Metaheurísticas

Seminario 3. Problemas de optimización con técnicas basadas en poblaciones

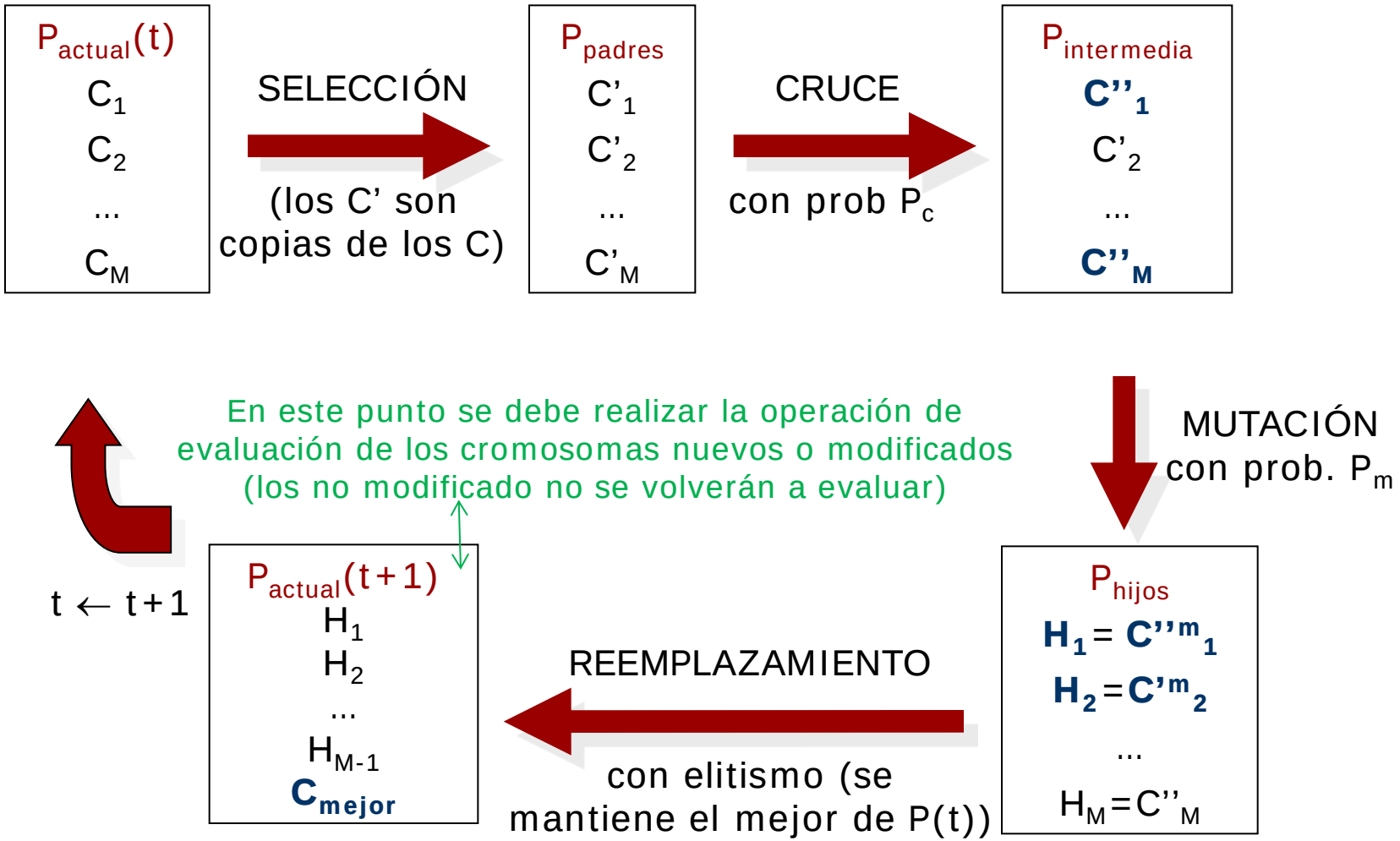
1. Estructura de un Algoritmo Genético y Aspectos de Implementación
2. Problemas de Optimización con Algoritmos Genéticos
   - SCP

Estructura de un Algoritmo Genético

Procedimiento Algoritmo Genético

```
Inicio (1)
    t = 0;
    inicializar P(t);
    evaluar P(t);
    Mientras (no se cumpla la condición de parada) hacer
    Inicio(2)
        t = t + 1
        seleccionar P' desde P(t-1)
        recombinar P'
        mutar P'
        reemplazar P(t) a partir de P(t-1) y P'
        evaluar P(t)
    Final(2)
Final(1)
```

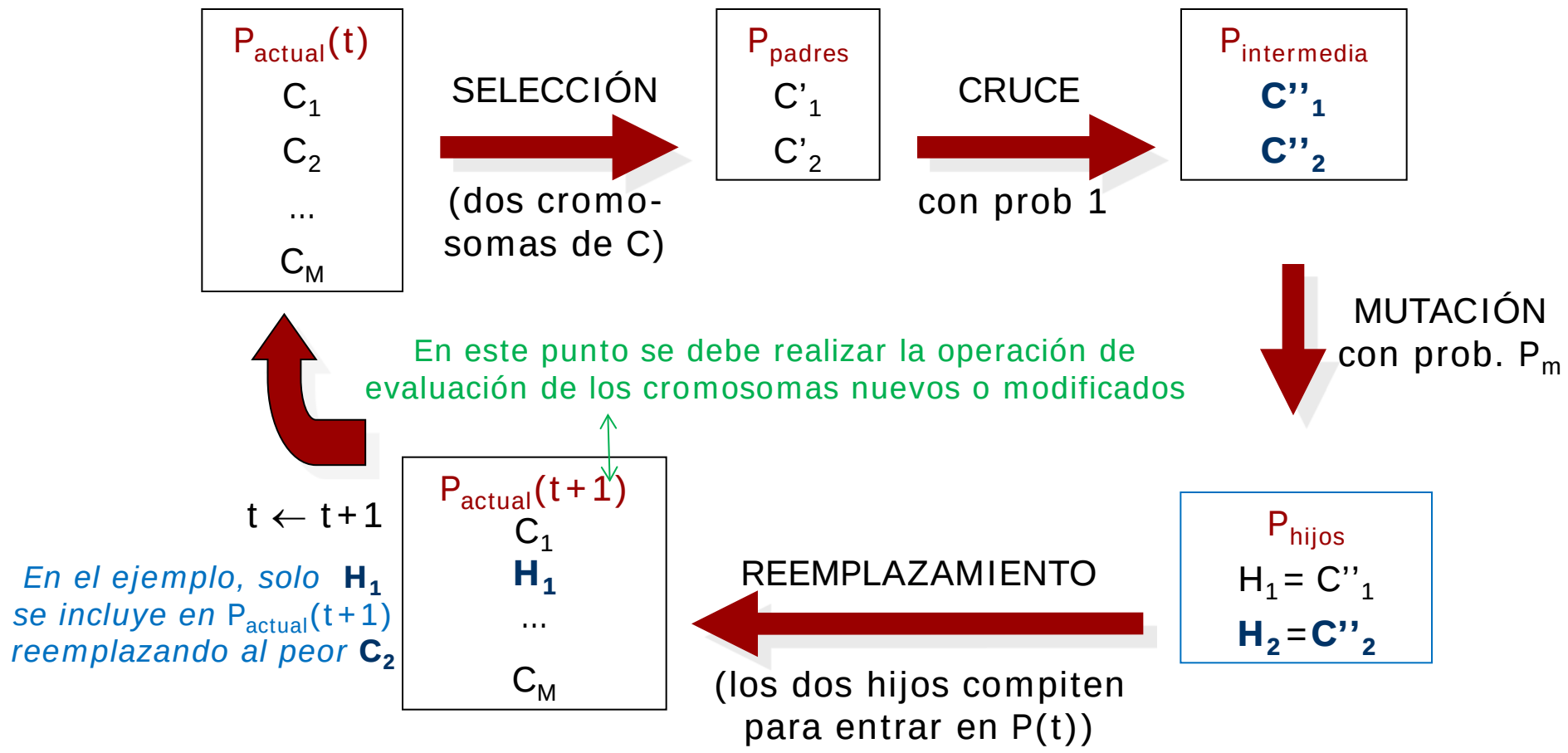
Modelo Generacional:



Elitismo: si la mejor solución de la generación anterior no sobrevive, sustituye directamente la peor solución de la nueva población

Los hijos generados en el cruce reemplazan a los padres, aunque no sean mejor que ellos

Modelo Estacionario:



Los dos descendientes (H_1, H_2) generados tras el cruce y la mutación sustituyen a los dos peores de la población actual $P_{actual}(t)$, en caso de ser mejores que ellos. La nueva población $P_{actual}(t+1)$, está compuesta por los individuos de la población anterior y los nuevos hijos que sustituyen a los peores (en caso de ser mejores que los peores).

Modelo Generacional: Aspectos de Implementación

- Lo más costoso en tiempo de ejecución de un Algoritmo Genético es la generación de números aleatorios para:
  - Aplicar el mecanismo de selección
  - Emparejar las parejas de padres para el cruce
  - Decidir si una pareja de padres cruza o no de acuerdo a P_c
  - **Decidir si cada gen muta o no de acuerdo a P_m**

- Se pueden diseñar implementaciones eficientes que reduzcan en gran medida la cantidad de números aleatorios necesaria:
  - Emparejar las parejas para el cruce: Como el mecanismo de selección ya tiene una componente aleatoria, se aplica siempre un emparejamiento fijo: el primero con el segundo, el tercero con el cuarto, etc.

Modelo Generacional: Aspectos de Implementación

- ✔ Decidir si una pareja de padres cruza: En vez de generar un aleatorio *u* en [0,1] para cada pareja y cruzarla si $u \leq P_c$, se estima a priori (al principio del algoritmo) el número de cruces a hacer en cada generación (esperanza matemática):

$$N^o \text{ esperado cruces} = P_c \cdot M/2$$

- ✔ Por ejemplo, con una población de 50 cromosomas (25 parejas) y una P_c de 0.7, cruzarán 0,7*25= 17,5 (18 parejas). Para el SCP se utilizara esta esperanza matemática en el AGG con el cruce HUX, no se usara en el cruce de tipo fusión debido a que genera un solo hijo

- ✔ De nuevo, consideramos la aleatoriedad que ya aplica el mecanismo de selección y cruzamos siempre las *Nº esperado cruces* primeras parejas de la población intermedia

Modelo Generacional: Aspectos de Implementación

- ✔ Decidir si cada gen muta: El problema es similar al del cruce, pero mucho más acusado
- ✔ Normalmente, tanto el tamaño de población M como el de los cromosomas n es grande. Por tanto, el número de genes de la población, M·n, es muy grande
- ✔ La P_m, definida a nivel de gen, suele ser muy baja (p.e. P_m=0.01). Eso provoca que se generen muchos números aleatorios para finalmente realizar muy pocas mutaciones
- ✔ Por ejemplo, con una población de 60 cromosomas de 100 genes cada uno tenemos 6000 genes de los cuales mutarían unos 60 (*Nº esperado mutaciones* = P_m · *nº genes población*, esperanza matemática)
- ✔ Generar 6000 números aleatorios en cada generación para hacer sólo 60 mutaciones (en media) es un gasto inútil. Para evitarlo, haremos siempre exactamente *Nº esperado mutaciones* en cada generación
- ✔ *En el modelo generacional el nº esperado mutaciones es 1 (50*0,02)*

Modelo Generacional: Aspectos de Implementación

- ✔ Aparte de hacer un número fijo de mutaciones, hay que decidir cuáles son los genes que mutan

- ✔ Normalmente, eso se hace también generando números aleatorios, en concreto dos, un entero en {1, …, M} para escoger el cromosoma y otro en {1, …, n} para el gen

- ✔ Existen también mecanismos más avanzados que permiten escoger el gen que muta generando un único número real en [0,1] y haciendo unas operaciones matemáticas

- ✔ En nuestro caso, se obtiene un número aleatorio entre el (1 y el nº total de individuos) y ese es el individuo que se muta (probabilidad mutación del cromosoma.

El Problema del SCP

- El **problema del Set Covering Problem (SCP)** consiste en encontrar el número mínimo de conjuntos que contengan todos los elementos de todos los conjuntos dados. Existen muchas aplicaciones de este tipo de problemas, siendo las principales la localización de servicios, balanceo de líneas de producción, entre otros.

- Ejemplo: Se tiene una ciudad dividida en 20 sectores y 10 lugares en donde pueden ser instalados ciertos servicios, como por ejemplo Cuarteles de Bomberos, Hospitales etc.. Cada lugar puede dar servicio a los sectores adyacentes. Se pide determinar donde ubicar los servicios para minimizar los costos, pero asegurándose que se cubran todos los sectores

El Problema del SCP

Definición de SCP

- A= a_{ij} matriz binaria. Ejemplo: regiones cubiertas por cada lugar
- M = {1…m} objetos. Son las filas de la matriz A_{ij}. Ejemplo: 20 regiones
- N = {1…n} subconjuntos. Son las columnas de la matriz A_{ij}. Ejemplo: 10 lugares.
- C = coste de cada objeto. Ejemplo: coste de los lugares
- X = vector binario que indica que subconjuntos forman la solución, es decir, cubren todos los objetos.
- Restricción todos los objetos deben estar cubiertos por al menos un subconjunto. Ejemplo: todas las regiones deben estar cubiertas por un lugar.

El Problema del SCP

Definición de SCP

- Función objetivo:

$$\text{mín}(z) = \sum_{j=1}^{n} c_j x_j$$

- Restricciones:

Sujeto a:

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 \qquad \forall i \in M$$

$$x_j = \begin{cases} 1 & \text{si } j \in S \\ 0 & \text{si no} \end{cases} \qquad \forall j \in N$$

- Representación de una solución (vector X): Binaria

$$[1000100010]$$

Algoritmo Genético para SCP

- ✓ J.E. Beasley, RC. Chu. A genetic algorithm for the set covering, European Journal of Operational Research 94 (1996) 392-404
- ✓ Pablo Itaim Ananias, Resolución del Problema de Set-Covering utilizando un Algoritmo Genético.

- Codificación individuos: representación binaria.

- Generación de la población inicial: Todos los cromosomas se generarán de forma aleatoria usando un método simple que genera soluciones factibles y sin columnas redundantes.
  - El método simple consiste en ir seleccionando de forma aleatoria columnas (que cubran objetos no cubiertos previamente), para de esta forma completar el individuo.
  - Posteriormente se eliminan las columnas redundantes.

Algoritmo Genético para SCP

- Tamaño de la población: 50
- Modelos de evolución: 2 variantes: generacional con elitismo / estacionario con 2 hijos que compiten con los dos peores de la población
- Mecanismo de selección: torneo binario (Se seleccionan aleatoriamente dos individuos y se comparan sus costes; aquél con menor coste pasa a la siguiente generación). En AGG con cruce HUX se seleccionaran 18 parejas (pág 6)
- Operadores de cruces: El alumno implementara y obtendrá resultados con dos operadores de cruces: Operador de fusión propuesto por Beasley y operador de cruce uniforme (HUX). En el caso del AGE solo se aplicara el cruce HUX.

Algoritmo Genético para SCP

- Probabilidad de cruce: Será 0.7 en el AGG y 1 en el AGE (siempre se cruzan los dos padres).

- Operador de Cruce1 (fusión de Beasley): Consiste en seleccionar 2 padres y generar un solo hijo. Su funcionamiento es: dadas las soluciones P1 y P2, con sus respectivos valores de fitness f_{P1} y f_{P2} y el individuo hijo C, se tiene $\forall i=1,\ldots,n$:

(i) if $P_1[i] = P_2[i]$, then $C[i] := P_1[i] = P_2[i]$,
(ii) if $P_1[i] \neq P_2[i]$, then
  (a) $C[i] := P_1[i]$ with probability $p = f_{P_2}/(f_{P_1} + f_{P_2})$,
  (b) $C[i] := P_2[i]$ with probability $1 - p$.

Considera los valores de fitness para determinar que elementos se copian en el hijo, logrando pasar información de mayor valor a los descendientes.

- ✓ Solo se aplica en el AGG

Algoritmo Genético para SCP

- Operador de Cruce2 (uniforme HUX): Intercambia exactamente la mitad de los alelos que son distintos en los padres

  Padres: 11001100 y 00000000 Hijos generados: 10 00 01 00 y 01 00 10 00

  - En AGG con el cruce HUX se seleccionaran 18 parejas (pág 6)

- Operador de mutación: consiste en seleccionar al azar un individuo (teniendo en cuenta la probabilidad de mutación), y cambiar ***k*** genes: si hay un 0 lo cambia por un 1 y si hay un 1 lo cambia por un 0. Donde *k* esta determinado por la probabilidad de mutación por gen.
  - Probabilidad de mutación por cromosoma: *será de 0.02. En el modelo generacional el nº esperado mutaciones es 1 (50*0,02). Es decir se selecciona un individuo y ese es el que se muta.*
  - *Probabilidad de mutación por gen: Será variable comenzando en 0,2 y cada 1000 evaluaciones se decrementa en 0,01.*

Algoritmo Genético para SCP

- Reparación: Antes de evaluar la función objetivo es necesario realizar una reparación del cromosoma para que la solución sea valida. En este punto es donde se incrementa el número de evaluaciones después de reparar el cromosoma.
  - Aplicando el greedy explicado para la práctica 1 (BL)
  - Posteriormente, se elimina columnas redundantes

- Número de evaluaciones: 20000

- Reinicialización: Se aplicara una reinicialización para evitar un estancamiento de la búsqueda. Será dinámico y se podrán hacer 0, 1 o muchas veces, dependerá de la evolución del algoritmo.

Algoritmo Genético para SCP

- En concreto se aplicara una reinicialización cuando se cumpla una de las siguientes condiciones:
  - Después de 20 generaciones no se haya conseguido mejorar la mejor solución encontrada hasta el momento, para el AGG. No se considera en el AGE.
  - O en el caso de que la población converja hacia una misma solución, es decir, que el 80% de la población contengan el mismo individuo. Para ambos algoritmos AGG y AGE.

- La forma de realizar la reinicialización es incluir la mejor solución y el resto de soluciones se generaran aleatoriamente siguiendo el mismo proceso usado para generar la población inicial.