

Práctica: Diseño óptimo de un parque eólico con algoritmos meta-heurísticos de optimización

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, un Temple Simulado o un algoritmo de Corales. Cualquier otra opción de algoritmo de tipo meta-heurístico puede ser aceptable (hay muchísimos tipos, muchos más que los vistos en clase incluso).

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), o un grid cuadrado de 2500 celdas (50 x50). El número de turbinas a posicionar será de 20 y 50 (exactamente). Comenzad por el caso de 20x20, y cuando tengáis buenas soluciones, pasad al de 50x50. Notad que 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 1 año (8760 valores horarios, 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 (o 50x50, con 50 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 (o la correspondiente al caso 50x50). 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 (la de 50x50 en un vector binario de longitud 2500). En este caso podéis implementar los operadores de cruce tradicionales de un EA (1 o 2 puntos), pero recordad que al evaluar el individuo tenéis que pasar el vector a formato matriz binaria 20x20 (50x50).

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 20/50 1s en la codificación binaria, representando 20/50 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 20/50 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, $1 \leq x \leq 20$ y $1 \leq y \leq 20$ (50 en su caso). Esta codificación os asegura que siempre tendréis 20/50 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 (os la doy yo), y un fichero con los datos de viento simulado (8760 valores de módulo y dirección). Si **gr** es una matrix binaria 20x20 con 20 1s, la llamada a la función para obtener la calidad de **gr** 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 binaria 20x20 o 50x50 que representa el *layout*, y **pwr_T2** representa la potencia total devuelta por la función (valor a maximizar). Tened cuidado porque en el fichero .zip que os paso, **gr** es aleatorio para tener un ejemplo. Evidentemente tenéis que pasar a la función como **gr** cada uno de las soluciones que queráis evaluar en vuestro algoritmo, no soluciones aleatorias.

Junto con la práctica os dejo un par de artículos de investigación que podéis usar para escribir la memoria final, fundamentalmente la introducción, y para comparar las soluciones que obtenéis con las de un algoritmo pseudo-aleatorio. Valoraré que busquéis otros diferentes a los que os doy yo.

A continuación, os dejo también los criterios de corrección que seguiré para puntuar este trabajo. La entrega de la práctica será vía Aula Virtual, con fecha tope de entrega la acordada en clase, y se entregará un documento pdf con la práctica a evaluar.

Criterios de corrección, práctica final

1. Introducción

- a. ¿la memoria tiene introducción más allá de decir lo que se ha hecho en la misma?
- b. ¿Se pone de relevancia la importancia del problema, su importancia, sus antecedentes y se han analizado trabajos previos?
- c. ¿Hay referencias adecuadas citadas en la introducción?
- d. ¿Finalmente se describe brevemente el contenido de la memoria y su estructura?*

2. Contenido/algoritmos

- a. ¿Se describe la aproximación que se va a implementar correctamente?
- b. ¿Se dan detalles de la implementación a nivel teórico?*
- c. ¿Se establecen los principales componentes de explotación y exploración del algoritmo, así como su codificación, restricciones y cómo manejarlas, etc.?
- d. ¿Se usa código para describir el algoritmo? Si es así, está bien explicado, comentado y tiene sentido?
- e. ¿Las figuras han sido copiadas de alguna otra fuente y en este caso se ha mencionado dicha fuente de origen?

3. Experimentación

- a. ¿Se ha analizado la función objetivo, qué significa y qué operaciones realiza?*
- b. ¿Las soluciones son de buena calidad (en el caso 20x20 por encima de 5290 o 5300 MW)
- c. ¿Se han hecho suficientes experimentos para comprobar la validez del algoritmo?
- d. ¿Hay una representación apropiada, en base a figuras de evolución, del funcionamiento del algoritmo?
- e. ¿Hay tablas de comparación con otros algoritmos?
- f. ¿Las mejores soluciones han sido dibujadas, y se ha dado el valor de fitness de las mismas?
- g. ¿Se ha comparado con distintas variantes del algoritmo implementado?
¿Se ha ideado algún tipo de modificación del algoritmo para mejorarlo?

4. Conclusiones y referencias

- a. ¿Se han realizado unas conclusiones sobre el trabajo serias, donde se establecen los pros y contras de usar este tipo de algoritmos, y se discuten las consecuencias de obtener soluciones de buena calidad en este problema?
- b. ¿La memoria tiene referencias serias, bien citadas, donde se han incluido los artículos que se distribuyeron con la práctica (al menos, entre otros)?

La nota de la práctica se establecerá a partir de comprobar que se dan estos elementos en la misma. Soluciones por debajo de 5280MW en el caso 20x20 son realmente malas (se

pueden obtener por búsqueda aleatoria), tenedlo en cuenta cuando implementéis el algoritmo. Los puntos con * son opcionales, los valoraré positivamente pero si no están, no penalizarán. El resto de puntos son claves para tener una buena nota en la práctica. La corrección de la práctica se realizará en el global de la misma, como un todo en el que deberían estar reflejados de alguna manera los puntos resaltados anteriormente.