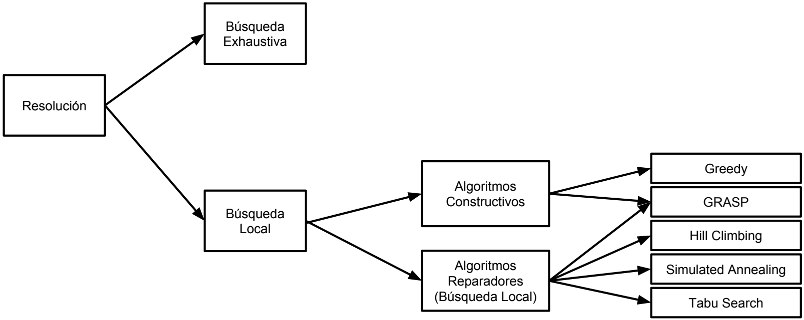
Inteligencia Artificial
Técnicas Incompletas de Búsqueda de Soluciones: Búsqueda Tabú y Simulated Annealing

Nicolás Rojas Morales

Departamento de Informática
Universidad Técnica Federico Santa María

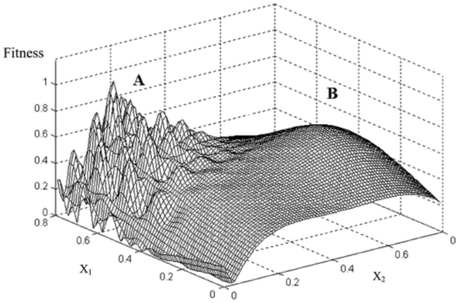
Clasificación de Técnicas de Resolución



Escape de óptimos locales

Problema

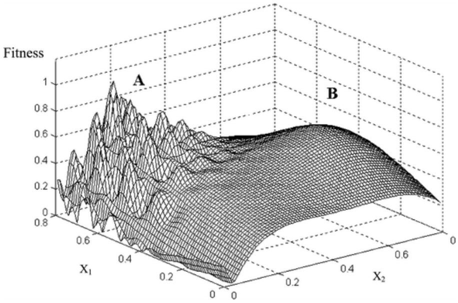
- Estancamiento de la búsqueda en óptimos locales



Escape de óptimos locales

Problema

- Estancamiento de la búsqueda en óptimos locales



Solución

- Opción 1: Reiniciar la búsqueda $\rightarrow$ pérdida de información
- Opción 2: Aceptar movimientos que empeoran la calidad de la solución actual
  - Pueden producir ciclos en la búsqueda.

Búsqueda Tabú (Tabu Search)

Búsqueda Tabú (Tabu Search)

- Búsqueda Tabú (Glover, 86) es una estrategia para resolver problemas de optimización combinatoria. Algo muy parecido sugirió Hansen al mismo tiempo *(steepest ascent/mildest descent)*.
- Utiliza una búsqueda local con memoria a corto plazo
  - El proceso es capaz de escapar de óptimos locales y evitar ciclos utilizando la información almacenada en la memoria.
- La memoria de corto plazo esta representada por la lista tabú, la cual registra las últimas soluciones visitadas e impide volver a ellas en los próximos movimientos.

Lista Tabú

- Por razones de eficiencia, no se guarda la solución completa en la lista tabú, sino una parte de sus atributos. Se define una lista tabú de condiciones.
- La lista tabú se actualiza normalmente en forma FIFO.
- El largo de la lista tabú controla la memoria del proceso de búsqueda.
  - Una lista tabú corta, controla áreas reducidas del espacio de búsqueda y una larga, fuerza a una búsqueda en áreas mayores.

Lista Tabú

- Por razones de eficiencia, no se guarda la solución completa en la lista tabú, sino una parte de sus atributos. Se define una lista tabú de condiciones.
- La lista tabú se actualiza normalmente en forma FIFO.
- El largo de la lista tabú controla la memoria del proceso de búsqueda.
  - Una lista tabú corta, controla áreas reducidas del espacio de búsqueda y una larga, fuerza a una búsqueda en áreas mayores.
- El largo de la lista puede cambiar a lo largo del proceso de búsqueda.

Pseudo-código

Procedure tabu search
initialize s_c at random
initialize $tabu_{list}$ as empty
initialize $s_{best} \leftarrow s_c$
Repeat
$s_v \leftarrow$select from $\mathcal{N}(s_c)$ the best non tabu point
$s_c \leftarrow s_v$
update $tabu_{list}$
If $f(s_c)$ is better than $f(s_{best})$ **then**
$s_{best} \leftarrow s_c$
Until $terminationCriterion()$
EndProcedure

Ejemplo

- Considerando el problema de la mochila

$$\begin{aligned} \text{máx } & 18 \cdot x_1 + 25 \cdot x_2 + 11 \cdot x_3 + 14 \cdot x_4 \\ \text{s.a. } & 2 \cdot x_1 + 2 \cdot x_2 + x_3 + x_4 \leq 3 \\ & x_1, x_2, x_3, x_4 \in \{0, 1\} \end{aligned}$$

- Proponga una representación acorde al problema
- Proponga un movimiento acorde al problema
- Describa el proceso de búsqueda realizado por Búsqueda Tabú.
  - Criterio de parada: 5 iteraciones
  - Largo de la lista: 2

Ejemplo: Búsqueda Tabú

- Representación binaria de largo n, donde n corresponde a la cantidad de objetos.

1	0	0	0

- Movimiento: Cambio de una variable desde 0 a 1, ó desde 1 a 0
- Observación: Se trabaja sólo con soluciones factibles

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		$\{\}$	

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		$\{\}$	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		$\{\}$	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		
					$j4$

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		$\{\}$	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		
					$j4$
2	100**1**(32)	1001(32)		$\{j4\}$	

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		{}	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		
					$j4$
2	100**1**(32)	1001(32)		$\{j4\}$	
			0001(14)		
			~~1101(57)~~		
			~~1011(43)~~		
			1000(18)		

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		{}	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		
					$j4$
2	100**1**(32)	1001(32)		$\{j4\}$	
			0001(14)		
			~~1101(57)~~		
			~~1011(43)~~		
			1000(18)		
					$j1$

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		{}	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		
					$j4$
2	100**1**(32)	1001(32)		$\{j4\}$	
			0001(14)		
			~~1101(57)~~		
			~~1011(43)~~		
			1000(18)		
					$j1$
3	**0**001(14)	1001(32)		$\{j4, j1\}$	

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		{}	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		
					$j4$
2	100**1**(32)	1001(32)		$\{j4\}$	
			0001(14)		
			~~1101(57)~~		
			~~1011(43)~~		
			1000(18)		
					$j1$
3	**0**001(14)	1001(32)		$\{j4, j1\}$	
			1001(14)		
			0101(39)		
			0011(25)		
			0000(0)		

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		{}	
			0000(18)		
			~~1100(43)~~		
			1010(29)		
			1001(32)		
					$j4$
2	100**1**(32)	1001(32)		$\{j4\}$	
			0001(14)		
			~~1101(57)~~		
			~~1011(43)~~		
			1000(18)		
					$j1$
3	**0**001(14)	1001(32)		$\{j4, j1\}$	
			1001(14)		
			0101(39)		
			0011(25)		
			0000(0)		
					$j2$

Ejemplo: Búsqueda Tabú (2)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
1	1000(18)	1000(18)		$\{\}$	
			0000(18)		
			$\cancel{1100(43)}$		
			1010(29)		
			1001(32)		
					$j4$
2	100**1**(32)	1001(32)		$\{j4\}$	
			0001(14)		
			$\cancel{1101(57)}$		
			$\cancel{1011(43)}$		
			1000(18)		
					$j1$
3	**0**001(14)	1001(32)		$\{j4, j1\}$	
			1001(14)		
			0101(39)		
			0011(25)		
			0000(0)		
					$j2$
4	0**1**01(39)	0101(39)		$\{j1, j2\}$	

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
4	$0\mathbf{1}01(39)$	$0101(39)$		$\{j1, j2\}$	

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
4	0**1**01(39)	0101(39)		$\{j1, j2\}$	
			~~1101(57)~~		
			0001(14)		
			~~0111(50)~~		
			0100(25)		

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
4	0**1**01(39)	0101(39)		$\{j1, j2\}$	
			~~1101(57)~~		
			0001(14)		
			~~0111(50)~~		
			0100(25)		
					$j4$

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
4	0**1**01(39)	0101(39)		$\{j1, j2\}$	
			~~1101(57)~~		
			0001(14)		
			~~0111(50)~~		
			0100(25)		
					$j4$
5	010**0**(25)	0101(39)		$\{j2, j4\}$	

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
4	0**1**01(39)	0101(39)		$\{j1, j2\}$	
			~~1101(57)~~		
			0001(14)		
			~~0111(50)~~		
			0100(25)		
					$j4$
5	010**0**(25)	0101(39)		$\{j2, j4\}$	
			~~1100(43)~~		
			~~0000(0)~~		
			0110(36)		
			0101(39)		

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
4	0**1**01(39)	0101(39)		$\{j1, j2\}$	
			~~1101(57)~~		
			0001(14)		
			~~0111(50)~~		
			0100(25)		
					$j4$
5	010**0**(25)	0101(39)		$\{j2, j4\}$	
			~~1100(43)~~		
			~~0000(0)~~		
			0110(36)		
			0101(39)		
					$j3$

Ejemplo: Búsqueda Tabú (3)

it	s_c	s_{best}	$\mathcal{N}(s_c)$	$tabu_{list}$	mov. tabú
4	0**1**01(39)	0101(39)		$\{j1, j2\}$	
			~~1101(57)~~		
			0001(14)		
			~~0111(50)~~		
			0100(25)		
					$j4$
5	010**0**(25)	0101(39)		$\{j2, j4\}$	
			~~1100(43)~~		
			~~0000(0)~~		
			0110(36)		
			0101(39)		
					$j3$
6	01**1**0(36)	0101(39)		$\{j4, j3\}$	

Intensificación y Diversificación en Búsqueda Tabú

- Intensificación: Búsqueda local.
- Diversificación: Aceptando soluciones de peor calidad.
- Balance estático a través del tamaño de la lista tabú.
  - Lista tabú larga: fomenta la diversificación a zonas más alejadas del espacio de búsqueda.

Intensificación y Diversificación en Búsqueda Tabú

- Intensificación: Búsqueda local.
- Diversificación: Aceptando soluciones de peor calidad.
- Balance estático a través del tamaño de la lista tabú.
  - Lista tabú larga: fomenta la diversificación a zonas más alejadas del espacio de búsqueda.
- Cambios dinámicos en el largo de la lista tabú (Reactive Tabu Search).
  - El largo de la lista varía según las propiedades de la trayectoria en el espacio de búsqueda.

Criterio de aspiración

- En el proceso se pierde información y buenas soluciones pueden ser excluidas del conjunto permitido.
- Para reducir este problema, se define un *criterio de aspiración* que permitiría a una solución estar dentro del conjunto de soluciones permitidas aún cuando figure en la lista tabú.

Pseudo-código

Procedure tabu search
    initialize s_c at random
    initialize $tabu_{list}$ as empty
    initialize $s_{best} \leftarrow s_c$
    Repeat
        s_v $\leftarrow$select from $\mathcal{N}(s_c)$ the best point that satisfies the aspiration criteria **or** the best non tabu point
        $s_c \leftarrow s_v$
        update $tabu_{list}$
        If $f(s_c)$ is better than $f(s_{best})$ **then**
            $s_{best} \leftarrow s_c$
    Until $terminationCriterion()$
EndProcedure

Inteligencia Artificial
Técnicas Incompletas de Búsqueda de Soluciones: Búsqueda Tabú y Simulated Annealing

Nicolás Rojas Morales

Departamento de Informática
Universidad Técnica Federico Santa María

Simulated Annealing (Recocido Simulado)

Simulated Annealing (Recocido Simulado)

- Se basa en el trabajo de Metropolis et al. (1953) en el campo de la termodinámica estadística.
- Modelamiento del proceso de recocido simulando los cambios energéticos en un sistema de partículas conforme decrece la temperatura, hasta que converge a un estado estable (congelado).

Idea:

- Permitir movimientos a soluciones que empeoren la función de evaluación de forma de poder escapar de los óptimos locales.
- Permitir movimientos que empeoran la calidad de la función de evaluación de acuerdo a una probabilidad asociada a la temperatura del sistema.
- Al comienzo, la temperatura es alta y cualquier transición entre estados es permitida y soluciones que empeoren la función de evaluación pueden ser aceptadas con mayor probabilidad que más tarde cuando la temperatura disminuye.
- Mientras que las soluciones que mejoran la función de evaluación siempre son aceptadas.

Pseudo-código I

Procedure simulated annealing
    $t \leftarrow 0$
    initialize T
    initialize s_c at random
    initialize $s_{best} \leftarrow s_c$
    Repeat
        Repeat
            $s_n \leftarrow$select a new point in $\mathcal{N}(s_c)$
            If $f(s_n)$ is better than $f(s_c)$ **then**
                $s_c \leftarrow s_n$
            Else If $random([0,1]) < e^{\frac{\Delta_{eval}}{T}}$ **then**
                $s_c \leftarrow s_n$
            If $f(s_c)$ is better than $f(s_{best})$ **then**
                $s_{best} \leftarrow s_c$
        Until $haltingCriterion()$
        $T \leftarrow g(T,t)$
        $t \leftarrow t+1$
    Until $terminationCriterion()$
EndProcedure

Probabilidad de Aceptación y Temperatura

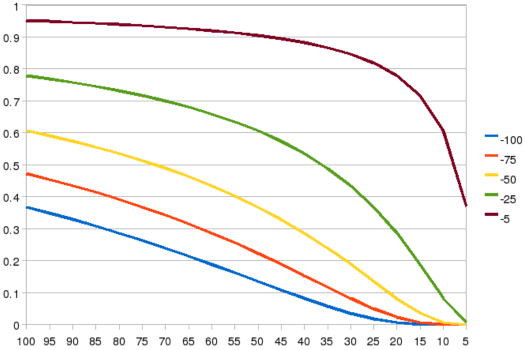
P(temperatura, solución nueva, solución actual) = P(.)

$$P(.) = e^{(\Delta_{eval}/temperatura)} \quad (1)$$

donde
Δ_{eval} = Se considera siempre como un valor negativo, pues implica un empeoramiento de la calidad de las soluciones.

Acerca de la Temperatura:

- Temperatura versus Probabilidad de aceptación
- Cada línea muestra un valor distinto de Δ_{eval}



Observaciones

- La temperatura del sistema es controlada con enfriamientos sucesivos (función logarítmica) y recalentamientos periódicos, que permiten escapar de óptimos locales.
- Enfriamiento: Cada cierto número de iteraciones

$$T_{i+1} = \alpha \cdot T_i \tag{2}$$

donde $0 < \alpha < 1$, se recomienda utilizar $\alpha \in \{0.8; 0.99\}$

Diversificación e Intensificación en Simulated Annealing

- Intensificación: Búsqueda local.
- Balance dinámico controlado a través del valor de la temperatura: Inicialmente alta diversificación y al final del algoritmo baja diversificación.
  - Alta Temperatura: gran diversificación
  - Baja Temperatura: poca diversificación.

Diversificación e Intensificación en Simulated Annealing

- Intensificación: Búsqueda local.
- Balance dinámico controlado a través del valor de la temperatura: Inicialmente alta diversificación y al final del algoritmo baja diversificación.
  - Alta Temperatura: gran diversificación
  - Baja Temperatura: poca diversificación.
- Diversificación: Técnicas de recalentado y posterior enfriamiento (enfriamiento no monótono).

Ejemplo

- Considerando el problema de la mochila

$$\begin{aligned} \text{máx } 18 \cdot x_1 + 25 \cdot x_2 + 11 \cdot x_3 + 14 \cdot x_4 \\ \text{s.a. } 2 \cdot x_1 + 2 \cdot x_2 + x_3 + x_4 \leq 3 \\ x_1, x_2, x_3, x_4 \in \{0, 1\} \end{aligned}$$

- Proponga una representación acorde al problema
- Proponga un movimiento acorde al problema
- Describa el proceso de búsqueda realizado por Simulated Annealing.
  - Criterio de parada: 4 iteraciones
  - Temperatura Inicial: t=10 (fija durante las 4 iteraciones)

Ejemplo: Simulated Annealing

- Representación binaria de largo n, donde n corresponde a la cantidad de objetos.

1	0	0	0

- Movimiento: Cambio de una variable desde 0 a 1, ó desde 1 a 0
- Observación: Se trabaja sólo con soluciones factibles

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				j1 ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				j1 ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado
3	1001(32)	1001(32)	10				

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó $\mathbf{j4}$	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				$\mathbf{j1}$ ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado
3	1001(32)	1001(32)	10				
				$j1$ ó $j4$			

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				j1 ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado
3	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				$j1$ ó **j4**	-14	$e^{-14/10} = 0.247$	aceptado

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				j1 ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado
3	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				$j1$ ó **j4**	-14	$e^{-14/10} = 0.247$	aceptado
4	1000(18)	1001(32)	10				

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				j1 ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado
3	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				$j1$ ó **j4**	-14	$e^{-14/10} = 0.247$	aceptado
4	1000(18)	1001(32)	10				
				$j1$, $j3$ ó $j4$			

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				j1 ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado
3	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				$j1$ ó **j4**	-14	$e^{-14/10} = 0.247$	aceptado
4	1000(18)	1001(32)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, **j3** ó $j4$	11		aceptado

Ejemplo: Simulated Annealing (2)

Considere la siguiente secuencia de números aleatorios:
0.72; 0.33; 0.41; 0.83; 0.23; 0.64 ; ...

it	s_c	s_{best}	T	mov.	Δ_{eval}	$e^{\Delta_{eval}/T}$	condición
1	1000(18)	1000(18)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, $j3$ ó **j4**	14		aceptado
2	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				j1 ó $j4$	-18	$e^{-18/10} = 0.165$	rechazado
3	1001(32)	1001(32)	10				
				$j1$ ó $j4$			
				$j1$ ó **j4**	-14	$e^{-14/10} = 0.247$	aceptado
4	1000(18)	1001(32)	10				
				$j1$, $j3$ ó $j4$			
				$j1$, **j3** ó $j4$	11		aceptado
5	1010(25)	1001(32)	10				

Simulated Annealing para TSP

- Representación: Xi la i-ésima ciudad visitada
- Función de Evaluación: Largo del tour
- Movimiento: Intercambio de 2 arcos (2-opt)
- Criterio de selección de los dos arcos: Aleatorio
- Solución inicial: Aleatoria factible.

Simulated Annealing para TSP

- Representación: Xi la i-ésima ciudad visitada
- Función de Evaluación: Largo del tour
- Movimiento: Intercambio de 2 arcos (2-opt)
- Criterio de selección de los dos arcos: Aleatorio
- Solución inicial: Aleatoria factible.
- Criterio de Termino: Máximo número de iteraciones o estancamiento en óptimo local
- Número de pasos: 3
- Maxiter: 4

Simulated Annealing para TSP

Procedure SA_TSP
Begin
  f(Mejor_solucion) = MAXVALUE
  Solucion_actual = Generar_solucion_inicial
  For i=1 to steps
    iteraciones=0;
    While(iteraciones<Maxiter and no_optimo_local)
      Arco1 = random(1,ciudades)
      Arco2 = random(1,ciudades)
      Candidata_solucion=2-opt(Solucion_actual, Arco1, Arco2)
      If f(Candidata_solucion)<f(Solucion_actual) **then**
        Solucion_actual=Candidata_solucion
      Else If $random([0,1]) < e^{\frac{f(Solucion\_actual) - f(Candidata\_solucion)}{T}}$ **Then**
        Solucion_actual=Candidata_solucion
      iteraciones++
      If f(Solucion_actual)<f(Mejor_solucion) **then**
        Mejor_solucion=Solucion_actual
      iteraciones++
    EndWhile
    T = α*T
  EndFor

Inteligencia Artificial
Técnicas Incompletas de Búsqueda de Soluciones: Búsqueda Tabú y Simulated Annealing

Nicolás Rojas Morales

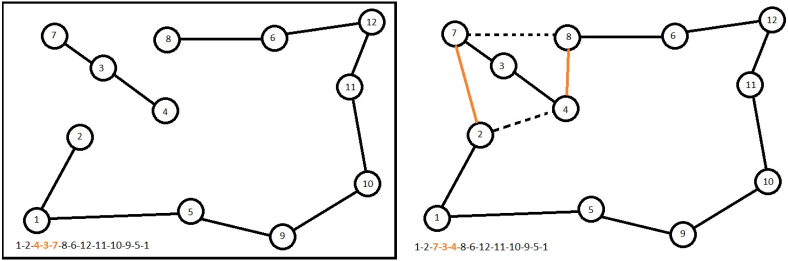
Departamento de Informática
Universidad Técnica Federico Santa María

Heurísticas K-opt

Heurísticas K-opt

- Definición K-opt (K-exchange): Es un procedimiento que reemplaza K arcos de un tour de TSP por K nuevos arcos, tal que el tour resultante es un tour factible del TSP.

2-opt



2-opt vs intercambio

- En el tour:
  Tour1: A - B - C - D - E - F - G - H - I - J
  El tour resultante después de swap es:
  Tour2: A - B - H - D - E - F - G - C - I - J

2-opt vs intercambio

- En el tour:
  Tour1: A - B - C - D - E - F - G - H - I - J
  El tour resultante después de swap es:
  Tour2: A - B - H - D - E - F - G - C - I - J

  En términos de costo, la nueva solución difiere en cuatro arcos respecto a la original:

  $$f(\text{Tour2}) = f(\text{Tour1}) - (d(B,C) + d(C,D) + d(G,H) + d(H,I)) \\ + (d(B,H) + d(H,D) + d(G,C) + d(C,I))$$

2-opt vs intercambio

- En el tour:
  Tour1: A - B - C - D - E - F - G - H - I - J
  El tour resultante después de swap es:
  Tour2: A - B - H - D - E - F - G - C - I - J

  En términos de costo, la nueva solución difiere en cuatro arcos respecto a la original:
  f(Tour2) = f(Tour1) - (d(B,C) + d(C,D) + d(G,H) + d(H,I))
  + (d(B,H) + d(H,D) + d(G,C) + d(C,I))
- En el tour:
  Tour1: A - B - C - D - E - F - G - H - I - J
  El tour resultante después de 2-opt es:
  Tour3: A - B - H - G - F - E - D - C - I - J

2-opt vs intercambio

- En el tour:
  Tour1: A - B - C - D - E - F - G - H - I - J
  El tour resultante después de swap es:
  Tour2: A - B - H - D - E - F - G - C - I - J

  En términos de costo, la nueva solución difiere en cuatro arcos respecto a la original:
  $$f(Tour2) = f(Tour1) - (d(B,C) + d(C,D) + d(G,H) + d(H,I)) + (d(B,H) + d(H,D) + d(G,C) + d(C,I))$$
- En el tour:
  Tour1: A - B - C - D - E - F - G - H - I - J
  El tour resultante después de 2-opt es:
  Tour3: A - B - H - G - F - E - D - C - I - J

  En términos de costo, la nueva solución difiere en dos arcos respecto a la original: $f(Tour3) = f(Tour1) - (d(B,C) + d(H,I)) + (d(B,H) + d(C,I))$

3-opt