


Facultad de Ciencias
Licenciatura en
Ciencias de la Computación

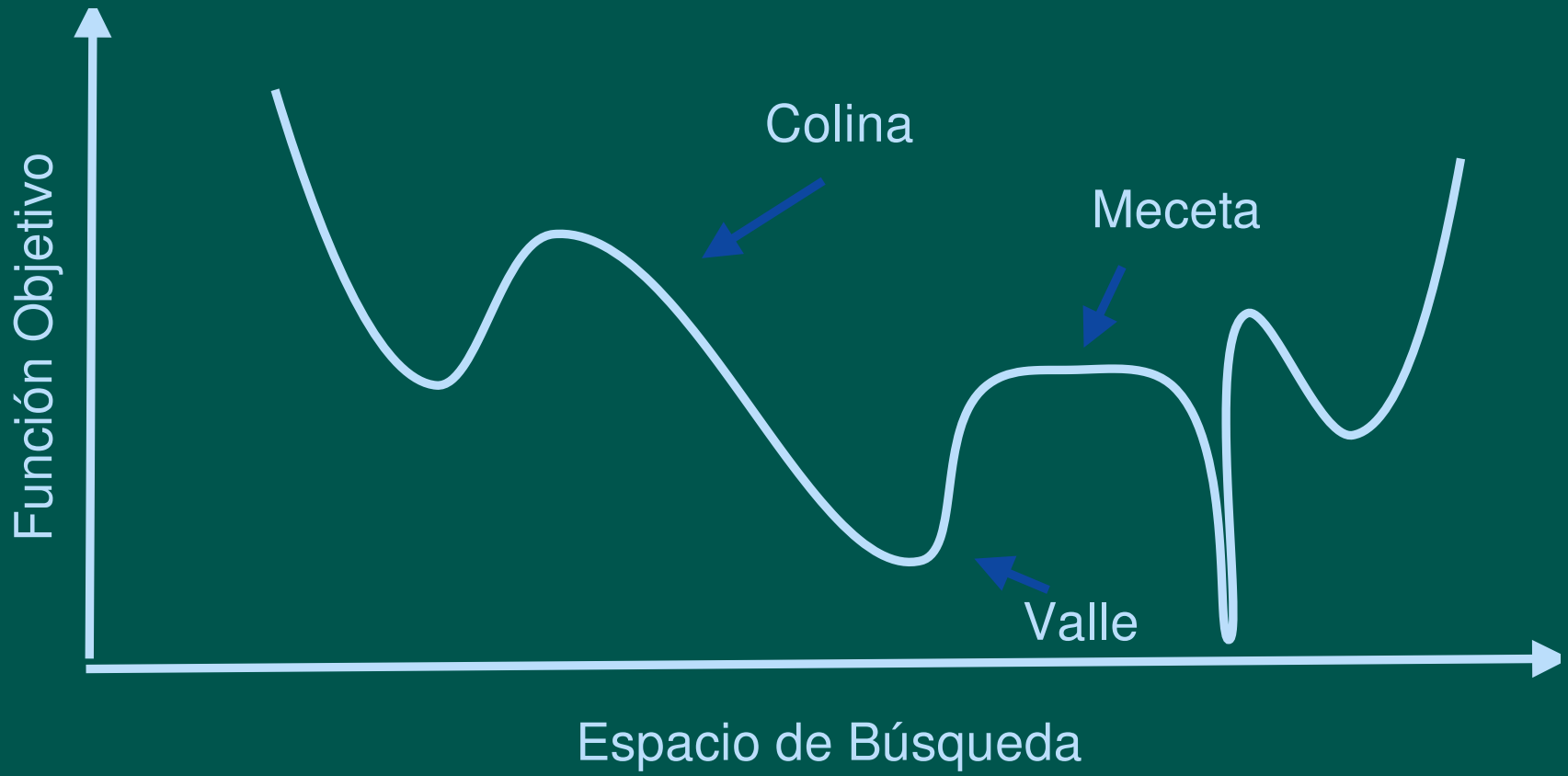
Cómputo Evolutivo

Búsqueda por Escalada

M. en C. Oscar Hernández Constantino
(constantino92@ciencias.unam.mx)

Contenido de la Presentación

Búsqueda por Escalada (Hill Climbing)


Función Objetivo
Colina
Meceta
Valle
Espacio de Búsqueda


Función Objetivo
Solución Inicial
Óptimo Local
Espacio de Búsqueda

Algoritmo 1: Búsqueda por Escalada (Hill Climbing)

Entrada: Una solución $s_0 \in S$, función de vecindad $N : S \longrightarrow 2^S$, función objetivo $f :\longrightarrow \mathbb{R}$

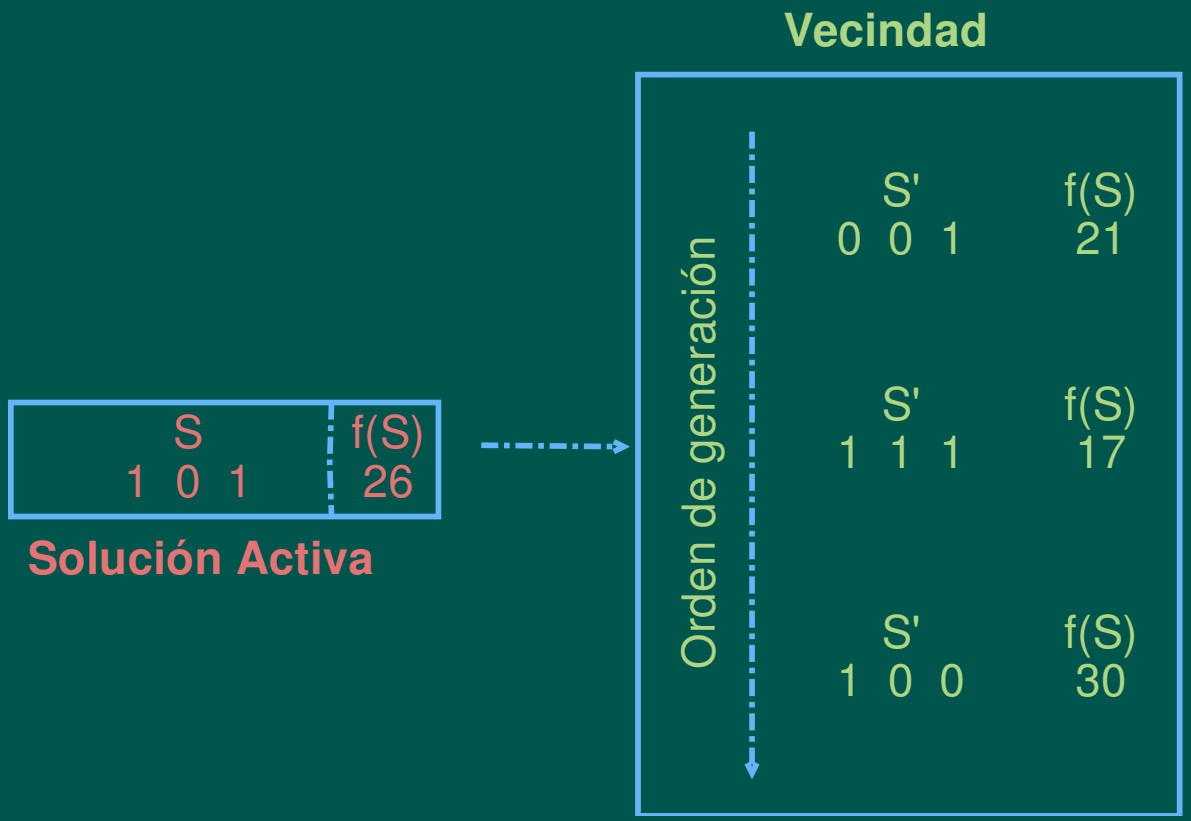
Resultado: Una solución s' que es un óptimo local respecto a N

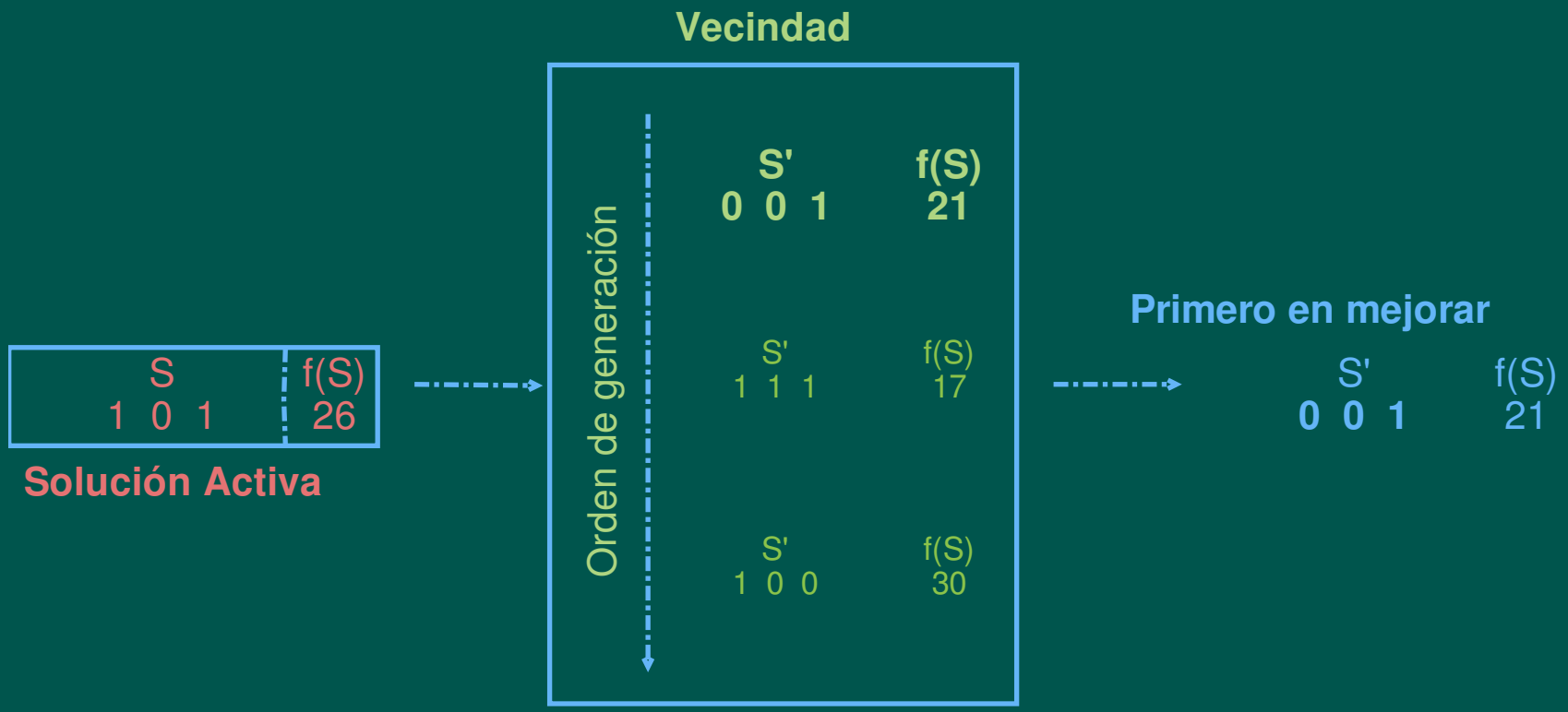
```
1  s = s0 ;
   // s0 = generarSolucionInicial( )
2  mientras Condición de Término hacer
3      /* Generación de candidatos (vecindad), N(s)        */ N(s) =
       generarVecinos(s) ;
4
5      si ∄ f(s') < f(s) ∀s' ∈ N(s) entonces
6          Detener ejecución
7      en otro caso
8          Seleccionar a un vecino s' ∈ N(s) tal que f(s') < f(s) ;
9          s = s' ;
10
11 devolver s
```

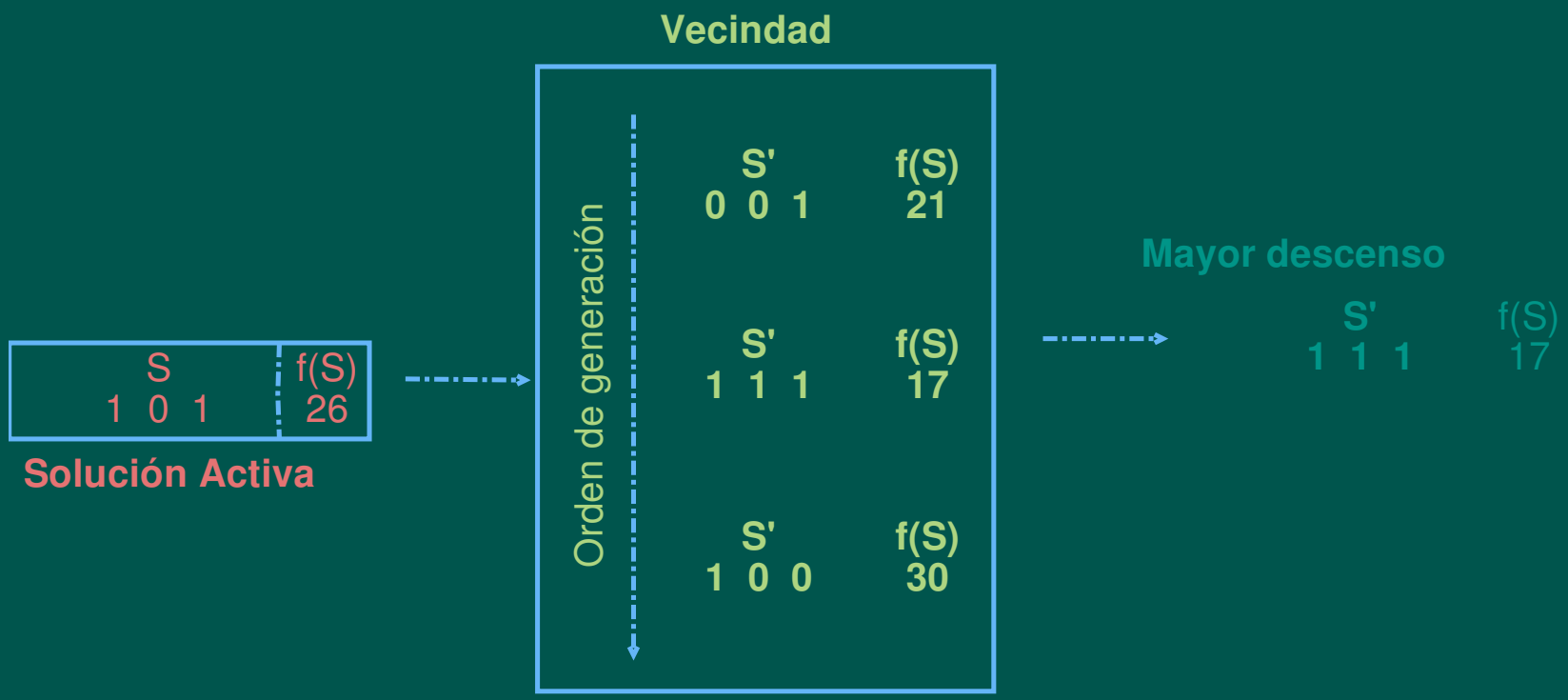
Variantes

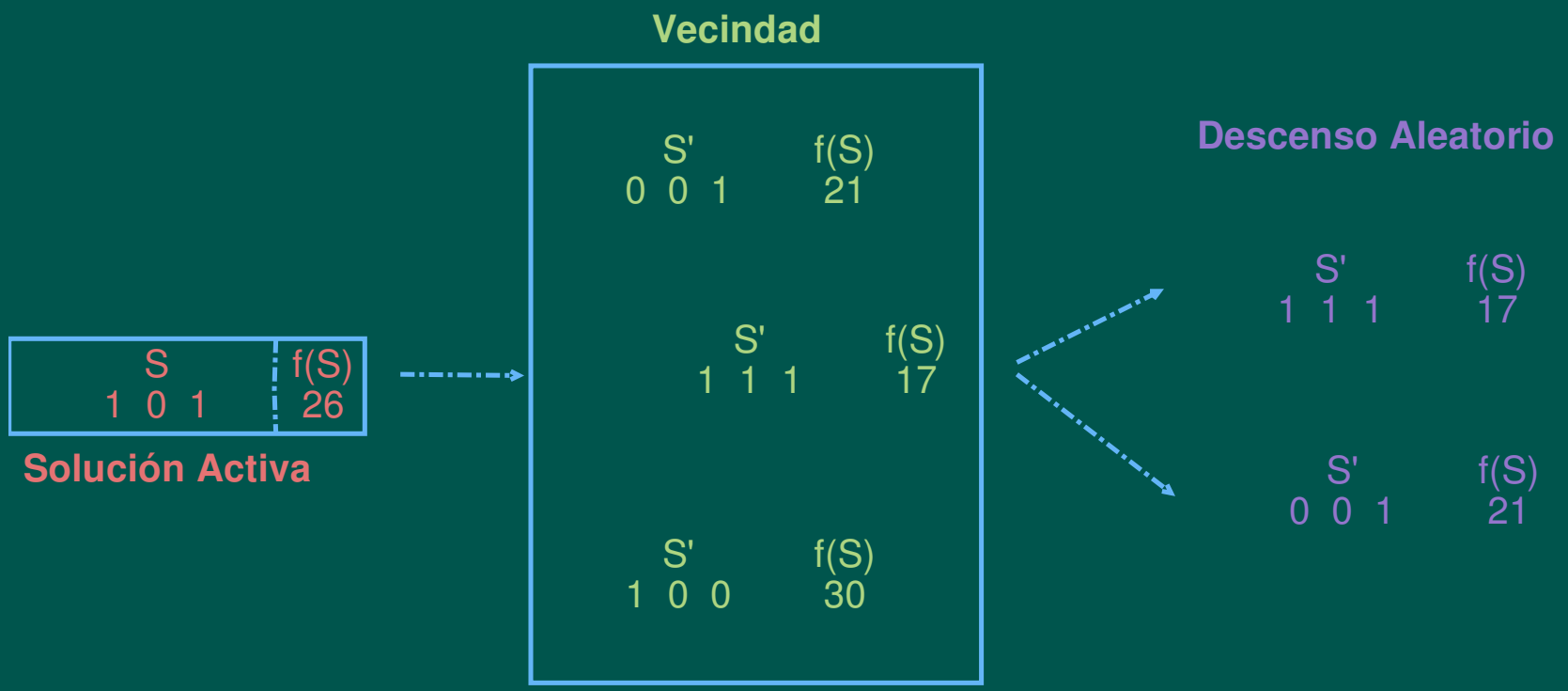
- Mayor descenso: Determinista completamente
- Primero en mejorar: Determinista parcialmente, depende de la forma en que se procesan los vecinos.
- Selección Aleatoria: Completamente estocástico, selecionando al primero que mejore.

Solución Activa


Vecindad
S
f(S)
1 0 1
26
Solución Activa
Orden de generación
S'
f(S)
0 0 1
21
S'
f(S)
1 1 1
17
S'
f(S)
1 0 0
30


Vecindad
S
1 0 1
f(S)
26
Solución Activa
Orden de generación
S'
0 0 1
f(S)
21
S'
1 1 1
f(S)
17
S'
1 0 0
f(S)
30
Primero en mejorar
S'
0 0 1
f(S)
21


Vecindad
S
1 0 1
f(S)
26
Solución Activa
Orden de generación
S'
0 0 1
f(S)
21
S'
1 1 1
f(S)
17
S'
1 0 0
f(S)
30
Mayor descenso
S'
1 1 1
f(S)
17


Vecindad
S
f(S)
1 0 1
26
Solución Activa
S'
f(S)
0 0 1
21
S'
f(S)
1 1 1
17
S'
f(S)
1 0 0
30
Descenso Aleatorio
S'
f(S)
1 1 1
17
S'
f(S)
0 0 1
21

Ventajas y Desventajas

- Fácil de implementar
- A lo más podemos llegar a un óptimo local, no hay garantía de llegar a un óptimo global.
- El resultado depende fuertemente de la solución inicial
- No hay una cota superior del tiempo de ejecución
  - En la práctica se tienen tiempos aceptables
  - Es posible tener casos con complejidad Exponencial

Algoritmo 2: Búsqueda Aleatoria

Entrada: $MAX > 0$, máximo de iteraciones; $f :\longrightarrow \mathbb{R}$, función objetivo

Resultado: s', mejor solución encontrada

```
1 s_mejor = inicializar solución ;
2 t = 1 ;
3 mientras t < MAX hacer
4     s' = inicializar solución ;
5     si f(s') < f(s) entonces
          // Actualizar la mejor solución encontrada
6         s_mejor = s' ;
7 devolver s_mejor
```

Algoritmo 3: Búsqueda por Escalada

Entrada: $s_0 \in S$, solución inicial; $N : S \longrightarrow 2^S$, función de vecindad; $f :\longrightarrow \mathbb{R}$, función objetivo

Resultado: s', mejor solución encontrada

1 $s = s_0$;
2 $t = 1$;
3 **mientras** $\exists s' \in N(s)$ *tal que* $f(s') < f(s)$ **hacer**
4 Seleccionar $s' \in N(s)$ tal que $f(s') < f(s)$;
5 $s = s'$
6 **devolver** s