1. **Explica brevemente (máximo 1 página) qué criterios utilizas para seleccionar la metaheurística a aplicar en la resolución de un problema.**

Lo primero sería hacer un estudio del tipo de problema al que nos enfrentamos, ya que las distintas metaheurísticas son mejores o peores según el tipo de problema al que se enfrente.

También deberíamos tener en cuenta que nuestro trabajo sea lo más cómodo posible, y ahí entra en juego la representación de las soluciones. Algunas metaheurísticas trabajan mejor con codificación binaria que otras y viceversa.

Otro criterio a explorar sería las soluciones que necesito y el objetivo que persigo, por ejemplo, ya que hay numerosos problemas multimodales para los que puede ser interesante usar una metaheurística u otra, además de que hay numerosas estrategias de hibridación que pueden hacer la metaheurística que elijamos mucho más potente, todo depende de los resultados que persiga.

1. **Explica las diferencias y similitudes entre en un algoritmo voraz y la búsqueda local.**

El algoritmo voraz, parte de un elemento de la solución inicial que tiene tamaño m y va construyendo la solución añadiendo elementos conforme a una heurística.

La búsqueda local, sin embargo, parte de una solución (aleatoria por ejemplo), y explora su vecindario (soluciones alcanzables a partir de la nuestra por algún tipo de movimiento en sus elementos) en busca de una mejor solución (su función objetivo sea mejor) o de la mejor solución. Termina tras un cierto número de iteraciones o cuando no encontramos mejora.

La diversificación de soluciones y exploración del entorno de soluciones es mucho mayor en la búsqueda local que en el Greedy. Además, la BL es parte importante de algoritmos como ILS, que intensifican la búsqueda en las mejores soluciones encontradas y además mejoran la exploración.

Ambos algoritmos pueden ser usados en problemas de optimización de diversa índole. Algunos algoritmos incluso se ayudan de ambos para obtener mejores soluciones.

1. **¿Utilizarías una codificación binaria para resolver el problema del viajante de comercio? ¿Por qué?**

No, puesto que haría más complicado la codificación del problema.

El problema del viajante de comercio se basa en encontrar el camino mínimo para recorrer varias ciudades. Con la codificación típica, se usa un vector de números que no se pueden repetir y las diferentes soluciones son permutaciones de este vector, donde influye el orden, ya que partimos de la “ciudad” que marca la primera posición del vector.

Si pasamos a una codificación binaria, lo más lógico sería representar que se recorre el arco entre una ciudad i y una j como 1, es decir (i,j) = 1. Aquí aparece la restricción de que el siguiente arco recorrido debe empezar por j.

No la usaría, porque, aunque sea factible emplearla, complica los operadores típicos de generación de vecinos e incluso la generación de soluciones aleatorias. La representación por reales es mucho más cómoda.

1. **¿Usarías una codificación basada en números reales para resolver el problema de la mochila? ¿Por qué?**

No, puesto que la codificación binaria me permite trabajar en mismo vector con los n objetos, que tienen 0 si no están incluidos en la mochila y 1 en caso contrario. Usar una representación basada en números reales haría que mi vector tuviera solo los objetos de la mochila. Los intercambios de objetos serían más farragosos y los operadores típicos de los distintos algoritmos estudiados más difíciles de implementar.

1. **Explica qué enuncia el Teorema de “No Free Lunch” y qué implicaciones tiene en el contexto de las metaheurísticas.**

El teorema dice que para cualquier algoritmo, cualquier desempeño elevado en un tipo de problemas es pagado en el rendimiento de otro tipo de problemas.

Esto tiene implicaciones en las MH puesto que nos dice que ganar en un cierto dominio implica perder en los restantes. Es decir, podemos hibridar los algoritmos con otros para mejorar el comportamiento del primero.

1. **Enumera los esquemas de enfriamiento que conoces para usar con el enfriamiento simulado. Indica sus ventajas e inconvenientes.**

Los distintos mecanismo de enfriamiento son:

* Enfriamiento basado en sucesivas temperaturas descendentes fijadas por el usuario.
* Enfriamiento con descenso constante de la temperatura.
* Descenso geométrico: Tk+1 = alfa \* Tk. Siendo alfa una constante cercana a 1.
* Criterio de Boltzmann: Rk = T0/(1+log(k))
* Criterio de Cauchy: Tk = T0/(1+k)
* Cauchy modificado (para controlar el número de iteraciones): Tk+1 = Tk/(1+beta\*Tk)

Para ejecutar exactamente M iteraciones beta = (T0-Tf)/(M\*T0\*Tf).

En general ningún mecanismo es mejor que otro porque depende del problema. Lo normal es hacer pruebas de ensayo y error para quedarse con el mejor.

1. **Explica en qué consiste el equilibrio entre intensificación y diversificación, y cómo se contempla en las diversas metaheurísticas estudiadas.**

Todos las metaheurísticas estudiadas comparten objetivos de explotación y exploración. El primero es la intensificación del proceso de búsqueda y es una característica de búsqueda local. El segundo es la diversificación y es una característica de búsqueda global.

El equilibrio correcto de intensificación-diversificación conduce hacia una metaheurística efectiva.

En las diversas metaheurísticas estudiadas (variantes básicas), se presenta de la siguiente manera:

* ES: equilibrio dinámico. Vamos desde una mayor diversificación y menor intensificación hacia una menor diversificación y mayor intensificación (se consigue con la temperatura).
* BT: equilibrio estático. Mientras mayor sea la lista tabú, menor intensificación habrá y mayor diversificación.
* ILS: depende del criterio de aceptación. Intensificación si es Aceptación del mejor y Diversificación si es último óptimo local aceptado.
* VNS: balance oscilatorio.
* AGs: equilibrio dinámico. Igual que enfriamiento simulado.

1. **Explica los argumentos a favor del uso de la paralelización de las metaheurísticas. Da un ejemplo de metaheurística paralela y explica sus ventajas.**

El paralelismo hace que la carga de trabajo se distribuya y conseguir mejores resultados con menos esfuerzo computacional.

Por ejemplo, las múltiples búsquedas independientes se han probado exitosamente con distintos algoritmos basados en trayectorias como ES, VNS, TS y GRASP. Pero tienen la desventaja de que las diferentes ejecuciones no comparten información entre ellas.

Tenemos también las múltiples búsquedas cooperativas, puesto que el intercambio de información entre procesos (que puede ser según diversas políticas, como estrella o anillo), se realiza durante el proceso de búsqueda y no solo al final. La clave está en qué información compartimos entre procesos. Este modo de paralelizar cobra especial importancia en los algoritmos basados en poblaciones.

1. **Explica las técnicas de metaheurísticas híbridas y da las diferencias básicas entre ellas: partículas (PSO), algoritmos de colonias de hormigas (ACO) y algoritmos evolutivos. Comparadlas 2 a 2 (ACO vs PSO, ACO vs AEs, PSO vs AEs).**
2. **Funcionamiento y propósito de los algoritmos de búsqueda multiarranque. Diferencias y similitudes.**

El propósito de los algoritmos de búsqueda multiarranque es usar algún método de búsqueda local iterativamente a una solución según la política y o bien, devolver el mejor algoritmo encontrado hasta el momento o devolver el mejor de entre una serie de soluciones devueltas por una serie de ejecuciones (que sería el BMB).

En resumidas cuentas, un algoritmo con arranque múltiple **Genera una solución y la guarda en mejor solución.** Ahora repetimos hasta un cierto criterio de parada lo siguiente: **aplicamos búsqueda local o ES por ejemplo a la Solución Actual, si la solución generada es mejor que la mejor solución entonces la sustituimos. Ahora generamos otra solución en Solución Actual y repetimos.**

Existen distintos algoritmos que se basan en la búsqueda multiarranque. El BMB parte de un conjunto de soluciones iniciales a las que se le aplica un algoritmo por trayectorias. De entre todas las soluciones, nos quedamos con la mejor. Esto puede presentar problemas puesto que, si las soluciones generadas al principio aleatoriamente están cercanas entre sí (en términos de vecindario), convergerá repetidamente en el mismo óptimo local el algoritmo de búsqueda.

En cuanto a las demás, tenemos GRASP, que se basa en la construcción de una solución greedy aleatorizada y la aplicación de una BL que toma dicha solución como punto inicial. Este procedimiento se repite varias veces y la mejor solución encontrada por todas las ejecuciones es la que se devuelve.

ILS por su parte se basa en la aplicación repetida de algún algoritmo como BL o ES a una solución inicial que se obtiene por mutación de un óptimo local previamente encontrado.

Para VNS, por último, es un algoritmo ILS en el que el operador de mutación cambia de entorno cuando la solución obtenida tras aplicar la BL es peor que la actual.

Por tanto, todos los métodos se parecen en el uso de algún algoritmo por trayectorias, pero cada uno de ellos trata la exploración y explotación de forma diferente, así como sobre qué individuos se efectúa la BL.

1. **Explicad las similitudes y diferencias entre la búsqueda local, el enfriamiento simulado y la búsqueda tabú.**

Las principales similitudes son que todos ellos son métodos de búsqueda basados en trayectorias, que parten de una solución inicial y que de una manera u otra, con unas estrategias diferentes, recorren el vecindario de una solución en busca de mejores soluciones.

En cuanto a las diferencias, la búsqueda local no permite movimientos de empeoramiento sobre la solución actual, ni modificar la estructura de entornos. La búsqueda tabú sí que permite tanto el empeoramiento de la solución actual como la modificación de la estructura de entornos. El enfriamiento simulado, por su parte, permite el empeoramiento de la solución actual si nos encontramos lejos de converger todavía.

1. **Describir los aspectos más críticos en el diseño de un Algoritmo memético para un problema concreto.**

Hay que tener en cuenta que no existe un procedimiento sistemático para diseñar un algoritmo memético para un problema. Podemos considerar heurísticas de diseño que resultarán muy probablemente en un AM efectivo.

La principal pregunta que nos podemos hacer en el diseño es, ¿qué algoritmo de BL utilizo?, a lo que la respuesta es que depende del problema. Lo que está claro es que para que un algoritmo híbrido se considere AM, la BL siempre debe aplicarse dentro del proceso evolutivo.

Tienen importancia también cómo se adapta la BL, cómo se aplica el optimizador y cuándo o sobre qué agentes se aplica.

1. **Explica similitudes y diferencias entre las técnicas PSO y Evolución Diferencial.**

Ambas técnicas son modelos evolutivos usados para optimización de variables continuas con múltiples óptimos y no diferenciables. La representación de las soluciones se hace con números reales en ambas técnicas.

En la Evolución Diferencial, los cambios en las soluciones se basan en un operador de mutación y posteriormente una recombinación entre el vector objetivo y el mutado nos da un nuevo vector, que sustituirá al objetivo si es mejor que este en términos de la función objetivo.

PSO tiene un enfoque menos matemático, ya que se inspira en el comportamiento de las bandadas de aves. Las partículas van volando por el espacio de soluciones con velocidad variable durante la ejecución del algoritmo. Este movimiento depende bien de la mejor posición de las partículas de su entorno (es decir, la mejor solución), o de la mejor de toda la nube encontrada desde que comenzó el algoritmo. Es decir, se diferencian principalmente en el modo de explorar el entorno de búsqueda en busca de mejores soluciones, uno haciéndolo mediante cambios según una cierta política y posterior recombinación según una cierta política y otro mediante el movimiento de las partículas hacia zonas mejores del espacio de búsqueda.

1. **Explica similitudes y diferencias entre Algoritmos Genéticos y PSO.**

Ambos son métodos evolutivos que generan nuevas soluciones en un entorno vía dos padres. La principal diferencia radica en que en los genéticos se hace con el operador de cruce y en PSO via atracción de las partículas hacia las mejores posiciones.

Otra diferencia puede ser que los genéticos son adaptables a problemas combinacionales, mientras que PSO está pensado para problemas de optimización continua.

Por otro lado, PSO tiene ventaja sobre AG porque converge más rápido debido al continuo movimiento de las partículas y tiene más diversidad en las trayectorias de búsqueda que hace.

1. **Explica similitudes y diferencias entre los Algoritmos Genéticos y la Evolución Diferencial.**

Ambos son ejemplo de métodos evolutivos. Podríamos decir que la DE es una forma de AG en el sentido de que el genotipo es un vector de reales y que la mutación y el cruce entre dos o más vectores de la población se hace mediante la diferencia de vectores o mediante otro método, en definitiva, trabajamos con números reales. En ambos algoritmos la mutación tiene la función de aportar diversidad.

Una de las principales diferencias es entonces la representación, que en los genéticos es binaria y en la evolución diferencial es con números reales.

1. **Explica las similitudes y diferencias entre los Algoritmos de Optimización Basada en Colonias de Hormigas y los algoritmos de PSO (Particle Swarm Optimization).**

PSO es un método basado en poblaciones motivado por el comportamiento de las bandadas de aves y que se usa típicamente para problemas de optimización continua.

ACO sin embargo es una técnica probabilística para resolver problemas no-determinísticos.

1. **Análisis comparativo entre los algoritmos genéticos con nichos (para problemas multimodales) y el algoritmo CHC. Pros y contras.**

**CHC**

Algoritmo evolutivo que se puede considerar una variante del AG típico estudiado introduciendo un equilibrio entre diversidad y convergencia a través de operadores específicos de cruce, de no posibilitar el incesto, operadores que no estaban presentes en las versiones clásicas de AG estudiadas. Por tanto, una de los pros de CHC es que mejora los AG, que ya producían resultados aceptables para ciertos problemas de combinatoria. Otro pro puede ser la compenetración entre explotación y exploración.

Uno de los contras, es que en problema multimodales, es decir, donde varias soluciones son óptimos admisibles. El problema es que los AG convergen muy rápido hacia una sola solución, por lo que los AG con nichos, permiten que el algoritmo evolucione a zonas de una población que permite tener soluciones en diferentes zonas del espacio de búsqueda.

1. **Describe el algoritmo evolutivo CHC, indicando cuáles de sus cuatro componentes destacadas tienen un comportamiento exploratorio y cuáles lo tienen explotación.**

CHC es una propuesta de AG que introduce un equilibrio entre la diversidad y la convergencia dado que combina una selección elitista que preserva los mejores individuos que han aparecido hasta el momento con un operador de cruce que produce hijos muy diferentes a sus padres. Esto lo hace a partir de cuatro componentes novedosas con respecto a los AG: la selección elitista, el cruce uniforme HUX, la prevención de incesto y la reinicialización.

La exploración forma parte de: cruce HUX, prevención de incesto y la reinicialización.

La explotación forma parte de la selección elitista.

1. **Explica el propósito y el funcionamiento de la regla de actualización de feromona en línea, paso a paso, del Sistema de Colonias de Hormigas.**

En los SCH, cada hormiga modifica automáticamente la feromona de cada arco que visita para diversificar la búsqueda.

Cada vez que una hormiga recorre un arco, aplica la actualización local de feromona:



Con esta operación, la feromona asociada a un arco disminuye cada vez que lo visita una hormiga. Los arcos visitados van siendo menos prometedores según los recorren más hormigas en la iteración actual, lo que favorece la exploración de los arcos no visitados. Esto se hace para que las hormigas tiendan a no converger a soluciones parecidas en la iteración actual.

1. **Es conocido el hecho de que un buen algoritmo de búsqueda debe tener un equilibrio entre exploración/diversificación y explotación/intensificación. Los algoritmos que hemos estudiado tienen distintas componentes para poner en práctica cada una de las dos tareas mencionadas. Proponer un algoritmo híbrido combinando algunas de estas componentes y que pudiesen tener buenos resultados.**
2. **Colonias de Hormigas (CH) frente a Algoritmos Evolutivos (AEs). Caracteriza dos tipos de problemas, uno en el que el comportamiento de las CH sea muy bueno (normalmente mejor que los AEs) y viceversa. (Problemas tipo A – mejor CH, Problemas tipo B – mejor AEs).**

CH trabaja mejor con problemas no deterministas de búsqueda de caminos óptimos en grafos.

AE trabaja mejor con problemas de optimización continua de funciones multidimensionales no derivables.

1. **Analizar la influencia del tamaño de la población en el funcionamiento de los diferentes algoritmos evolutivos. Indicar cuando el tamaño debe ser grande o pequeño. Dar ejemplos de intervalos aproximados para las diferentes propuestas.**
2. **Indicar las diferencias entre los Algoritmos de Colonias de Hormigas y los Sistemas de Hormigas (ACS vs AS). ¿Cuál crees que funciona mejor? Razona la respuesta.**

Las principales diferencias son que el ACS extiende al AS en tres aspectos:

* La regla de transición establece un equilibrio entre la exploración de nuevos arcos y la explotación de la información acumulada.
* Para la actualización global de feromona sólo se considera la hormiga que generó la mejor solución hasta ahora. Es decir, sólo se evapora feromona en los arcos que componen esta.
* Se añade una nueva actualización local de feromona, basada en que cada hormiga modifica automáticamente la feromona de cada arco que visita para diversificar la búsqueda.

El ACS funcionará mejor, ya que hay más equilibrio entre la exploración y la explotación, y la actualización local consigue que los arcos ya visitados vayan siendo menos prometedores según los recorren más hormigas, por lo que las hormigas tienden a no converger a soluciones parecidas.

1. **Explica los fundamentos de los algoritmos de Evolución Diferencial. (Características diferenciadoras frente a otras técnicas, y el algoritmo).**

Los algoritmos de Evolución Diferencial es un modelo evolutivo que enfatiza la mutación, utiliza un operador de cruce/recombinación a posteriori de la mutación. Está pensado para optimización con parámetros reales.

El algoritmo consta de varias partes:

* Una población de Np vectores de parámetros D-dimensionales.
* Generación: mutación diferencial, que con respecto a cada vector en la población actual, llamado vector objetivo, se genera un vector mutado añadiendo un vector diferencia, escalado y aleatoriamente muestreado, a un vector base aleatorio de la población actual.
* Recombinación discreta: para cada vector objetivo en la población actual, un nuevo vector se genera cruzando el vector objetivo con el mutado.
* Reemplazamiento: si el vector resultante tiene mejor valor de función que su correspondiente vector objetivo, se sustituye.

La política es parecida a la de los AG, pero se usan para problemas de optimización de variables continuas, y los operadores cambian un poco. Tanto la mutación como la recombinación discreta se pueden hacer siguiendo diversas políticas.

1. **Define las características de la metaheurística ideal.**

La Metaheurística ideal podría reunir, entre otras, las siguientes características:

* Conocimiento completo del problema.
* Útil con cualquier tipo de representación.
* Útil con cualquier tipo de problema.
* Encuentra el mínimo o soluciones muy cercanas a este en poco tiempo.
* Equilibrio perfecto entre exploración y explotación del espacio de búsqueda de las soluciones.
* Ajuste de parámetros sin ser necesario pruebas de ensayo y error, según algún tipo de regla.