Máster en Sistemas Telemáticos e Informáticos

Optimización de Sistemas de Comunicación

Simulated Annealing

Miguel París Díaz
Moisés Vázquez Sánchez

2011-2012

Índice

1. Introducción

- Algoritmo de búsqueda meta-heurística para problemas de optimización global.
- **Objetivo**: encontrar una buena aproximación al valor óptimo de una función en un espacio de búsqueda grande (óptimo global).
- **Idea**: permitir movimientos que empeoran la función objetivo para escapar de óptimos locales.

1. Introducción

- Se basa en el proceso de recocido del acero y cerámicas:
  - Calentar un sólido cristalino con defectos y luego enfriarlo lentamente.
  - Al aumentar la temperatura los átomos incrementan su energía y pueden así desplazarse de sus posiciones iniciales.
  - El enfriamiento lento permite recristalizar en configuraciones con menor energía que la inicial.

2. Análisis teórico

- En la búsqueda local, se perturba paulatinamente una solución mientras se produce una mejora, de modo que el proceso acaba en un óptimo local.
- Para evitar quedar atrapado en un óptimo local, Simulated Annealing (SA) permite movimientos hacia soluciones peores.
- Estos movimientos de escape deben controlarse adecuadamente para no desviar la búsqueda cuando se dirija hacia una buena solución.
- Para ello se utiliza una función de probabilidad que disminuye la probabilidad de movimientos de escape cuando la búsqueda avanza.

2.1. Algoritmo

- *f(x): coste de la solución* x.
- *N(x): entorno de* x.
- **Inicialización.** Seleccionar:
  1. Solución inicial $\boldsymbol{x_0}$.
  2. Temperatura inicial $\boldsymbol{t_0} > 0$.
  3. Función de reducción de la temperatura $\boldsymbol{\alpha}$.
  4. Número de repeticiones ***nrep***.
  5. Criterio de parada **C**.

2.1. Algoritmo

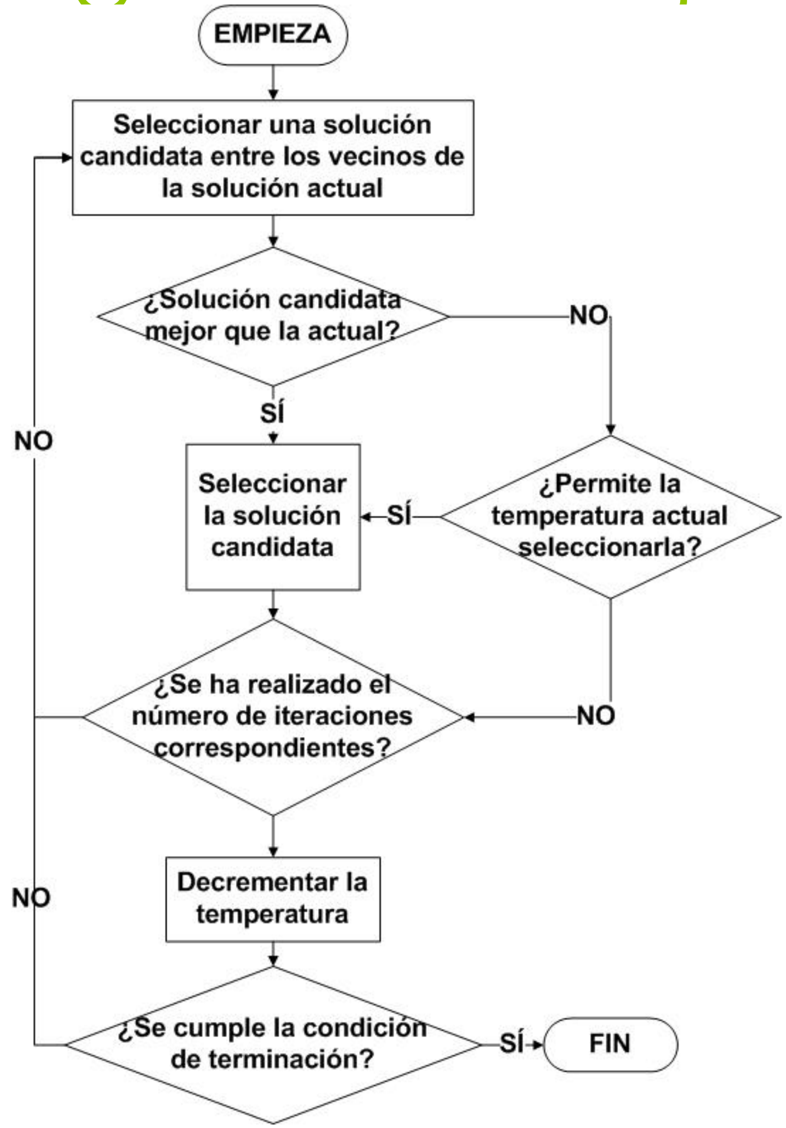
pseudo - Local Search

```
repeat
      repeat
            x := selecRand( N(x0) )
            ΔE := f(x)-f(x0)
            if ΔE < 0 then {x es mejor que x0}
                  x0 := x
            else {x es peor que x0}
                  u := rand(1)
                  if u < exp(-ΔE/t) then
                        x0 := x
                  end if
            end if

            i := i+1
      until i = nrep
      t = α(t)
until C
```

Simulated Annealing

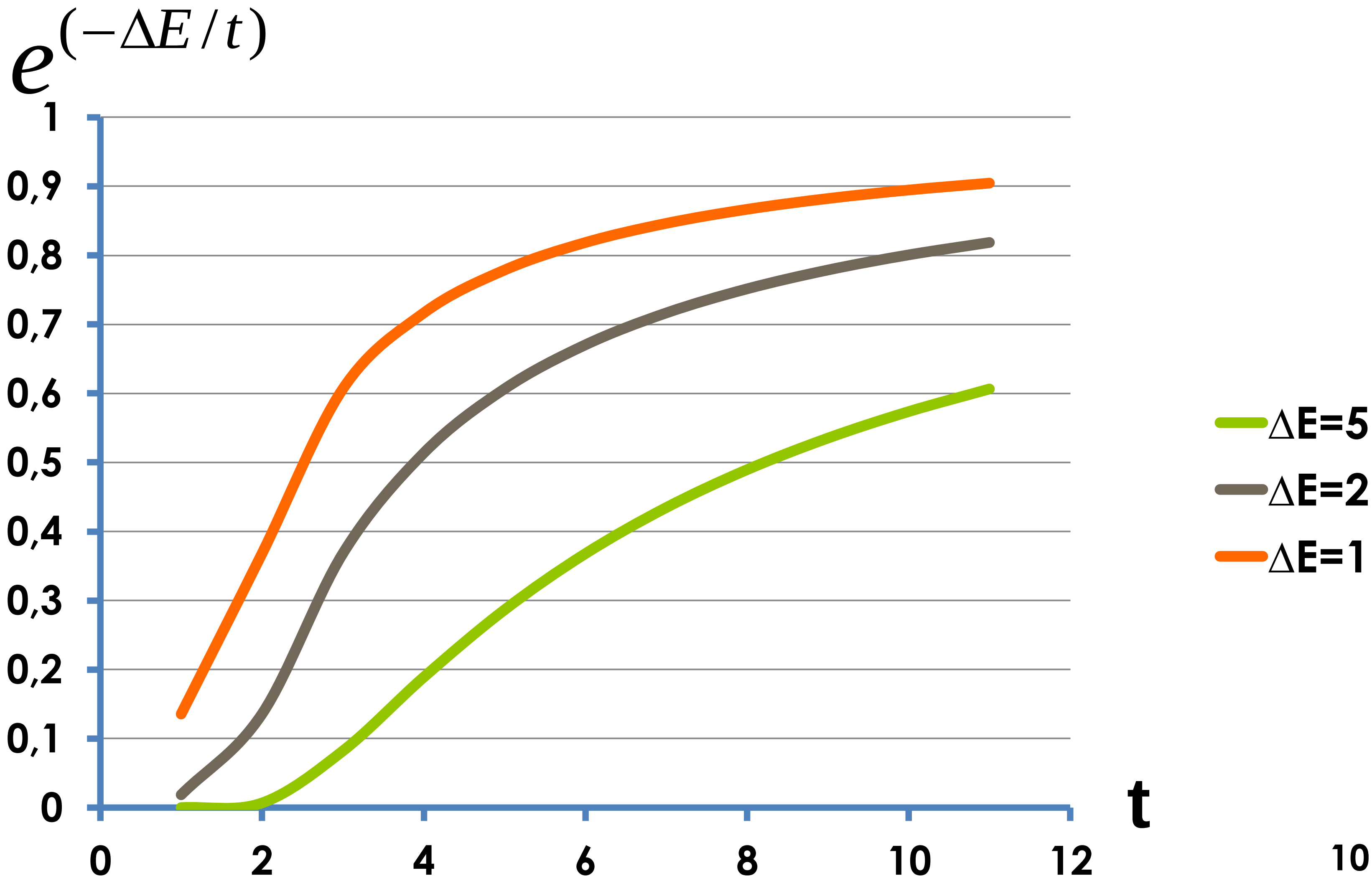
2.2. Diagrama de flujo



2.3. Parámetros

- El comportamiento del algoritmo estará caracterizado fundamentalmente por la parametrización del mismo.
- ***Temperatura (t):***
  - Controla la probabilidad de movimientos de escape.
  - Determina en qué medida se pueden aceptar soluciones vecinas peores que la actual.
- $e^{(-\Delta E/t)}$
  - *A mayor temperatura, mayor probabilidad de aceptación de soluciones peores.*
  - A menor diferencia de costes, mayor probabilidad de aceptación de soluciones peores.

2.3. Parámetros



2.3. Parámetros

- **La temperatura inicial t_0** debe ser:
  - Independiente de la solución inicial.
  - Lo suficientemente alta como para aceptar casi libremente las soluciones el entorno.

- **La temperatura final t_f** debería ser 0, pero en la práctica el proceso converge antes:
  - Si es muy baja, se desaprovecha tiempo.
  - Si es muy alta, no lograremos un óptimo local.

2.3. Parámetros

- La temperatura se va <u>reduciendo</u> en cada iteración mediante un ***mecanismo de enfriamiento*** (**α**).
  - *Se aceptan soluciones mucho peores al principio de la ejecución (exploración) pero no al final (explotación).*
  - *Cuando t es suficientemente pequeña no se producirán movimientos de escape y la búsqueda acabará en un óptimo local.*
  - Diversos métodos.

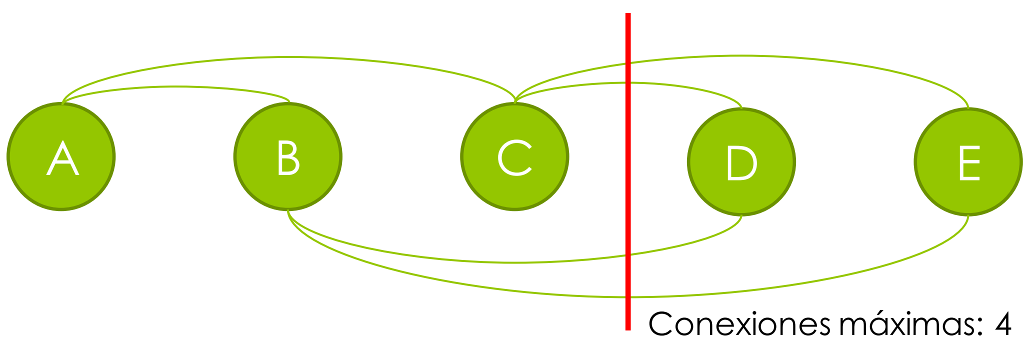
2.3. Parámetros

- Generación de la **solución inicial (x_0)**.
  - Usar técnicas eficientes par obtenerla.
  - Uso de conocimiento experto(ej. algoritmo de greedy).
- **Condición de parada (C):**
  - Cuando t alcanza un t_{final} fijado previamente.
  - Después de un número de iteraciones.

3. Ejemplo práctico

Problema de CutWdith (CutWidth Problem – CWP)

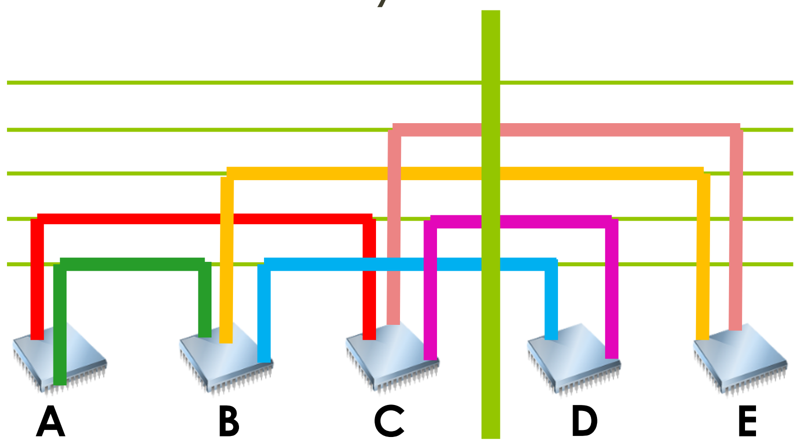
- Dado un grafo, el problema del minimizado del *cutwidth* consiste en encontrar una ordenación lineal del grafo de tal forma que el <u>máximo</u> número de aristas cortadas entre dos vértices consecutivos sea <u>**mínima**</u>.



3. Ejemplo práctico

Caso práctico CutWidth

- Una placa con 5 componentes y 6 conexiones
- *El objetivo es **minimizar el número de líneas de conexión en circuitos electrónicos***
- *Se conocen las conexiones entre los componentes del circuito (con una tabla de conexiones)*



Conexiones máximas: 4

	A	B	C	D	E
A	-	1	1	0	0
B	1	-	0	1	1
C	1	0	-	1	1
D	0	1	1	-	0
E	0	1	1	0	-

Tabla de conexiones

3. Ejemplo práctico

Ejemplo de solución para el CutWith

- N = 5 componentes y M = 6 conexiones.
- Un ejemplo de solución sería $\mathrm{S_{act}} = \{\mathrm{A, B, D, C, E}\}$
- Una representación para esta solución sería:

1	2	3	4	5
A	B	D	C	E

- Valor:
$$Coste(S_{act}) = \{\max(c(A, B), c(B, C), c(C, D), c(D, E)\}$$

$$Coste(S_{act}) = \mathbf{4}$$

3. Ejemplo práctico

Vecindad:

- Calculamos la vecindad formada por todos los posibles intercambios del componente (c = D) seleccionado aleatoriamente, con todos los demás.

	1	2	3	4	5
Componente	A	B	C	D	E

	1	2	3	4	5
Vecindad 1	D	B	A	E	D

	1	2	3	4	5
Vecindad 2	D	B	A	C	E

	1	2	3	4	5
Vecindad 3	C	D	A	B	E

	1	2	3	4	5
Vecindad 4	C	B	D	A	E

3. Ejemplo práctico

Coste de la Vecindad:

- Calculamos el valor del coste asociado a cada elemento de la vecindad.

	1	2	3	4	5	
Componente	A	B	C	D	E	Coste = 4
Vecino 1	D	B	C	A	E	Coste = 4
Vecino 2	A	D	C	B	E	Coste = 4
Vecino 3	A	B	D	C	E	**¡Coste = 3!**
Vecino 4	A	B	C	E	D	Coste = 4

3. Ejemplo práctico

Evaluamos la vecindad:

- Evaluamos la función de coste de todos los vecinos.
- Comenzamos a evaluar vecinos
- Proseguimos con la temperatura **igual** hasta que no evaluemos toda la vecindad.
- Si encontramos un vecino mejor a la solución inicial le asignamos automáticamente, si no evaluamos si se puede aceptar (dependiendo de la temperatura).

3. Ejemplo práctico

Evaluamos el vecino elegido:

- Función de coste anterior (4 conexiones)
- Función de coste nueva para el intercambio (3 conexiones)
- Se selecciona *automáticamente* ya que el coste para la nueva solución sería **menor**. Nuestra solución actual sería (Vecino 3):

1	2	3	4	5
A	B	D	C	E

Coste = 3

3. Ejemplo práctico

Vecindad - Coste de la nueva iteración:

- Calculamos la vecindad y el valor del coste asociado a los vecinos de esta iteración (elección aleatoria del componente pivote c = B)

	1	2	3	4	5	
Componente	A	B	D	C	E	**Coste = 3**
Vecino 1	B	A	D	C	E	**Coste = 3**
Vecino 2	A	D	B	C	E	**Coste = 4**
Vecino 3	A	C	D	B	E	**Coste = 3**
Vecino 4	A	E	D	C	B	**Coste = 6**

3. Ejemplo práctico

Evaluamos el vecino elegido:

- Función de coste anterior (3 conexiones)
- Función de coste nueva para el intercambio (6 conexiones)
  - **Se podría aceptar o no dependiendo de:** $e^{(-\delta/T)}$. Suponemos que se acepta para la temperatura actual.
  - Aunque sea una solución **peor**, el hecho de tener una temperatura T_i (inicialmente grande) nos permite aceptar, con una probabilidad dada, esta solución peor (evitamos caer en un **mínimo local**).
  - Continuaremos con el algoritmo para el número de repeticiones dadas y disminuiremos la temperatura.

1	2	3	4	5
A	E	D	C	B

Coste = 6

3. Ejemplo práctico

Evaluamos el vecino elegido:

- Una vez que hemos salido de ese mínimo local, seguiremos con el algoritmo para el número de repeticiones establecidas (C).
- Una vez la temperatura llegue a T_f (temperatura final) terminaremos el algoritmo y nos quedaremos con la solución actual , que será una buena solución. Aunque no se puede asegurar que sea el valor óptimo del problema.

4. Dudas y preguntas

Simulated Annealing

- Miguel París Díaz
- Moisés Vázquez Sánchez

Máster en Sistemas Telemáticos e Informáticos

2011-2012

Optimización de Sistemas de Comunicación

Simulated Annealing

Miguel París Díaz
Moisés Vázquez Sánchez