**Práctica Final. Diseño óptimo de un parque eólico con algoritmos meta-heurísticos de optimización (nota máxima: 4 puntos)**

**Introducción**

El diseño óptimo de parques eólicos (layout de turbinas) es un problema clásico de optimización que ha sido abordado en numerosas ocasiones en la literatura científica. En numerosos trabajos, este problema ha sido abordado a partir de algoritmos meta-heurísticos de optimización como los que hemos visto en esta asignatura. En esta práctica se propone el diseño del layout óptimo de turbinas en un parque eólico, a partir de un modelo sencillo y de la implementación de un algoritmo meta-heurístico, que puede ser un algoritmo genético, un algoritmo de Harmony Search o un Temple Simulado. Cualquier otra opción de algoritmo de tipo meta-heurístico puede ser aceptable (hay muchísimos tipos, muchos más que los vistos en clase incluso), pero antes de decidiros por implementar otro meta-heurístico, consultad por favor con el profesor de esta parte de la asignatura.

**Modelo de parque y codificación del problema**

El modelo de parque eólico a implementar será un grid cuadrado de 400 celdas (20 filas x 20 columnas). El número de turbinas a posicionar será de 20 (exactamente). El problema con el posicionamiento de las turbinas lo vimos en clase: cada turbina afecta a la velocidad de viento, de modo que genera una estela. En la estela la velocidad de viento disminuye, de modo que dependiendo de la dirección y velocidad del viento, el posicionamiento de las turbinas óptimo será uno u otro. En este caso vamos a considerar el posicionamiento de turbinas en el parque dependiendo de una simulación de viento de 250 valores (dato del problema). El modelo de estela y la función objetivo serán también datos del problema (tendréis el código .m de dicha función objetivo). El objetivo del problema es maximizar la potencia generada en el parque (maximizar el valor que os dé la función objetivo), a la que tendréis que pasar una matriz binaria de tamaño 20x20, con exactamente 20 1s.

**Codificación del problema:**

La codificación del problema en el algoritmo la elegís vosotros. Os hago una serie de observaciones al respecto:

1. Independientemente de la codificación, a la función objetivo le tenéis que pasar una matriz binaria 20x20 con 20 1s. Un 1 en la matriz implica que en esa posición habéis situado un aerogenerador. Un 0 significa que en esa posición no posicionáis generador.
2. Si elegís codificar en el algoritmo directamente la matriz binaria, tenéis que definir con mucho cuidado el operador de cruce tendría que ser un cruce entre matrices, puede ser complicado.
3. Una opción alternativa (creo que mejor) es convertir la matriz binaria 20x20 en un vector binario de longitud 400. En este caso podéis implementar los operadores de cruce tradicionales que hemos visto en clase (1 o 2 puntos), pero recordad que al evaluar el individuo tenéis que pasar el vector a formato matriz 20x20.
4. Si optáis por una codificación binaria, tenéis además el problema de controlar el número de 1s tras los operadores de cruce y mutación (siempre tiene que haber 20 1s en la codificación binaria, representando 20 turbinas). Tendréis que corregir el individuo tras cada cruce/mutación. Si la mutación la implementáis como un swap entre 0 y 1, no tendréis que corregir.
5. Otra posible codificación del problema sería un vector de 20 posiciones, donde cada posición contenga los valores (x,y) de las celdas a posicionar. Notad que en este caso, como x representa las filas e y las columnas de la posición de las turbinas, 1<=x<=20 y 1<=y<=20. Esta codificación os asegura que siempre tendréis 20 turbinas en vuestro parque, pero tenéis que tener cuidado de que no haya dos (x,y) iguales en la codificación.

**Otras consideraciones**

Con esta práctica encontraréis un fichero .m con la función objetivo, y un fichero con los datos de viento simulado (250 valores de módulo y dirección). Si **X** es una matrix binaria 20x20 con 20 1s, la llamada a la función para obtener la calidad de **X** como layout se haría así:

[pwr\_T2,gan\_T2,cost\_T2,obj\_T2] = f\_powerPlantsT\_fast(vVec,gr)

Donde vVec representa el vector de vientos por horas de la simulación, gr es vuestra matriz 20x20 con 0s y 1s, y pwr\_T2 representa la potencia total devuelta por la función.

Junto con la práctica os dejo un artículo que realiza un layout con un algoritmo aleatorio. Es sólo informativo, y lo podéis usar para escribir la memoria de la práctica. No os voy a pedir en ningún caso que implementéis el algoritmo aleatorio del artículo, pero si lo hacéis y comparáis con vuestro meta-heurístico lo valoraré positivamente, por supuesto.