Metaheurística

1. **Busqueda local.**

   El término "local" se utiliza frecuentemente en los estudios teóricos y prácticos de las metaheurísticas de búsqueda. Se asocia al uso de estructuras de entorno, reflejando el concepto de proximidad o vecindad entre las soluciones alternativas del problema.

   Todas las soluciones incluidas en el entorno de la solución actual, que viene delimitado por un operador de generación de soluciones, se denominan soluciones vecinas.

   Los algoritmos basados en esta estrategia efectúan un estudio local del espacio de búsqueda, puesto que analizan el entorno de la solución actual para decidir cómo continuar el recorrido de la búsqueda.

   Una búsqueda local es un proceso que, dada la solución actual en la que se encuentra el recorrido, selecciona iterativamente una solución de su entorno para continuar la búsqueda. Basta con diseñar la estructura de entorno para obtener un modelo genérico de algoritmo de búsqueda.

   - Descripción:
     - Se fija una codificación para las soluciones.
     - Se define un operador de generación de vecino y, en consecuencia, se fija una estructura de entorno para las mismas
     - Se escoge una solución del entorno de la solución actual hasta quese satisfaga el criterio de parada.
   - Elementos en el proceso de búsqueda:
     - Proceso de elección de la solución inicial.
     - Operador de vecino: Proceso de selección de solución/generación de una solución vecina: S $\rightarrow$ S' si S' pertenece al E(S).
     - Proceso de aceptación de solución vecina como solución actual.

   <u>Busqueda basada en Poblaciones:</u>

   En un modelo de poblaciones de soluciones debemos definir cómo generar nuevas poblaciones. Las nueva población debería estar asociada a la anterior y tener en cuenta la calidad de las anteriores soluciones.

   Se podría tener un modelo que seleccione soluciones, opere con ellas y puedan ser reinsertadas en la población dando lugar a una nueva población. Podemos imitar a la evolución de especies y la genética, cómo se combinan cromosomas (algoritmos genéticos)

   <u>Busqueda Aleatoria Pura:</u>

   Se elige aleatoriamente una muestra de soluciones del espacio de búsqueda y se devuelve la mejor. Se diría que el entorno de una solución es todo el espacio de búsqueda. Si el problema tiene m soluciones y el óptimo es único, la probabilidad de que al generar aleatoriamente una solución se obtenga la óptima es 1/m.

Busqueda aleatoria Por Recorrido al Azar:

La solución inicial se genera aleatoriamente. El entorno de cualquier solución es propio (no consta de todas las soluciones del espacio de búsqueda). La solución vecina a la solución actual se escoge aleatoriamente dentro del entorno y se acepta automáticamente. Se almacena la mejor solución obtenida hasta el momento. Ésta es la solución que se devuelve finalmente.

En definitiva, puede considerarse como una búsqueda local en la que se acepta el primer vecino generado, independientemente de que sea mejor o peor que la solución actual.

Busqueda Local del Mejor (Ascensión de colinas, mejor vecino):

Genera el entorno completo de la solución actual y selecciona la mejor solución vecina. Si ésta es mejor que la solución actual, la sustituye y se continúa la iteración. En otro caso, el algoritmo finaliza.

Busqueda Local del Primero Mejor (Acensión simple de colinas, primero mejor):

Se va generando paso a paso el entorno de la solución actual hasta que se obtiene una solución vecina que mejora a la actual o se construye el entorno completo. En el primer caso, la solución vecina sustituye a la actual y se continúa iterando. En el segundo, se finaliza la ejecución del algoritmo.

Métodos Búsqueda Local Básicos:

- Esquema de representación: Permutación de {1, ..., n}
- Función objetivo: $MinC(S)=\sum_{i=1}^{n-1}(D[S[i],S[i+1]])+D[S[n],S[1]]$
- Mecanismo de generación solución inicial: Permutación aleatoria.
- Operador de generación de nuevas soluciones: escoger dos posiciones e intercambiar sus valores.
- Mecanismo de selección: Selección del mejor o el primer mejor.
- Criterio de parada: Cuando el vecino generado no mejore a la solución actual.

Problemas de la Busqueda local:

- Se suele caer en óptimos locales, que a veces están bastante alejados del óptimo global del problema.

Soluciones al problema:

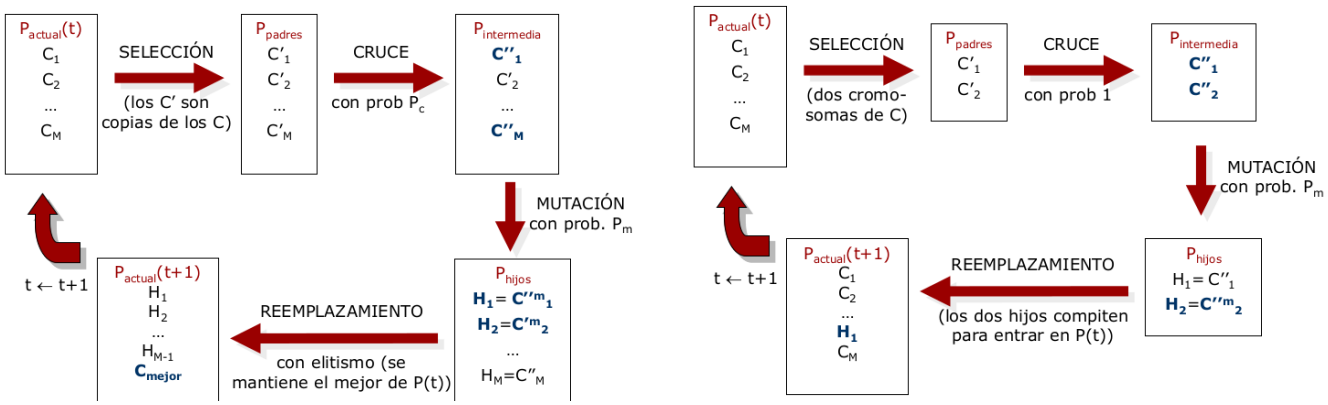
- Permitir movimientos de empeoramiento de la solución actual: Enfriamiento Simulado, Búsqueda Tabú.
- Modificar la estructura de entornos: Búsqueda Tabú, Búsqueda en Entornos Variables: VNS.
- Volver a comenzar la búsqueda desde otra solución inicial: Búsquedas Multiarranque, ILS, VNS.

2. **Algoritmos genéticos.**

Son algoritmos de optimización búsqueda y aprendizaje inspirados en los procesos de **Evolución Natural** y **Evolución Genética.** Los ingrediestes son: Una población T, en la que se produce una reproducción y selección t+1 en la que se ha aplicado un cruce y obtenido una mutación.

Podemos encontrar dos modelos:

- **Modelo generacional**. Durante cada iteración se crea una población completa con nuevos individuos. La nueva población reemplaza directamente a la antigua.
- **Modelo estacionario**. Durante cada iteración se escogen dos padres de la población (diferentes mecanismos de muestreo) y se les aplican los operadores genéticos. El/los descendiente/s reemplazan a uno/dos cromosoma/s de la población anterior.



El modelo estacionario es elitista. Además produce una presión selectiva alta (convergencia rápida) cuando se reemplazan los peores cromosomas de la población.

Representacion:

Debemos disponer de un mecanismo para codificar un individuo como un genotipo. Existen muchas maneras de hacer esto y se ha de elegir la más relevante para el problema en cuestión. Una vez elegida una representación, hemos de tener en mente como los genotipos (codificación) serán evaluados y qué operadores genéticos hay que utilizar.

- Representación binaria: Inicialización 1...0 prob. 0.5
- Representación real: Inicialización uniforme sobre un intervalo.
- Representación de orden: Represntan permutaciones, problemas de secuenciación de acciones.

Estrategia selección:

Debemos de garantizar que los mejores individuos tienen una mayor posibilidad de ser padres (reproducirse) frente a los individuos menos buenos. Debemos de ser cuidadosos para dar una oportunidad de reproducirse a los individuos menos buenos. Éstos pueden incluir material genético útil en el proceso de reproducción. Esta idea nos define la presión selectiva que determina en qué grado la reproducción está dirigida por los mejores individuos. Estrategias de selección:

- **Selección por torneo**: Escoger aleatoriamente k individuos, con reemplazamiento, y seleccionar el mejor de ellos. k se denomina tamaño del torneo. A mayor k, mayor presión selectiva y viceversa.
- **Orden Lineal**: La población se ordena en función de su fitness y se asocia una probabilidad de selección a cada individuo que depende de su orden.
- **Selección por Ruleta**: Se asigna una probabilidad de selección proporcional al valor del fitness del cromosoma
- **Selección Aleatoria**.

Operador de cruze:

Podríamos tener uno o más operadores de cruce para nuestra representación. Algunos aspectos importantes a tener en cuenta son:

- Los hijos deberían heredar algunas características de cada padre. Si éste no es el caso, entonces estamos ante un operador de mutación.
- Se debe diseñar de acuerdo a la representación.
- La recombinación debe producir cromosomas válidos.
- Se utiliza con una probabilidad alta de actuación sobre cada pareja de padres a cruzar (Pc entre 0.6 y 0.9), si no actua los padres son los descendientes del proceso de recombinación de la pareja.

Existe muchos operadores específicos para la codificación real, como por ejemplo

- **BLX-alpha**: que dados cromosomas genera dos descendientes donde cada caracteristica se genera aleatoriamente en el intervalo: $[C_{min} - I \cdot alpha\ C_{max} + I \cdot alpha]$ donde $I = C_{max} - C_{min}$
- **OX:** los datos representan orden por lo que seleecionamos una parte del padre1 y lo ponemos en el resultado el resto sería el resultado de ordenar ambos padre1 y padre2 donde de padre2 se quitan los genes menores y mayores de los seleccionados en padre1.

Operador mutación:

Podemos tener uno o más operadores de mutación para nuestra representación. Algunos aspectos importantes a tener en cuenta son:

- Debe permitir alcanzar cualquier parte del espacio de búsqueda.
- El tamaño de la mutación debe ser controlado.
- Debe producir cromosomas válidos.
- Se aplica con una probabilidad muy baja de actuación sobre cada descendiente obtenido tras aplicar el operador de cruce (incluidos los descendientes que coinciden con los padres porque el operador de cruce no actúa).

La mutación ocurre con una probabilidad p_m para cada gen. Para una representación real la perturbación de los valores mediante un valor aleatorio. Frecuentemente, mediante una distribución Gaussiana/normal N(omega,phi), donde, omega es la media y phi es la desviación típica para cada parámetro. Para una presentación en orden sería coger dos genes y hacer una permutación.

Reemplazamiento:

La presión selectiva se ve también afectada por la forma en que los cromosomas de la población son reemplazados por los nuevos descendientes. Podemos utilizar métodos de reemplazamiento aleatorios, o determinísticos. Podemos decidir no reemplazar al mejor cromosoma de la población: Elitismo (el uso del Elitismo es aconsejado en los modelos generacionales para no perder la mejor solución encontrada). Un modelo con alto grado de elitismo consiste en utilizar una población intermedia con todos los padres (N) y todos los descendientes y seleccionar los N mejores. Esto se combina con otras componentes con alto grado de diversidad.

- Reemplazos estacionarios:
  - Reemplazar al peor de la población: Genera alta presión selectiva.
  - Torneo Restringido (RTS): Se reemplaza al mas (RW)parecido de entre w (w=3, ...). Mantiene una cierta diversidad.
  - Peor entre semejantes (WAMS): Se reemplaza el peor cromosoma del conjunto de los w (w=3, ...) padres más parecidos al descendiente generado (seleccionados de toda la población). Busca equilibrio entre diversidad y presión selectiva.
  - Algoritmo de Crowding Determinístico (DC): El hijo reemplaza a su padre más parecido. Mantiene diversidad.

Sobre su utilización:

Nunca se deben sacar conclusiones de una única ejecución utilizar medidas estadísticas (medias, medianas, ...) con un número suficiente de ejecuciones independientes

No se debe ajustar/chequear la actuación de un algoritmo sobre ejemplos simples si se desea trabajar con casos reales.

Existe una comentario genérico en el uso de los Algoritmos no determinísticos: “Se puede obtener lo que se desea en una experimentación de acuerdo a la dificultad de los casos utilizados” (se encuentran propuestas en las que basta encontrar un caso adecuado para un algoritmo para afirmar que es muy bueno, pero esta afirmación no puede ser extensible a otros casos, es el error en el que incurren algunos autores)

3. Algoritmos basados en Trayectorias.

Problemas de la Búsqueda Local: Suele caer en óptimos locales, que a veces están bastante alejados del óptimo global del problema. Existen tres soluciones ya vista y vamos a describirlas:

Enfriamiento Simulado:

El Enfriamiento o Recocido Simulado es un algoritmo de búsqueda por entornos con un criterio probabilístico de aceptación de soluciones basado en Termodinámica. Un modo de evitar que la búsqueda local finalice en óptimos locales, hecho que suele ocurrir con los algoritmos tradicionales de búsqueda local, es permitir que algunos movimientos sean hacia soluciones peores. Pero si la búsqueda está avanzando realmente hacia una buena solución, estos movimientos “de escape de óptimos locales” deben realizarse de un modo controlado.

En el caso del Enfriamiento Simulado (ES), esto se realiza controlando la frecuencia de los movimientos de escape mediante una función de probabilidad que hará disminuir la probabilidad de estos movimientos hacia soluciones peores conforme avanza la búsqueda (y por tanto estamos más cerca, previsiblemente, del óptimo local). Se aplica la filosofía habitual de búsqueda de diversificar al principio e intensificar al final

Algoritmo de Metrópolis: El fundamento de este control se basa en el trabajo de Metrópolis (1953) en el campo de la termodinámica estadística Básicamente, Metrópolis modeló el proceso de enfriamiento simulando los cambios energéticos en un sistema de partículas conforme decrece la temperatura, hasta que converge a un estado estable (congelado). Las leyes de la termodinámica dicen que a una temperatura t la probabilidad de un incremento energético de magnitud δE se puede aproximar por: P[δE] = exp(-δE/kt) siendo k una constante física denominada Boltzmann.

En el modelo de Metrópolis, se genera una perturbación aleatoria en el sistema y se calculan los cambios de energía resultantes: si hay una caída energética, el cambio se acepta automáticamente; por el contrario, si se produce un incremento energético, el cambio será aceptado con una probabilidad indicada por la anterior expresión

El algoritmo de Enfriamiento Simulado es un método de búsqueda por entornos caracterizado por un criterio de aceptación de soluciones vecinas que se adapta a lo largo de su ejecución Hace uso de una variable llamada Temperatura, T, cuyo valor determina en qué medida pueden ser aceptadas soluciones vecinas peores que la actual. La variable Temperatura se inicializa a un valor alto, denominado Temperatura inicial, T0, y se va reduciendo cada iteración mediante un mecanismo de enfriamiento de la temperatura, alpha(·), hasta alcanzar una Temperatura final, Tf. En cada iteración se genera un número concreto de vecinos, L(T), que puede ser fijo para toda la ejecución o depender de la iteración concreta Cada vez que se genera un vecino, se aplica el criterio de aceptación para ver si sustituye a la solución actual Si la solución vecina es mejor que la actual, se acepta automáticamente, tal como se haría en la búsqueda local clásica En cambio, si es peor, aún existe la probabilidad de que el vecino sustituya a la solución actual. Esto permite al algoritmo salir de óptimos locales, en los que la BL clásica quedaría atrapada. Esta probabilidad depende de la diferencia de costes entre lasolución actual y la vecina, δ, y de la temperatura T. A mayor temperatura, mayor probabilidad de aceptación de soluciones peores. Así, el algoritmo acepta soluciones mucho peores que la actual al principio de la ejecución (exploración) pero no al final (explotación). A menor diferencia de costes, mayor probabilidad de aceptación de soluciones peores. Una vez finalizada la iteración, es decir, tras generar L(T) soluciones vecinas, se enfría la temperatura y se pasa a la siguiente iteración

- Representación: Permutaciones, binario y vector de numero reales.
- Solución Inicial: Uso de técnicas eficientes o de conocimiento experto. Un ejemplo es el uno de algoritmo greedy. No parece conveniente considerar valores fijos independientes del problema. Propuesta: usar la emperatura.
    - Valor inicial para la temperadura tiene que ser dependiente del problema $T=(mu / \ln(phi)) \cdot C(S_0)$.
    - Mecanismos de enfriamiento: Basado en temperaturas descentender fijadas, decenso constante de la tempetatura, geometrico y dos más complejos, Bolzman y cauchy q se puede modificar para M iteracciones.
- Condición de patada, cuando el sistema se enfria T=0, o cuando Tf está por debajo

de un valor fijo o despues de un número de iteracciones. Otro citerio sería parar cuando la generación de vecinos no existe ningún beneficio.

Búsqueda Tabú: es un procedimiento de búsqueda por entornos cuya característica distintiva es el uso de memoria adaptativa y estrategias especiales de resolución de problemas. La memoria adaptativa permite:

- Restringir el entorno de búsqueda.
- Introducir mecanismos de reinicialización de la búsqueda mediante intensificación sobre zonas del espacio de búsqueda ya visitadas, o diversificación sobre posibles zonas del espacio de búsqueda poco visitadas

La Búsqueda Tabú (TS) es una técnica de búsqueda por entornos caracterizada por dos aspectos principales:

- Permite movimientos de empeoramiento para escapar de óptimos locales. Para evitar recorridos cíclicos, incorpora un mecanismo de generación de vecinos modificado que evita la exploración de zonas del espacio de búsqueda que ya han sido visitadas.
- Emplea mecanismos de reinicialización para mejorar la capacidad del algoritmo para la exploración-explotación del espacio de búsqueda.

Para ello hace uso de dos memorias:

- **Memoria a corto plazo**: guarda información que permite guiar la búsqueda de forma inmediata, desde el comienzo del procedimiento.
- **Memoria a largo plazo**: guarda información que permite guiar la búsqueda a posteriori, después de una primera etapa en la que se han realizado una o varias ejecuciones del algoritmo aplicando la memoria a corto plazo. Esta info se usa para comenzaar la búsqueda desde otra solución inicial de acuerdo a dos filosofías distintas.
    - Intensificar la búsqueda, volviendo a visitar zonas del espacio prometedoras (que contenían buenas soluciones), ya exploradas parcialmente.
    - Diverdificar la búsqueda visitando nuevas zonas no exploradas aún.

Memoria a corto plazo (implicaciones).

La BT extriende la búsqueda local sustituyendo el entorno actual por otro. En cada iter se acepta el mejor becino de dicho entorno tanto si es peor como mejor a las solución actual. La memoria corto plazo es la lista tabú que permite al lagoritmo determinar el nuevo entorno y organizar el espacio que explora. Las soluciones admitidaas en el nuevo entorno dependen de la estructura de la lista tabú.

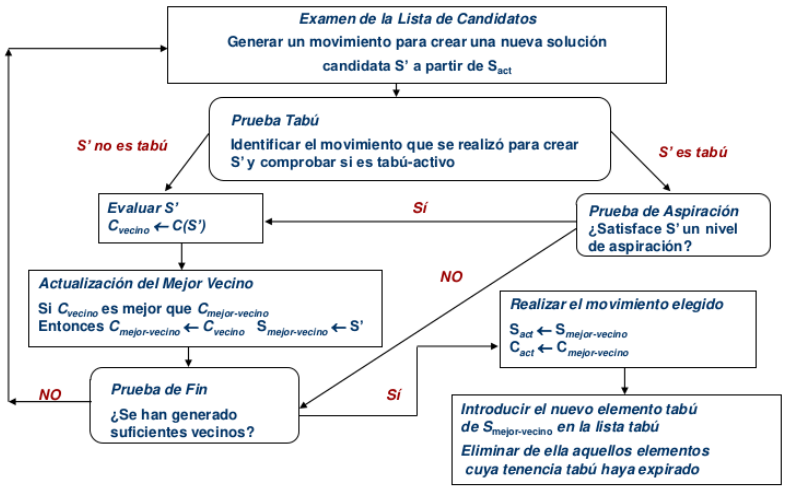
- Lista de soluciones tabú: Se identifican soluciones ya visitadas y se marcan como tabú para no volver a ellas, eliminándolas del vecindario.
- Lista de movimientos tabú: Se eliminan del entorno todos los vecinos resultantes de aplicar sobre S act un movimiento realizado anteriormente.
- Lista de valores de atributos tabú: Se eliminan del entorno todos aquellos vecinos con un par (atributo,valor) determinado que ya presentara alguna solución explorada anteriormente.

Un atributo/movimiento o solución que se haya incluido en la lista tabú en algún momento de la búsqueda no permanece en ella para siempre denominaremos “tenencia tabú” al intervalo de

tiempo durante el que un atributo/movimiento permanece tabú-activo o una solución es tabú. Este parámetro se mide en número de iteraciones. Una vez transcurrido el valor especificado, el elemento en cuestión deja de ser tabú activo y se elimina de la lista.

El "criterio de aspiración" introduce un elemento importante de flexibilidad en la búsqueda tabú. El estado tabú de un movimiento o atributo puede ser ignorado si se cumplen ciertas condiciones, en la forma de niveles de aspiración. Por ejemplo, un vecino que sea mejor que cualquiera de las soluciones encontradas anteriormente merece ser considerado admisible, incluso aunque dicho vecino sea una solución tabú

Las estrategias para la lista de candidatos se usan para restringir el número de vecinos examinados en una iteración dada, para los casos en los que E*(S act) es grande o la evaluación de sus elementos es costosa. Se busca el mejor movimiento disponible que pueda ser determinado con una cantidad apropiada de esfuerzo. Así, no se genera el entorno reducido completo sino una parte del mismo y se toma el mejor vecino. La selección adecuada de candidatos puede reducir apreciablemente los tiempos de ejecución.



Memoria a largo Plazo. (implicaciones)

En algunas aplicaciones, las componentes de la memoria TS de corto plazo son suficientes para producir soluciones de muy alta calidad. No obstante, en general, la TS se vuelve mucho más potente incluyendo memoria de largo plazo y sus estrategias de reinicialización asociadas La memoria de largo plazo se puede usar de dos modos distintos:

- Estrategias de Intensificación: Se basan en una reinicialización de la búsqueda que efectúa un regreso a regiones atractivas del espacio para buscar en ellas más extensamente. Se mantiene un registro de las mejores soluciones visitadas, insertando una nueva solución cada vez que se convierte en la mejor global Se puede introducir una medida de diversificación para asegurar que las soluciones registradas difieran una de otra en un grado deseado. Tenemos 4 variantes resultantes de combinar soluciones/ memoria:
    - Solución desde la que se reinicializa. Que pueden ser: reaunadur desde la mejor solución o usar una pila de mejores soluciones de longitud limitada y escoger el tope de pila.
    - Restauración de la memoria de corto plazo. Que pueden ser: reanudar la memoria a corto plazo de cuando se encontrarno buenas soluciones o borrar la memoria de

corto plazo para iniciar la búsqueda desde cero.

- Estrategias de Diversificación: Conducen la búsqueda hacia nuevas regiones del espacio de búsqueda no exploradas aún. La búsqueda se reinicializa cuando se estanca, partiendo de una solución no visitada. Esta solución se genera a partir de la memoria de frecuencias, dando mayor probabilidad de aparición a los valores menos habituales. Una estructura muy empleada es la memoria de frecuencias, que registra el número de veces que cada valor de un atributo ha pertenecido a soluciones visitadas en la búsqueda. Existen dos posibilidades para el uso de memoria de freciuencias:
  - Generar directamente la nueva solución inicial a partir de la información almacenada en la memoria de frecuencias M, dando mayor probabilidad de aparición a los valores menos habituales
  - Usar la información almacenada en M para modificar temporalmente el caso del problema, potenciando los valores heurísticos de los atributos menos usados en la búsqueda

4. **Algoritmos multiarranque**.

Una Búsqueda con Arranque Múltiple es un algoritmo de búsqueda global que itera las dos etapas:

- Generación de una solución inicial: Se genera una solución S de la región factible. En algunas aplicaciones, la Etapa 1 se limita a la simple generación aleatoria de las soluciones, mientras que en otros modelos se emplean sofisticados métodos de construcción que consideran las características del problema de optimización para obtener soluciones iniciales de calidad
- Búsqueda Local: Se aplica una BL desde S para obtener una solución optimizada S'. se puede emplear una búsqueda local básica, o procedimientos de búsqueda basados en trayectorias más sofisticados.

Estos pasos se repiten hasta que se satisfaga algún criterio de parada. Se devuelve como salida del algoritmo la solución S' que mejor valor de la función objetivo presente. La Búsqueda Multiarranque Básica se caracteriza porque las soluciones iniciales se generan de forma aleatoria. En cuanto a la condición de parada, se han propuesto desde criterios simples, como el de parar después de un número dado de iteraciones, hasta criterios que analizan la evolución de la búsqueda. En muchas de las propuestas que se encuentran en la literatura especializada se fija un número de iteraciones de la búsqueda local

Algoritmo GRASP:

Un algoritmo GRASP es un método multiarranque, en el que cada iteración consiste en la construcción de una solución greedy aleatorizada y la aplicación de una búsqueda local que toma dicha solución como punto inicial de la búsqueda. Este procedimiento se repite varias veces y la mejor solución encontrada sobre todas las iteraciones GRASP se devuelve como salida del algoritmo.

Cuando se utiliza la función de selección para construir una solución greedy, se crea una lista de restringida de candidatos con un número determinado de mejores candidatos. Se realiza una selección aleatoria de un candidato de la lista. Se adapta la función de selección para recalcular la nueva lista de candidatos para el siguiente paso del proceso constructivo.

Existen diferentes formas de construir la lista de LRC (lista restringida de candidatos):

- La variante más habitual consiste en incluir en la LRC los l mejores candidatos de acuerdo a la función de selección. El parámetro l controla la diversidad de generación de soluciones. Puede ser fijo o variable.
- LRC adaptativa (tamaño variable): Existen variantes que incluyen en la LRC todos los candidatos con un valor de selección por encima de un umbral de calidad mu = cmejor +- alpha · (cmejor- cpeor) (cmejor=coste mejor candidato; cpeor=coste peor candidato). Esto provoca que la LRC sea adaptativa y tenga un tamaño variable en cada iteración de la etapa de generación.
- Otras variantes incluyen un sesgo en la LRC con una probabilidad mayor de selección de los mejores candidatos.

La búsqueda local desempeña un papel importante en GRASP ya que sirve para buscar soluciones localmente óptimas en regiones prometedoras del espacio de soluciones. Aunque los algoritmos greedy pueden producir soluciones iniciales razonables para búsquedas locales, su principal desventaja es su falta de diversidad, de ahí las propuestas de LRC para generar más diversidad en la primera etapa del GRASP y mejorar las posibilidades de la búsqueda local.

Una extensión de GRASP es la técnica parh relinking se propuso inicialmente para explorar las trayectorias entre soluciones elite (conjunto de las mejores soluciones obtenidas en varias ejecuciones de la búsqueda tabú u otro algoritmo). Usando una o más soluciones elite, se exploran las trayectorias en el espacio de soluciones que conducen a otras soluciones elite para buscar mejores soluciones. Para generar trayectorias, los movimientos se seleccionan para introducir atributos en la solución actual que estén presentes en la solución elite guía

Otra extensión es GRASP Híbrido, se introduce una mutación (perturbación fuerte) a la solución encontrada tras la etapa de la Búsqueda Local y se repite la optimización.

ILS (BLIterativa):

La ILS está basada en la aplicación repetida de un algoritmo de Búsqueda Local a una solución inicial que se obtiene por mutación de un óptimo local previamente encontrado. La ILS necesita la definición de cuatro componentes:

- Una solución inicial (usualmente, aleatoria): se elige aleatoriamente o mediante una heurística constructiva, como greedy
- Un procedimiento de modificación (mutación) que aplica un cambio brusco sobre la solución actual para obtener una solución intermedia
- Un procedimiento de Búsqueda Local
- Un criterio de aceptación que decide a qué solución se aplica elprocedimiento de modificación.
    - Criterio del Mejor: Este criterio favorece la intensificación.
    - Criterio RW (Random walk): Este criterio favorece la diversificación sobre la intensificación.
    - Criterios Intermedios: Si el algoritmo no mejora la solución durante it_0 iteraciones, se asume que se ha llegado a un óptimo local y se reinicializa parcialmente la solución (mutación fuerte).

Búsqueda de entorno variable (VNS):

La Búsqueda de Entorno Variable (VNS) es una metaheurística para resolver problemas de optimización cuya idea básica es el cambio sistemático de entorno dentro de una búsqueda local (aumentando el tamaño cuando la búsqueda no avanza). La VNS está basada en:

- Un mínimo local con una estructura de entornos no lo es necesariamente con otra.
- Un mínimo global es mínimo local con todas las posibles estructuras de entornos.
- Para muchos problemas, los mínimos locales con la misma o distinta estructura de entorno están relativamente cerca

Existen dos cariantes para la VNS:

- Búsqueda Descendente Basada en Entornos Variables (VND): Algoritmo de BL del mejor cuyo operador de vecino cambia de entorno (ampliándolo) cuando el mejor vecino generado es peor que la solución actual (extensión de la búsqueda local clásica).
- Búsqueda Basada en Entornos Variables (VNS): Algoritmo ILS en el que el operador de mutación cambia de entorno cuando la solución obtenida tras aplicar la BL es peor que la solución actual

El procedimiento básico de VNS es, sea un entorno con un conjunto finito de estructuras vecindario preseleccionada y sea un conjunto de soluciones para el entorno. VNS aplica progresivamente una BL sobre una solución S' obtenida apartir de una mutación de la actual, realizada de acuerdo al tipo de entorno utilizado en cada iteración. Si la última BL efectuada resultó efectiva, es decir, la solución mejoró la actual trabajará con el entorno primero, en otro caso se pasa al siguiente entorno para probocar una perturvaciñón q estará más alejada de la solución inicial de la zona del espacio de búsqueda en la que está situada la actual S.

La seleeción de estructuras de vecindario, sería posible seleccionar diferentes heurísticas para utilizar en cada iteración en la que se aplica la BL. Existen diferentes posibilidades:

- Cambiar los parámetros de los métodos existentes en cada iteración
- Utilizar movimientos de diferente tamaño k para generar vecindarios que aumentan de tamaño de acuerdo al aumento del parámetro k
- Combinar las estrategias previas

5. **Algoritmos evolutivos.**

Evolución Diferencial:

Es un modelo evolutivo que enfatiza la mutación, utiliza un operador de cruce/recombinación a posteriori de la mutación. Fué propuesto para optimización con parámetros reales.

- **Inicialización:** Una población de vectores de parámetros se genera aleatoriamente dentro de unos límites inferiores y superiores previos.
- **Generación: Mutación diferencial:** con respecto a cada vector x i,g en la población actual, llamado vector objetivo, se genera un vector mutado v i,g añadiendo un vector diferencia, escalado y aleatoriamente muestreado, a un vector base aleatoriamente seleccionado de la población actual. En la generación g-ésima, se general una población $P_{u,g}$ consistente de N_p vectores D-dimensionales $u_{i,g} = [u_{1,i,g}, ..., u_{D,i,g}]$ a través de operadores

de mutación y recombinación aplicados a la población actual $P_{x,g}$.

- **Recombinación Discreta:** Con respecto a cada vector objetivo $x_{i,g}$ en la población actual, un nuevo vector $u_{i,g}$ se genera cruzando el vector objetivo $x_{i,g}$ con el correspondiente vector mutado $v_{i,g}$ bajo un ratio predefinido de cruce.
- **Reemplazamiento:** Si el vector $u_{i,g}$ tiene mejor valor de la función objetivo que su correspondiente vector objetivo $x_{i,g}$, sustituye el vector objetivo en la generación (g+1); si esto no ocurre, el vector objetivo permanece en la generación (g+1).

Nubes de partículas:

La "Particle Swarm Optimization" (PSO) es una metaheurística poblacional inspirada en el comportamiento social del vuelo de las bandadas de aves y el movimiento de los bancos de peces. La población se compone de varias partículas (nube de partículas = particle swarm) que se mueven ("vuelan") por el espacio de búsqueda durante la ejecución del algoritmo.

Este movimiento de cada partícula p depende de: Su mejor posición desde que comenzó el algoritmo (pBest), la mejor posición de las partículas de su entorno (lBest) o de toda la nube (gBest) desde que comenzó el algoritmo. En cada iteración, se cambia aleatoriamente la velocidad de p para acercarla a las posiciones pBest y lBest/gBest.

PSO simula el comportamiento de las bandadas de aves. Supongamos que una de estas bandadas busca comida en un área y que solamente hay una pieza de comida en dicha área. Los pájaros no saben donde está la comida pero sí conocen su distancia a la misma. La estrategia más eficaz para hallar la comida es seguir al ave que se encuentre más cerca de ella.

La nube de partículas es un sistema multiagente. Las partículasson agentes simples que se mueven por el espacio de búsqueda y que guardan (y posiblemente comunican) la mejor solución quehan encontrado. Cada partícula tiene un fitness, una posición y un vector velocidad que dirige su "vuelo". El movimiento de las partículas por el espacio está guiado por las partículas óptimas en el momento actual.

Cada particula está compuesta por:

- Tres vectores:
  - El vector X almacena la posición actual (localización) de la partícula en el espacio de búsqueda,
  - El vector pBest almacena la localización de la mejor solución encontrada por la partícula hasta el moemento, y
  - El vector V almacena el gradiente (dirección) según el cuál se moverá la partícula.
- Dos valores de fitness:
  - El x_fitness almacena el fitness de la solución actual (vector X), y
  - El p_fitness almacena el fitness de la mejor solución local (vector pBest).

Tratamiento del algoritmo:

- **Inicialización:** La nube se inicializa generando las posiciones y las velocidades iniciales de las partículas. Las posiciones se pueden generar aleatoriamente en el espacio de búsqueda, de forma regular, o con una combinación de ambas. Las velocidades se generan aleatoriamente, con cada componente en el intervalo, no es bueno inicializarlas a cero.

- **Movimiento de particulas**: Se hace simplemente añadiendo el vector velocidad V_i al vector posición X_i para obtener un nuevo vector posición: $X_i \leftarrow X_i + V_i$. Una vez calculada la nueva posición de la partícula, se evalúa ésta. Si el nuevo fitness es mejor que el que la partícula tenía hasta ahora, pBest_fitness. De este modo, el primer paso es ajustar el vector velocidad, para después sumárselo al vector posición. La velicidad de las particulas poseen dos ratios de aprendizaje cognitivo y social, primero basado en su propio aprendizaje, su mejor posición y el segundo basado en su distancia con la mejor, cuatro modelos: completo, cognitivo, social, social exclusivo.

Algoritmo Meméticos:

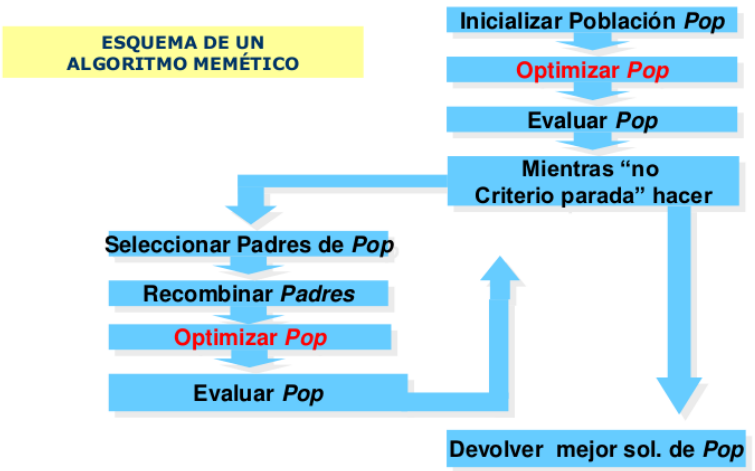
Algoritmo basado en la evolución de poblaciones que para realizar búsqueda heurística intenta utilizar todo el conocimiento sobre el problema (usualmente conocimiento en términos de algoritmos específicos de búsqueda local para el problema).

¿Por que hibridamos? Algoritmos evolutivos son buenos exploradores. Algoritmos de búsqueda local son malos exploradores en global y Algoritmos evolutivos son malos explotadores, Algoritmos de búsqueda local son buenos explotadores.

Los Algoritmos Meméticos (AMs) se construyen sobre la noción de meme. Significado: Unidad de imitación, análoga a un gen pero en el contexto de la "evolución cultural". *Del mismo modo que los genes se propagan en el acervo genético a través de gametos, los "memes" se propagan en el acervo memético saltando de cerebro a cerebro en un proceso que, en un ampliosentido, puede denominarse imitación.*

En los Algoritmos Meméticos se utiliza el término de agentes en lugar de individuos ya que se consideran una extensión de los segundos. Tanto la selección como la actualización (reemplazo), son procesos puramente competitivos. La reproducción es la encargada de crear nuevos agentes (cooperación). Aunque puede aplicarse una gran variedad de operadores de reproducción, existen básicamente dos:

- Recombinación: Realiza el proceso de cooperación. Crea nuevos agentes utilizando principalmente la información extraída de los agentes recombinados. Se suele hablar de combinación inteligente de información.
- Mutación: Permite incluir información externa creando nuevos agentes mediante modificación parcial del agente mutado.



¿Qué algorito de BL usar?:

¿Cúando? Los optimizadores locales, considerados como un operador más, pueden aplicarse de diferentes formas, además para que un algoritmo híbrido sea considerado AM, la Búsqueda Loca siempre debe aplicarse dentro del proceso evolutivo:

- En la fase de inicialización de la población
- En cada generación o cada cierto número
- Como fin del ciclo reproductivo o durante los operadores de recombinación

¿Sobre qué? Sobre toda la población, agentes resultantes de la reproducicción o sobre un subconjunto de ella:

- Sobre el mejor
- Representantes de clases tras un proceso de agrupación
- Una probabilidad de actuación de la busqueda local.

¿Cómo aplicarlo? Existen dos modelos clásicos:

- Lamarkiano: El agente resultante del proceso de optimización local se introduce en la población (cede su genotipo) y reemplaza en la población al agente sobre el que se inició el proceso o al más cercano.
- Baldwiniano: El agente inicial del proceso de optimización local recibe el fitness del agente final pero no su genotipo (cede su fitness).

Al aplicar los optimizadores locales, es esencial regular adecuadamente el equilibrio entre:

- Anchura (frecuencia de aplicación del optimizador) (uso de probabilidad de actuación de la Búsqueda Local).
- Profundidad (intensidad del optimizador).

6. **Algoritmos de adaptación social.**