Computo Evolutivo
Tarea 4: Evaluación NSGA_II con funciones del grupo WFG

Cesar Amilkar Rivera Covarrubias

August 2024

Esta implementación se puede separar en tres partes fundamentales, el uso de la implementación de las funciones WFG en el *Toolkit*, la implementación del algoritmo NSGA-II implementado para la tarea pasada y la implementación del cálculo de hipervolumen.

A continuación, describire brevemente las partes más relevantes para este trabajo de cada una de las partes.

- Toolkit: Este es un conjunto de archivos que contienen todas las transformaciones necesarias para aplicar las funciones **WFG**, en este caso, este conjunto fue proporcionado como material de clase, por lo que no se profundizara mucho en él. La manera en como se incorporo para ser utilizado con los archivos principales fue incluyendo la cabecera ExampleProblems.h para poder llamar a las funciones utilizando la siguiente linea

  $$using\ WFG::Toolkit::Examples::Problems::WFGX;\ ,$$

  dónde X representa el número del problema a utilizar. Es importante tener en cuenta que todos los archivos necesarios deben ser incluidos en la compilación.

- NSGA-II: Se crearon cuatro archivos importantes, dos cabeceras y dos de implementación, el primero de ellos contiene una refactorización del codigo que se entregó en la Tarea 3, implementando el algoritmo como una función disponible en la cabecera "nsga_2.h", en el otro archivo se incluyeron dos estructuras importantes para llevar a cabo esta implementación, son las clases *Individuo* que nos permite guardar todas las características de un individuo de la población que se utiliza en el algoritmo NSGA; además se implemento la clase *Función* que permite almacenar una función que se le pasará al algoritmo NSGA. Es resaltable que esta versión del algoritmo NSGA-2 recibe como parámetro si se busca maximizar o minimizar la función, pero solo esta adaptada para dos objetivos, el código puede ser manipulado para que funcione con k objetivos. Estas dos estructuras están disponibles a través de la cabecera "estructuras.h".

- Calculo de hipervolumen: En este caso, el hipervolumen en dos dimensiones se puede interpretar como el área, de manera brevem se ordenan las soluciones y se suman las aportaciones de las regiones dominadas por cada solución. Este lo usaremos como un indicador sobre la calidad de un frente de soluciones en relación a un punto, en este caso usamos el puntos (2.2, 4.4).

Nota: la linea para compilar que utilice fue g++ -O2 -o solver Tarea4.cpp nsga_2.cpp estructuras.cpp ExampleProblems.cpp ExampleShapes.cpp ExampleTransitions.cpp FrameworkFunctions.cpp Misc.cpp ShapeFunctions.cpp TransFunctions.cpp hipervolumen.cpp

Elegí las funciones 2, 3 y 7, en la figura 1 podemos ver el resultado de cien generaciones con una población de 100 individuos, los puntos de colores representan el frente alcanzaro, mientras que los puntos grises representan la población inicial.

En la figura 2 podemos observar el resultado de la misma configuración de cien individuos después e cien generaciones, además, podemos observar el punto (2.2, 4.4) que utilizaremos como punto de referencia.

Por otro lado, para verificar la eficacia del algoritmo, se hizo una ejecución con quinientos individuos y en la figura 3 podemos observar el resultado tras quinientas ejecuciones.

El principal cambio que podemos observar es que el frente de la función WFG2 se alcanza a cubrir de mejor manera. El resto de frentes también tienen una mejor representación, sin embargo, el resultado

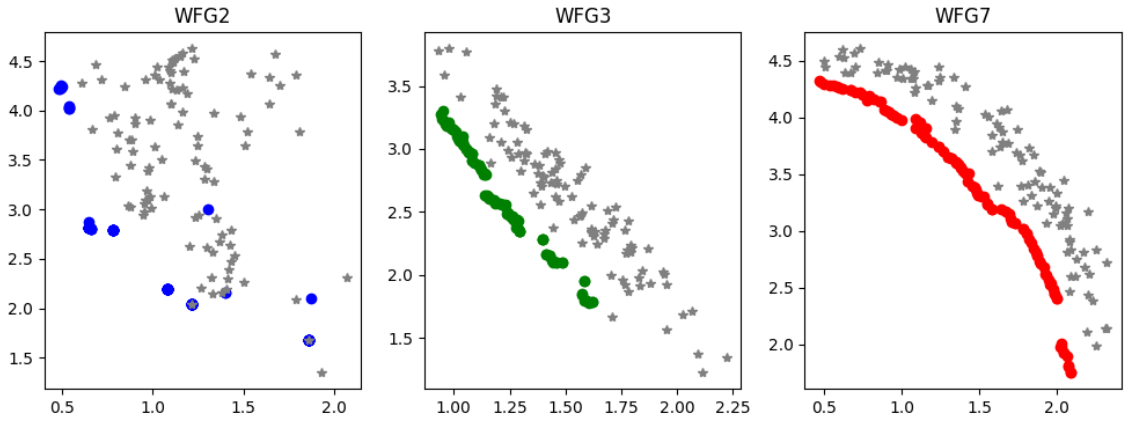

Figure 1: Resultado de 100 generación con una población de tamaño 100.

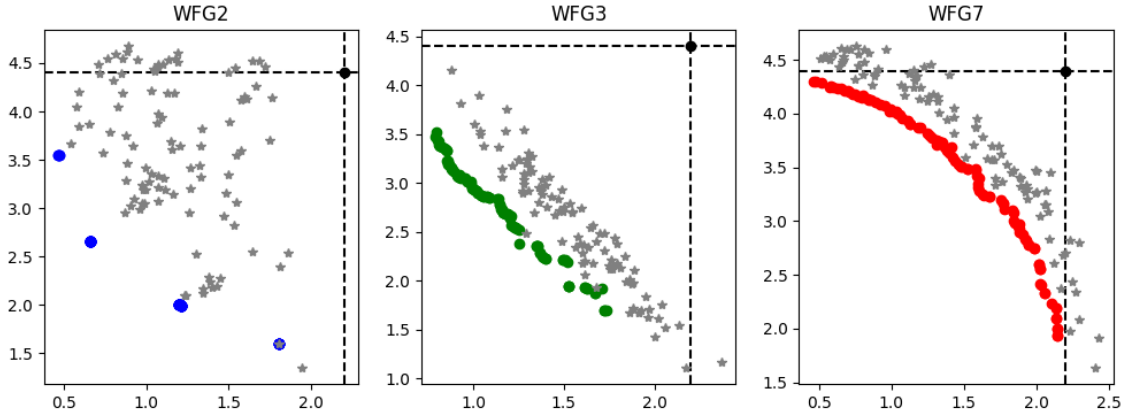

Figure 2: Resultado de cien generaciones con población de tamaño 100 incluyendo el punto de referencia para el calculo del hipervolumen.

con la población de cien individuos se aproximaba de buena manera.

Durante el proceso de evaluaciones se hicieron experimentos variando el parámetro de cruce y mutación, en la siguiente figura mostramos un mapa de calor que representa la relación entre las variaciones.

En la figura 4 se representan las relaciones entre las variaciones, se realizaron 16 ejecuciones para cada algoritmo con cada una de las combinaciones posibles de los parámetros de cruza, 1, 10, 20, 30; así como los parametros de mutación, 1, 25, 50, 75. Los resultados que se muestran son normalizados. En las tres gráficas podemos observar que la columna que corresponde al parametro de cruce mas bajo siempre se contiene el valor más alto del hipervolumen, mientras, que la fila, que corresponde a la tasa de mutación, va variando, además, de que el segundo valor más alto, no se encuentra en la misma columna que el primero, corresponde a otra combinación de parametros.

Podemos intuir que los valores óptimos de cada parámetro son propios de cada problema a solucionar.

La información en la figura nos permite concluir que si bien, la configuración de los parámetros es importante, no es única, pues se pueden repetir los resultados, por ejemplo, en el mapa de calor de la función WFG2 en las celdas [1, 3] y [4, 1], en el formato [i, j] para la celda con la fila i y la columnas j.

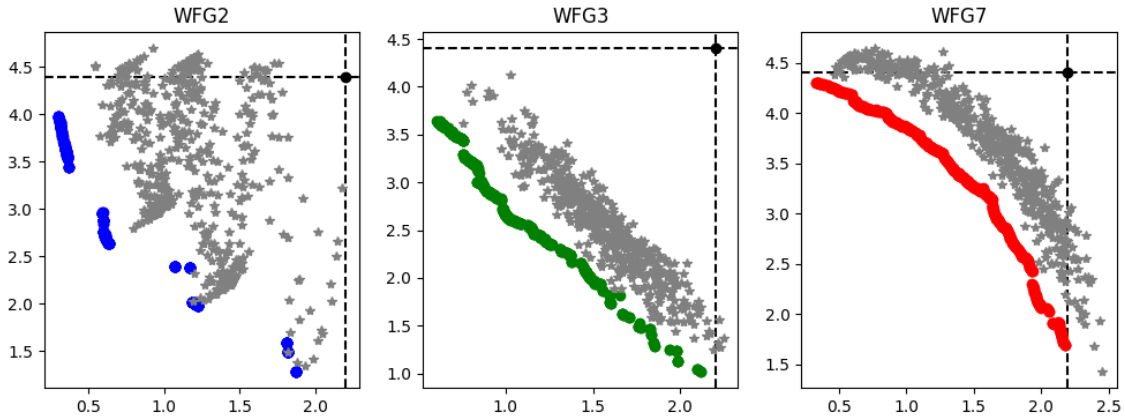

Figure 3: Resultado de la ejecución de quinientas generaciones con quinientos pobladores

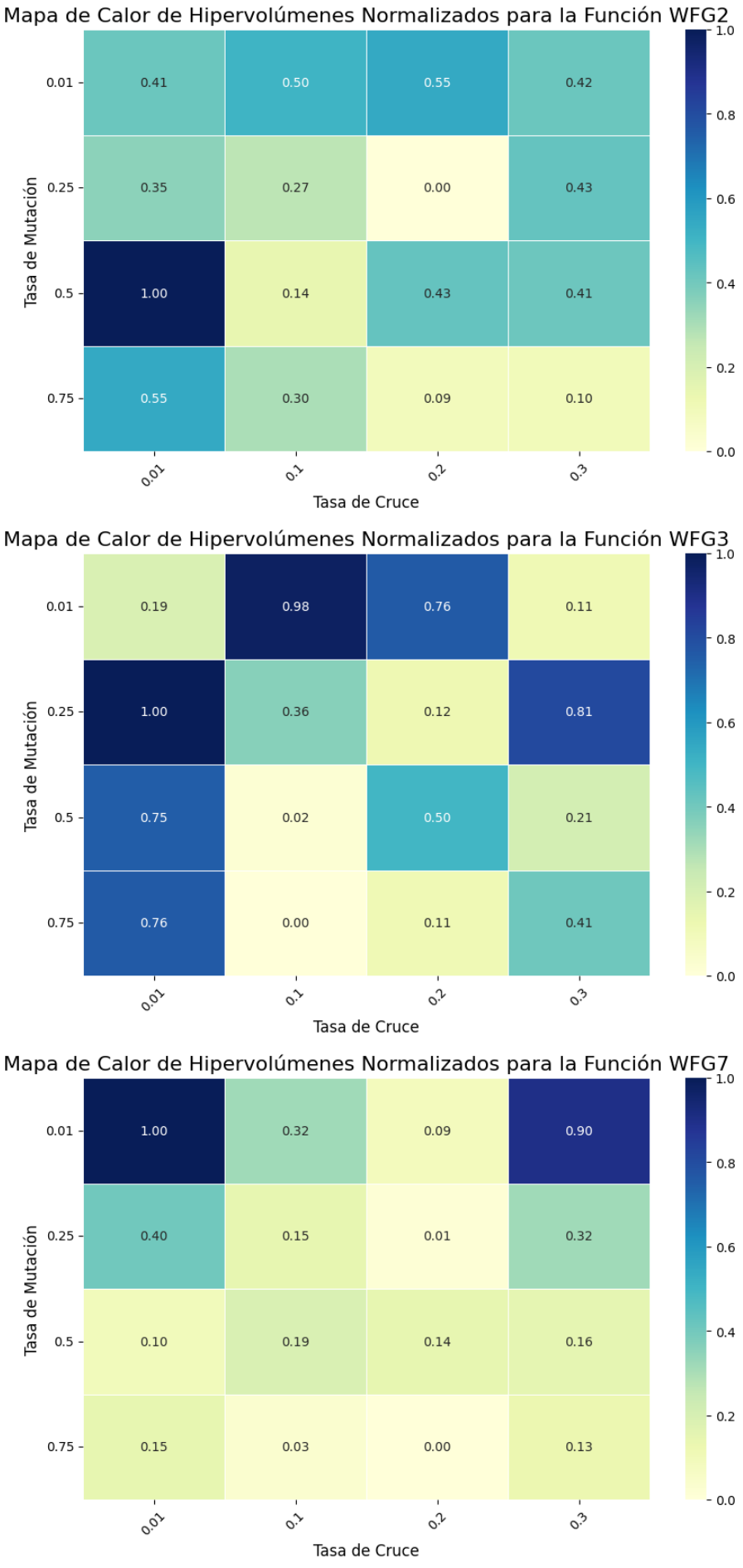

Figure 4: Mapas de calor que relacionan los parámetros internos de mutación y cruce