


Facultad de Ciencias
Licenciatura en
Ciencias de la Computación

Cómputo Evolutivo

Metaheurísticas de Trayectoria

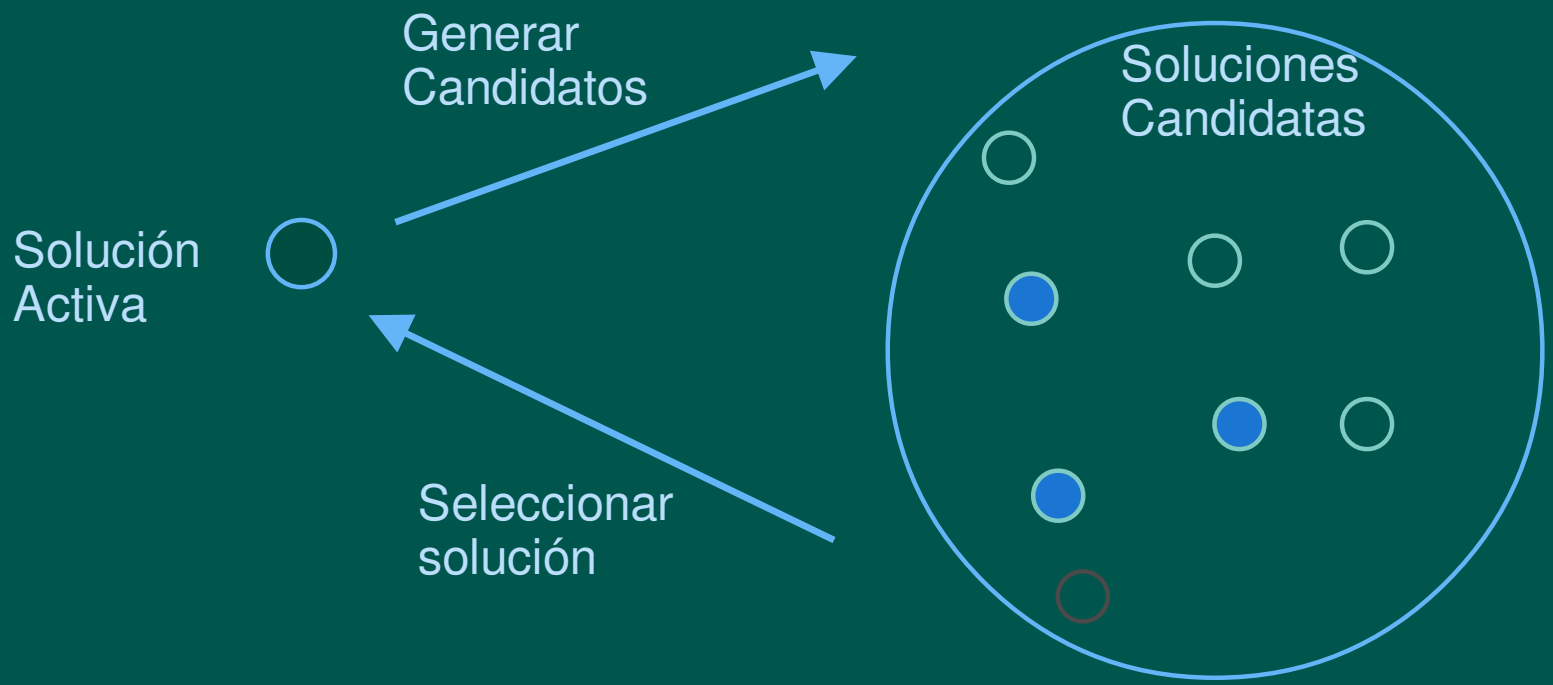
M. en C. Oscar Hernández Constantino
(constantino92@ciencias.unam.mx)

Contenido de la Presentación

Diseño de Metaheurísticas

Esquema general

Metaheurísticas de Trayectoria



Desde una solución inicial s_0 se genera la secuencia $s_1, s_2, \ldots, s_k$

- $s_{i+1} \in N(s_i), \forall i \in [0, k-1]$
- $f(s_{i+1}) < f(s_i), \forall i \in [0, k-1]$

Componentes

Componentes

> Solución Inicial

Generación de Solución Inicial

Tipos de inicialización

- Generación Aleatoria
- Generación mediante heurística
- Soluciones Parciales o Completas

- Entre más grande sea la vecindad, menos impacto tiene la inicialización de la solución

Componentes

> Vecindades

Vecindades

Defínción de Vecindad

Una función de **vecindad** es un mapeo $N : S \rightarrow 2^S$ que asigna a cada solución $s \in S$ un conjunto de soluciones $N(s) \subset S$.

Vecindades en espacios continuos

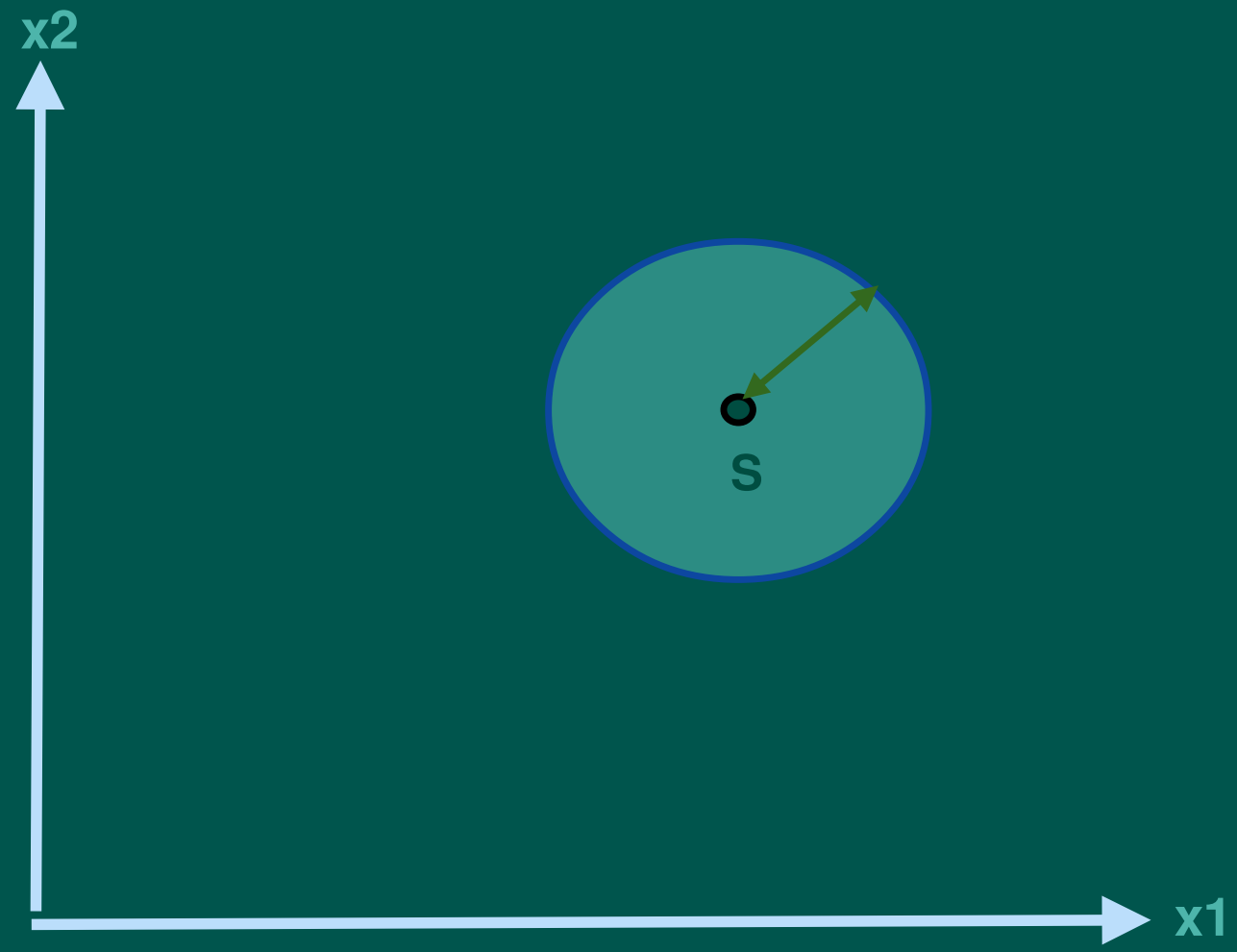
La vecindad $N(s)$ de una solución s en un **espacio continuo** es la bola con centro s y radio igual a $\epsilon > 0$.

$$N(s) = \{ s' \in \mathbb{R} \mid ||s' - s|| < \epsilon \}$$

¿Tamaño de la vecindad?

Depende de la representación utilizada

Ejemplo vecindad en espacio continuo



Vecindades II

Vecindades en espacios discretos

En un problema de optimización discreto, la vecindad $N(s)$ de una solución s es:

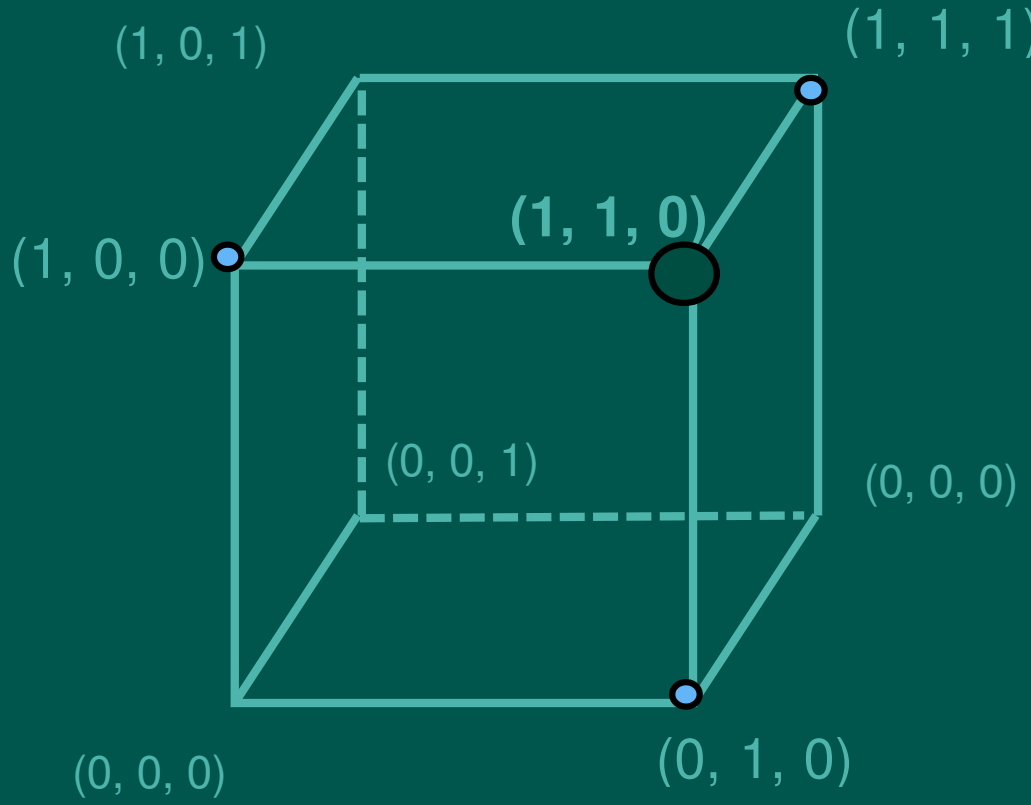
$$N(s) = \{ s' \mid d(s', s) \leq \epsilon \}$$

donde d representa una medida de distancia que está relacionada con el operador de movimiento.

- La vecindad depende fuertemente de la representación utilizada
- Una medida común para representaciones binarias es la distancia de Hamming:

$$d(s, s') = |\{ i \mid s_i \neq s'_i \}|$$

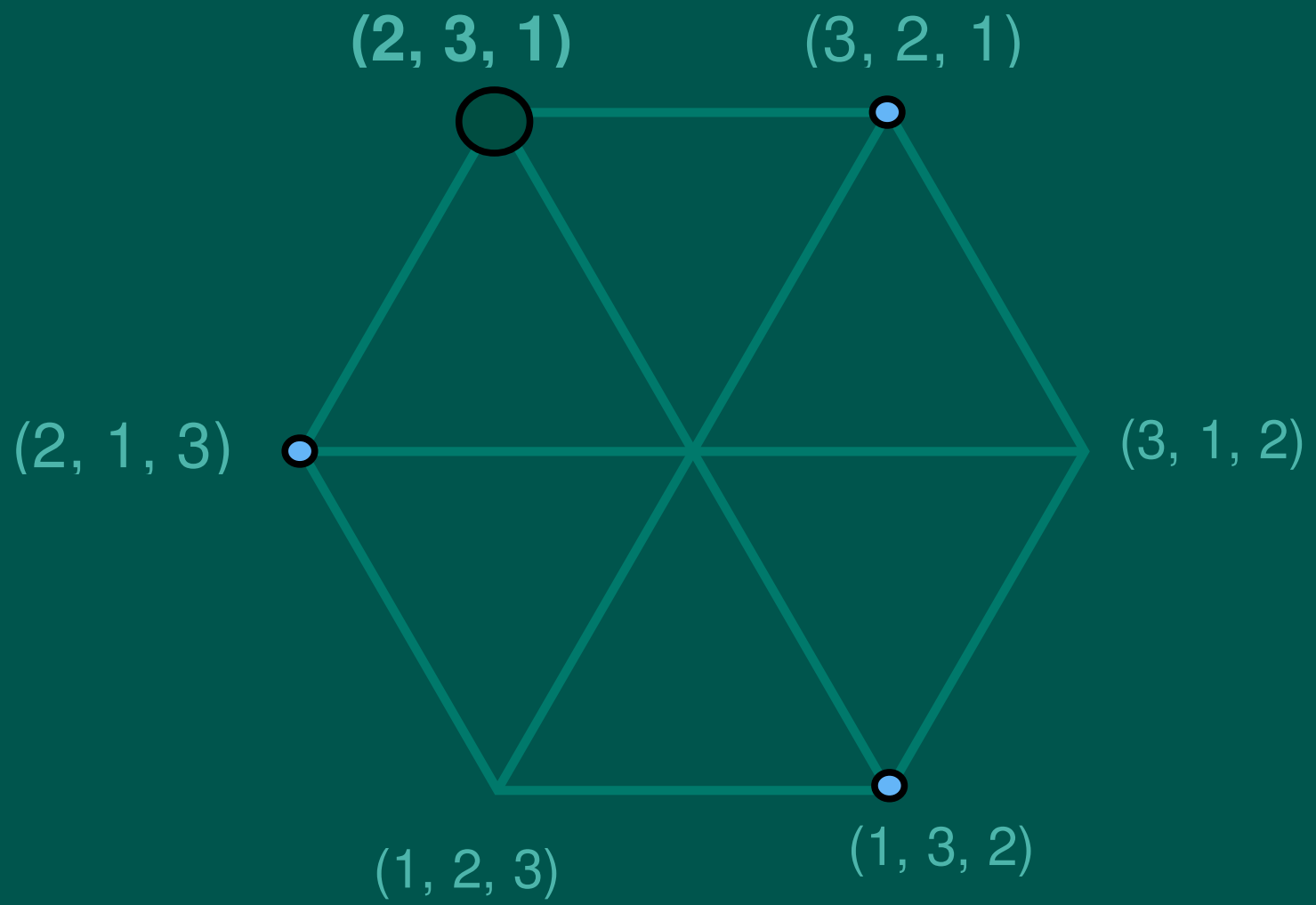
Ejemplo de Vecindad en Representaciones Binarias



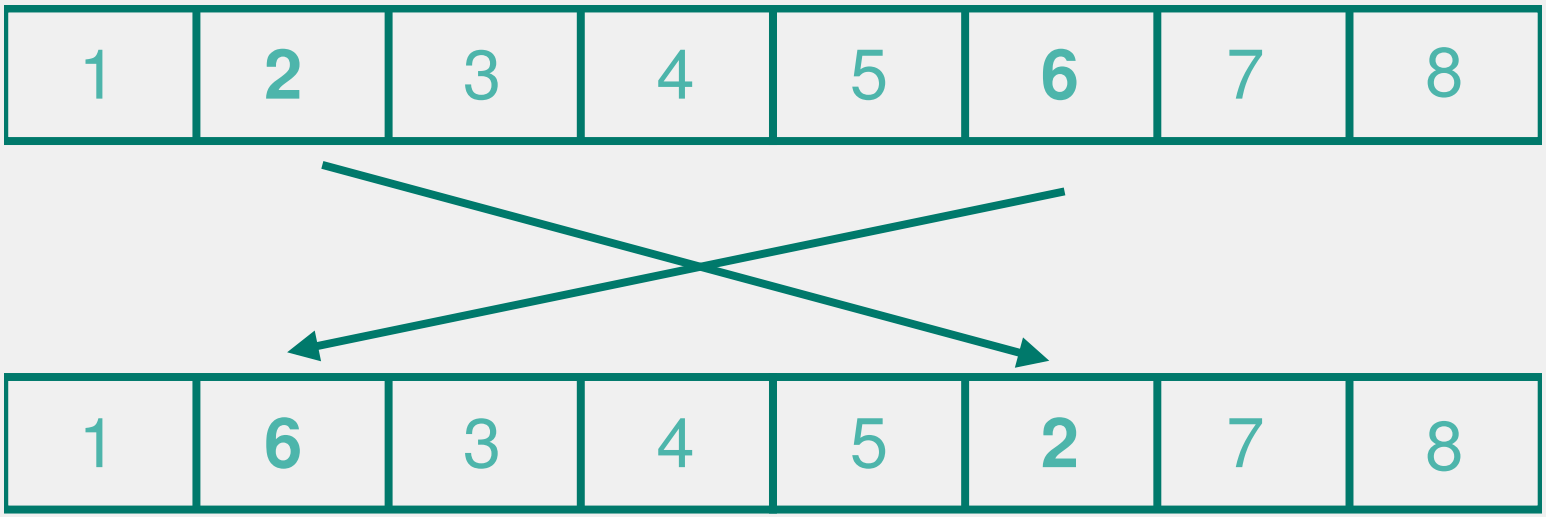
¿Tamaño de la vecindad?

$$|N(s)| = |s| = n$$

Ejemplo de Vecindad en Representaciones con Permutaciones



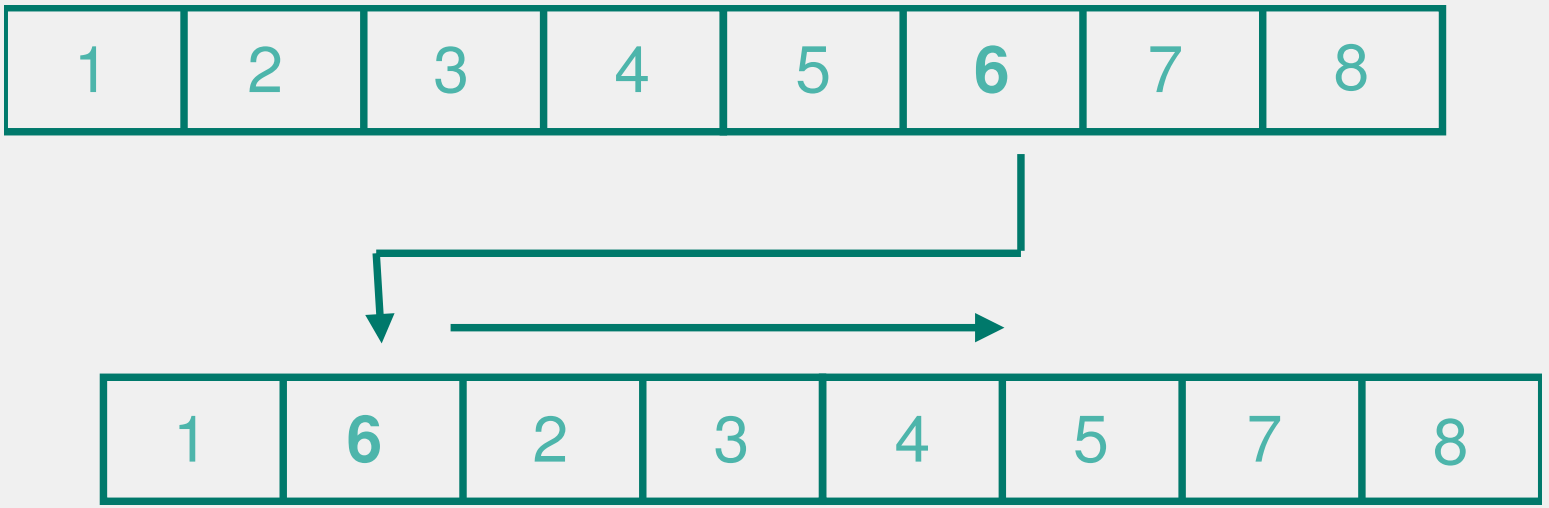
Operador de Intercambio



¿Tamaño de la vecindad?

$$|N(s)| = \binom{|s|}{2} = \frac{n \times (n-1)}{2},\ O(n^2), n = |s|$$

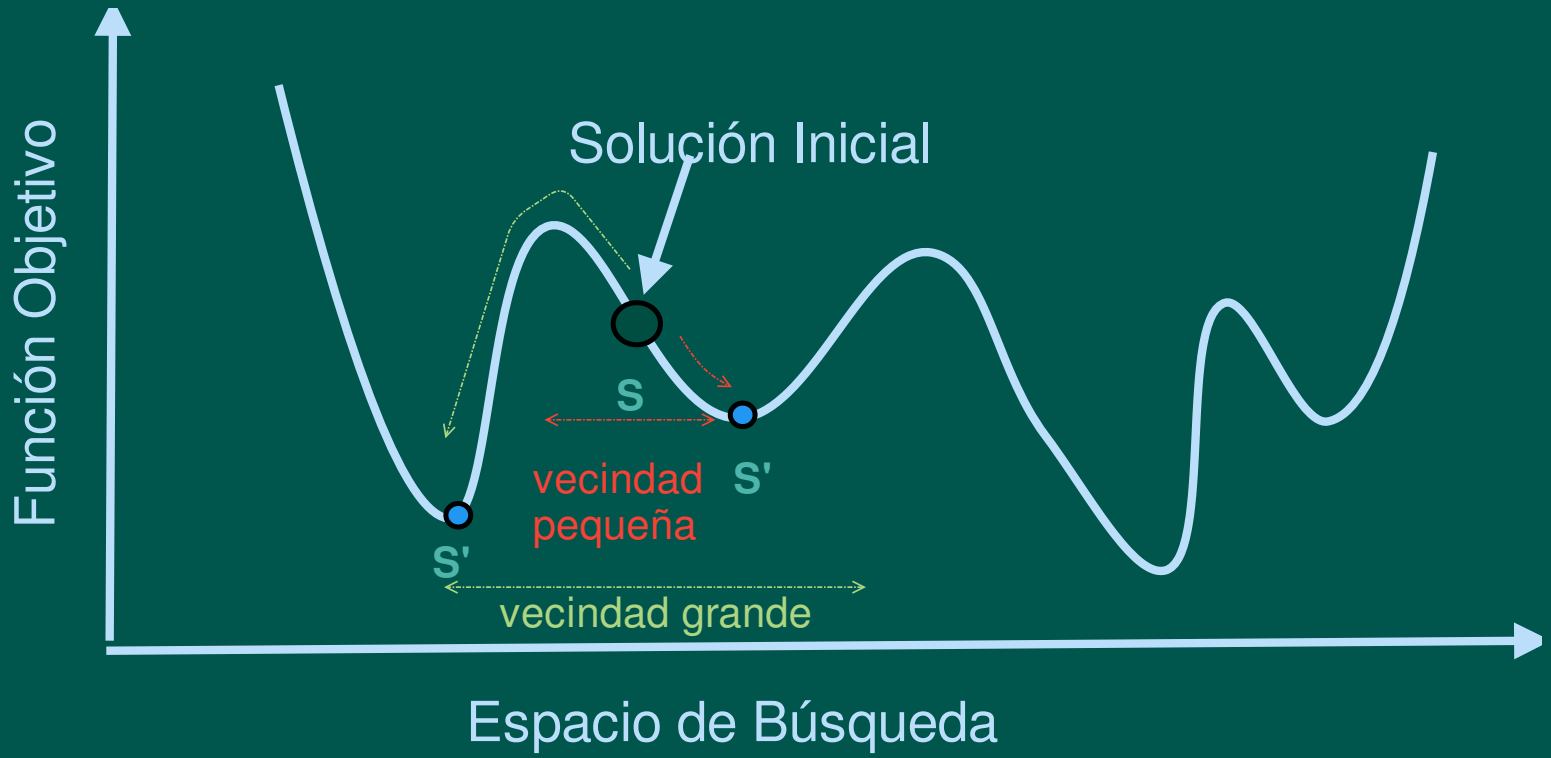
Operador de Inserción



¿Tamaño de la vecindad?

$$|N(s)| = \frac{|s| \times (|s|-1)}{2} = \frac{n(n-1)}{2},\ O(n^2)$$

Vecindades grandes



Diseño de Metaheurísticas

Componentes
> Evaluación

Tipos de Evaluación

- **Evaluación Exacta**
  Se evalua la función objetivo, o bien, se calcula la evaluación del movimiento utilizando alguna propiedad
- **Evaluación por Estimación**
  Se obtiene una estimación que podría o no coincidir con la evaluación real.
- **Evaluación Incremental**
  - Una forma más eficiente de evaluar a las soluciones candidatas es calculando la diferencia que se produce después de aplicar el movimiento m a la solución actual s, $\delta(s, m)$
  - Se puede utilizar en combinación con una representación de soluciones parciales para obtener una evaluación temporal

Componentes

> Criterios de Término

Criterios de Término I

- Llegar al resultado deseado

$$f(s) \leq L$$

  - LB (Lower Bound) : Mejor cota inferior conocida, $f(s) \geq LB$, s mejor solución conocida
  - UB (Upper Bound) : Mejor cota superior conocida, $f(s) \leq LB$, s mejor solución conocida
  - BKS (Best Know Solution) : Mejor solución conocida, $f(s) \leq LB$, s mejor solución conocida
  - OPT : Solución óptima global, $f(s) = OPT$, $s = s^*$, solución óptima global.

Error Absoluto

$|f(s) - f(s^*)| \leq L$

Error Relativo

$\left|\frac{f(s)-f(s^*)}{f(s^*)}\right| \leq L$

Criterios de Término II

- Número de Iteraciones

$$t \leq max\_iteraciones$$

- Máximo número de iteraciones sin mejora

$$Si\ f(s) \geq f(s') \ :\ t_{sin\_mejora} + +$$

$$t_{sin\_mejora} \leq max\_iteraciones\_sin\_mejora$$

Es el único criterio que podría garantizar llegar a un óptimo local

Criterios de Término III

- Número de evaluaciones de función
  Con cada llamada a $f(s)$: $num\_evaluaciones++$

$$num\_evaluaciones \leq max\_evaluaciones$$

- Tiempo total de ejecución Al inicio de la ejecución : $t_{ini} = now()$
  Al inicio o al final de cada iteración : $t_{act} = now()$

$$(t_{act} - t_{ini}) < max\_tiempo$$

¿ Preguntas ?