graduateCourses/Algoritmica
More Secuencia de enfriamientos in:
http://www.btluke.com/simanf1.html

Secuencia de enfriamiento









Code: http://josema.izqui.org/docencia/CoolingSchedule.html

Criterio de parada

- Respecto al criterio de parada, la teoría sugiere que la temperatura final sea 0. En la práctica, algunas secuencias no alcanzan el 0 pero se debería parar cuando la probabilidad de aceptar un movimiento a una solución peor sea despreciable. Los criterios de parada habituales son:
  - Alcanzar una temperatura final Tf es criterio de parada más habitual. La temperatura debe ser baja (e.g., *T*min = 0.01).
  - Alcanzar un número predeterminado de iteraciones sin que la mejor solución encontrada mejore
  - Alcanzar un número predeterminado de veces un porcentaje de vecinos aceptados a cada temperatura; esto es, un contador incrementa en uno cada vez que la temperatura ha sido completada con menos de un porcentaje dado de movimientos aceptados y se resetea a 0 cuando una nueva solución se ha encontrado. Si el contador llega un límite predeterminado R, el algoritmo se para

Source: Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

Secuencia de enfriamiento

INPUT (T_0, α, L, T_f)
$T \leftarrow T_0$
$S_{act} \leftarrow$ Genera_Solución_Inicial
WHILE $T \geq T_f$ DO
    BEGIN
        FOR count ←1 TO L(T) DO
        BEGIN
        $S_{cand} \leftarrow$Genera_Vecino_Aleatorio(S_{act})
        $\delta \leftarrow$Cost(S_{cand})-Cost(S_{act})
        IF (U(0,1)<$e^{(-\delta/T)}$) OR (δ<0)
        THEN $S_{act} \leftarrow S_{cand}$
        END
    $T \leftarrow \alpha(T)$
    END

{Devuelve el mejor S_{act} visitado}

Valor inicial del parámetro de control

Criterio de terminación

Tiempo en una temperatura dada

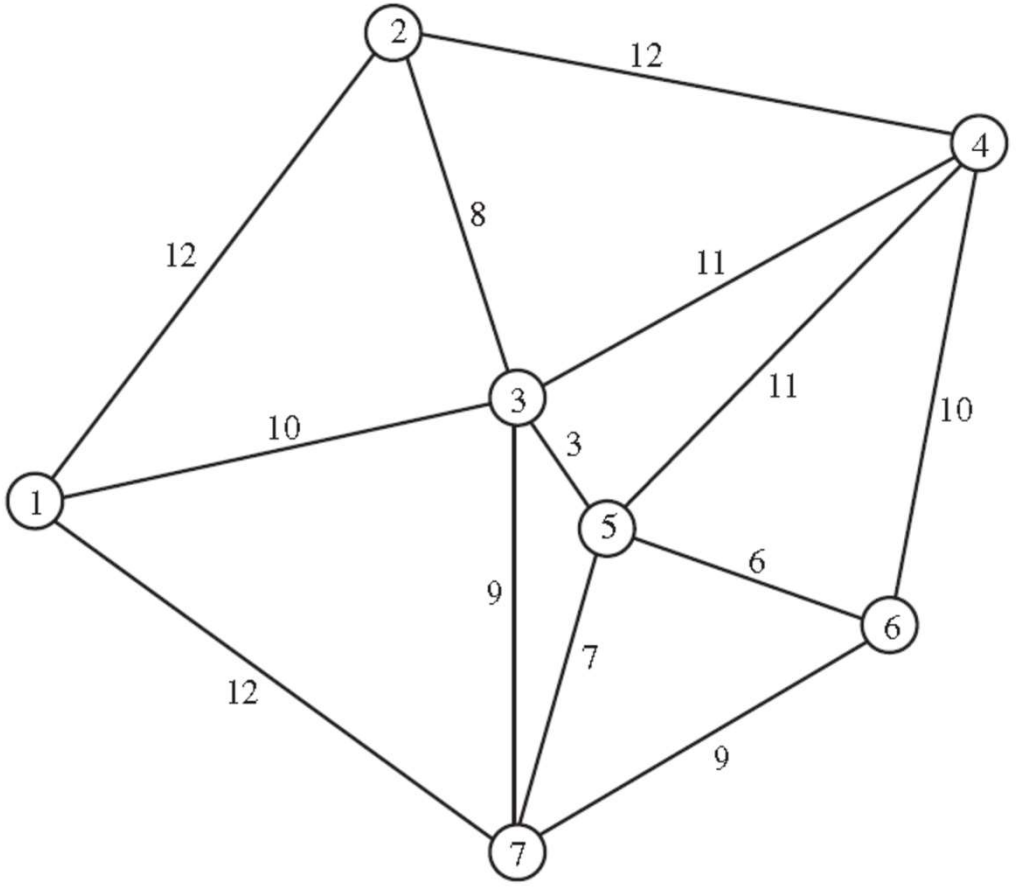
Generación de una nueva solución

Cálculo de la diferencia en la función objetivo

Criterio de aceptación

Secuencia de enfriamiento

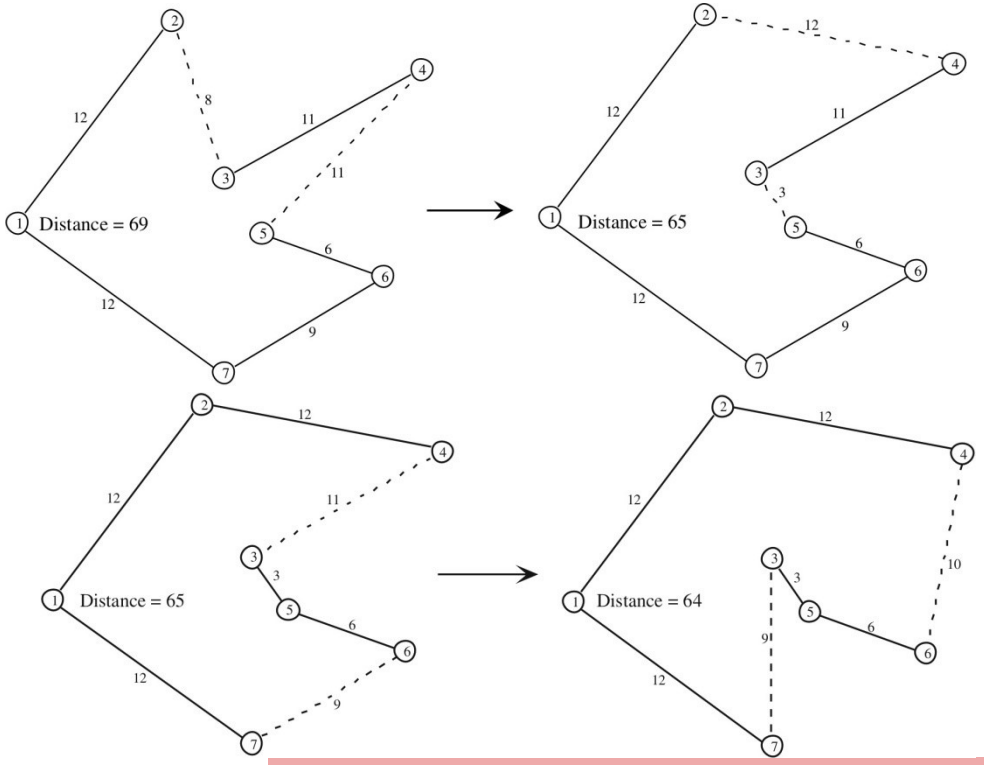
El problema del viajante



Source: Hillier, F. S. (2012). Introduction to operations research. Tata McGraw-Hill Education.

El problema del viajante

- Solución inicial aleatoria
- Estructura de vecindad: *subinversión 2-opt*



1 2 3 4 5 6 7 1
1 2 4 3 5 6 7 1

1 2 4 3 5 6 7 1
1 2 4 6 5 3 7 1

Source: Hillier, F. S. (2012). Introduction to operations research. Tata McGraw-Hill Education.

El problema del viajante

- Selección aleatoria de un vecino

1	2	3	4	5	6	7	1

0.0000–0.1999:	Sub-tour begins in slot 2.
0.2000–0.3999:	Sub-tour begins in slot 3.
0.4000–0.5999:	Sub-tour begins in slot 4.
0.6000–0.7999:	Sub-tour begins in slot 5.
0.8000–0.9999:	Sub-tour begins in slot 6.

Suppose that the random number generated happens to be 0.2779.

0.2779: Choose a sub-tour that begins in slot 3.

0.0000–0.2499:	Sub-tour ends in slot 4.
0.2500–0.4999:	Sub-tour ends in slot 5.
0.5000–0.7499:	Sub-tour ends in slot 6.
0.7500–0.9999:	Sub-tour ends in slot 7.

Suppose that the random number generated for this purpose happens to be 0.0461.

0.0461: Choose to end the sub-tour in slot 4.

1	2	4	3	5	6	7	1

Source: Hillier, F. S. (2012). Introduction to operations research. Tata McGraw-Hill Education.

El problema del viajante

- Secuencia de temperaturas (ad hoc)

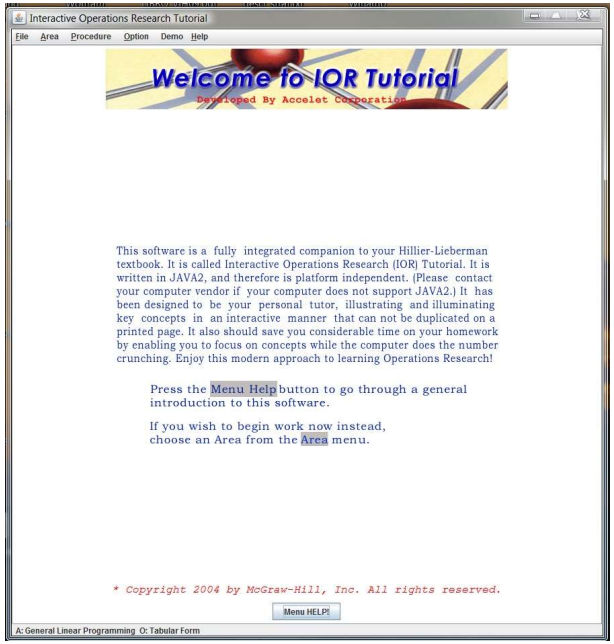
T1=	0,2 Zc
T2=	0,5 T1
T3=	0,5 T2
T4=	0,5 T3
T5=	0,5 T4

Source: Hillier, F. S. (2012). Introduction to operations research. Tata McGraw-Hill Education.

El problema del viajante

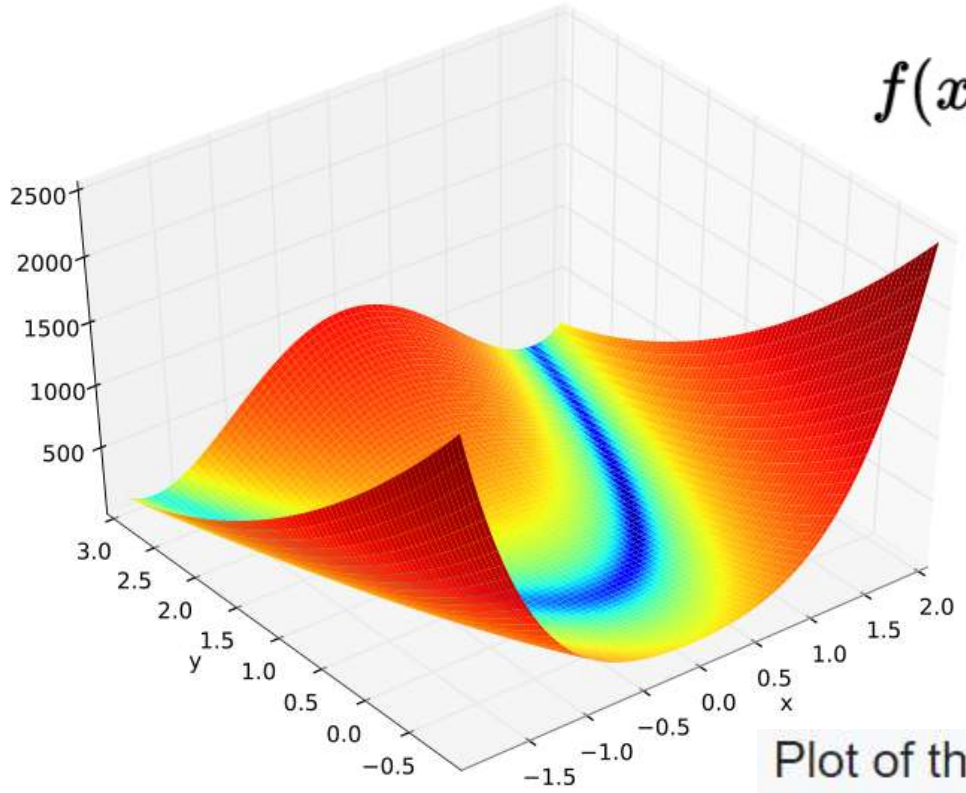
TABLE 14.5 One application of the simulated annealing algorithm in IOR Tutorial to the traveling salesman problem example

Iteration	*T*	Trial Solution Obtained	Distance
0		1-2-3-4-5-6-7-1	69
1	13.8	1-3-2-4-5-6-7-1	68
2	13.8	1-2-3-4-5-6-7-1	69
3	13.8	1-3-2-4-5-6-7-1	68
4	13.8	1-3-2-4-6-5-7-1	65
5	13.8	1-2-3-4-6-5-7-1	66
6	6.9	1-2-3-4-5-6-7-1	69
7	6.9	1-3-2-4-5-6-7-1	68
8	6.9	1-2-3-4-5-6-7-1	69
9	6.9	1-2-3-5-4-6-7-1	65
10	6.9	1-2-3-4-5-6-7-1	69
11	3.45	1-2-3-4-6-5-7-1	66
12	3.45	1-3-2-4-6-5-7-1	65
13	3.45	1-3-7-5-6-4-2-1	66
14	3.45	1-3-5-7-6-4-2-1	63 ← Minimum
15	3.45	1-3-7-5-6-4-2-1	66
16	1.725	1-3-5-7-6-4-2-1	63 ← Minimum
17	1.725	1-3-7-5-6-4-2-1	66
18	1.725	1-3-2-4-6-5-7-1	65
19	1.725	1-2-3-4-6-5-7-1	66
20	1.725	1-3-2-4-6-5-7-1	65
21	0.8625	1-3-7-5-6-4-2-1	66
22	0.8625	1-3-2-4-6-5-7-1	65
23	0.8625	1-2-3-4-6-5-7-1	66
24	0.8625	1-3-2-4-6-5-7-1	65
25	0.8625	1-3-7-5-6-4-2-1	66



Source: Hillier, F. S. (2012). Introduction to operations research. Tata McGraw-Hill Education.

Rosenbrock function



$$f(x, y) = (a - x)^2 + b(y - x^2)^2$$

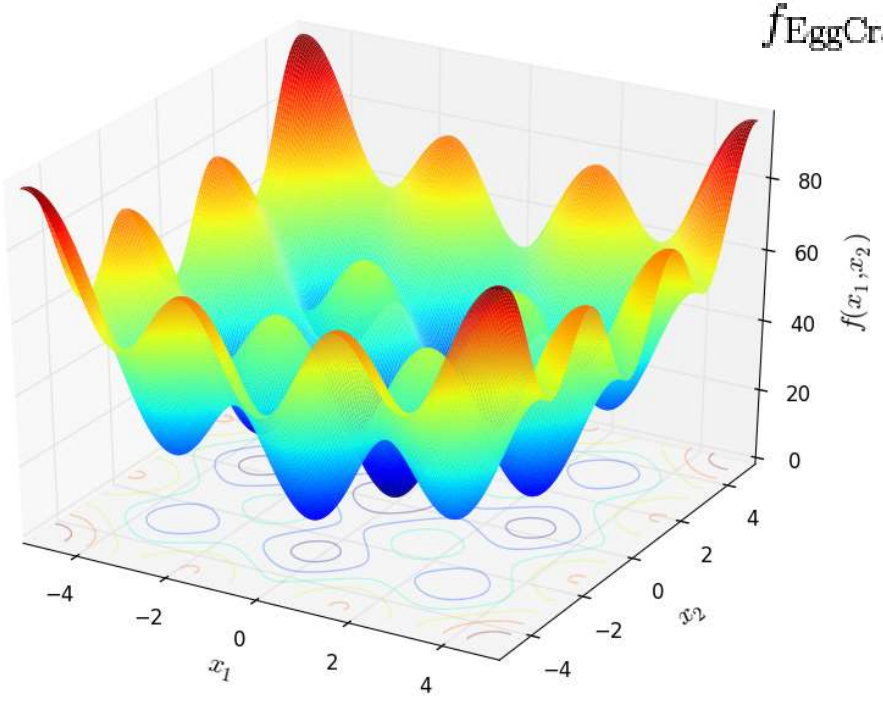
```
% Function evaluations at new locations
  ns=guess+rand(1,2)*randn;
  E_new = f(ns(1),ns(2));
```

Plot of the Rosenbrock function of two variables. Here $a = 1, b = 100$, and the minimum value of zero is at $(1, 1)$.

Source: Yang, X. S. (2010). Engineering optimization: an introduction with metaheuristic applications. John Wiley & Sons.
Image: https://en.wikipedia.org/wiki/Rosenbrock_function

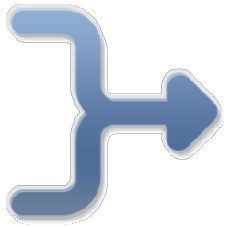
Egg crate function

$$f_{\text{EggCrate}}(\mathbf{x}) = x_1^2 + x_2^2 + 25\left[\sin^2(x_1) + \sin^2(x_2)\right]$$




```
% Function evaluations at new locations
ns=guess+rand(1,2)*randn;
E_new = f(ns(1),ns(2));
```


Source: Yang, X. S. (2010). Engineering optimization: an introduction with metaheuristic applications. John Wiley & Sons.
Image: http://infinity77.net/global_optimization/test_functions_nd_E.html



Resumen

- Recocido
  - Metáfora
- Recocido simulado
  - Algoritmo estocástico de búsqueda por trayectorias que bajo ciertas condiciones permite la degradación de la solución
  - Permite escapar de óptimos locales
  - Sin memoria
- Algoritmo básico
- Implementación
  - Representación
  - Solución inicial
  - Mecanismo de transición entre soluciones
  - Secuencia de enfriamiento
- Uno de los mejores y más populares algoritmos de optimización metaheurística en la industria y la academia