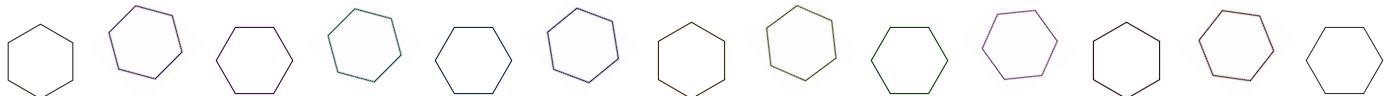


ASC BLOQUE 3 ALGORITMOS GENÉTICOS

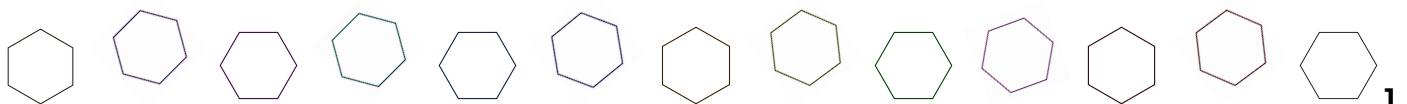
Mateos Gómez, Fernando José. fermatgom@alum.us.es

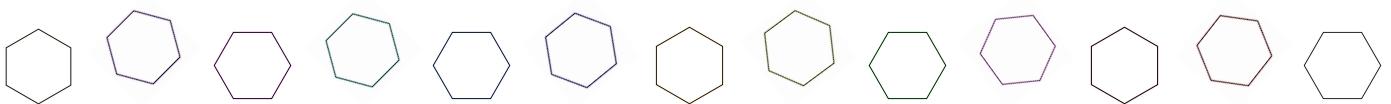
Repository: [asc_problems](https://github.com/fermatgom/asc_problems)



Index

| | |
|---------------------|-----------|
| Overview | 2 |
| GENERAL | 2 |
| ZDT3 | 4 |
| 10.000 PRESTACIONES | 5 |
| 4.000 PRESTACIONES | 6 |
| CF6 | 8 |
| 4 DIMENSIONES | 8 |
| 16 DIMENSIONES | 10 |
| Conclusiones | 11 |





Overview

En este documento se analizarán los 2 algoritmos presentados para el ejercicio evaluable de la asignatura de ASC.

Se ha codificado el código en Lenguaje Perl.

Se han analizado los resultados y las métricas correspondientes a 10K evaluaciones y 4K evaluaciones, por cada problema.

Es importante destacar que no se ha logrado resolver en su totalidad el ejercicio, llegando a mejorar la solución de NSGA II, esto será algo que se estudiará en este documento.

GENERAL

En este apartado explicaré los pasos realizados para ambos problemas, ya que comparten la gran mayoría de características salvo parámetros de entrada.

Como ya se ha explicado, ambos problemas se van a resolver optimizando 2 funciones.

Para evaluarlas aplicaremos la Formulación de Tchebycheff, con la cual seremos capaces de darle pesos a las soluciones obtenidas por los fenotipos que se han generado.

Antes de comenzar con cada evaluación, siempre se generan un listado de valores aleatorios(de tamaño **D**, que se generarán hasta **N**, siendo N el número de subproblemas, y un vector **B**, el cual almacenará, por cada subproblema que exista, un listado de parejas de valores.

Estas parejas de valores deberán de estar ordenadas de menor a mayor distancia euclídea, y tendrán un tamaño asignado debido al parámetro de vecindad, denominado **T** a partir de ahora.

Se evaluarán, en base a la función del problema, y guardaremos el menor valor que se obtenga para cada función objetivo.

Tras esto, por cada subproblema, reproduciremos cada fenotipo con aquellos que se encuentran en su vecindad, **B(i)**.

En mi caso, he encontrado, que a pesar de que las soluciones no sean las más acertadas, he tenido buenos resultados con 3 padres, un coeficiente de cruce de 1/2, y el mismo valor como factor de mutación.

Siendo los 3 padres distintos entre sí, y asegurándose de que al menos un campo del cromosoma del nuevo hijo, sea generado por una mutación.

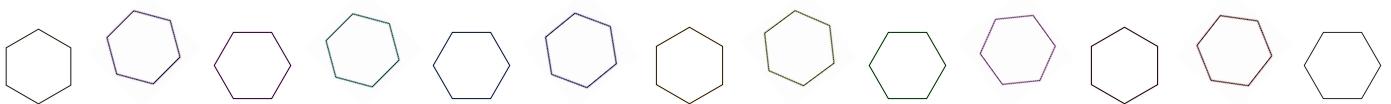
Tras esto se ha evaluado a la solución que se ha obtenido en el paso de la reproducción, y se han sustituido los valores de la mejor función objetivo si ha procedido.

El siguiente paso es el más complejo y con el que no se ha podido mejorar la solución.

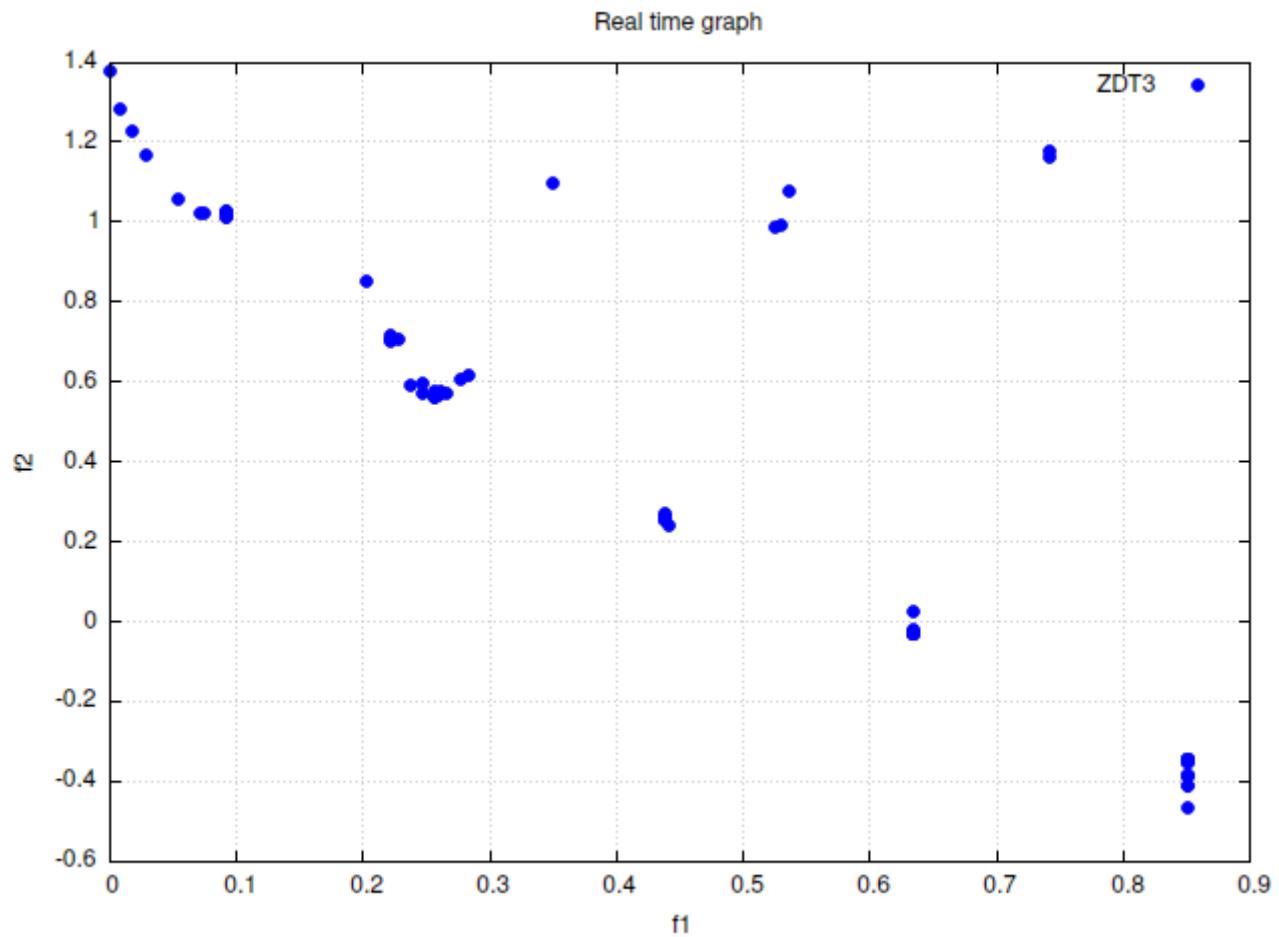
A la hora de actualizar los vecinos, buscamos sustituir el valor del fenotipo **X[j]** de aquellos que se encuentren en la vecindad del subproblema con el que estamos trabajando.

Obtenemos los pesos asignados a cada función objetivo, de ese vecino, y aplicando la Formulación de Tchebycheff, si comparamos los resultados al reproducir el individuo respecto al vecino, es decir en el momento en el que una Función Objetivo ha logrado mejorar en algún aspecto, respecto a las demás, y es mejor que el mejor resultado de su vecino, entonces se modifica el vecino.





Según se ha analizado con las gráficas, e aquí una:



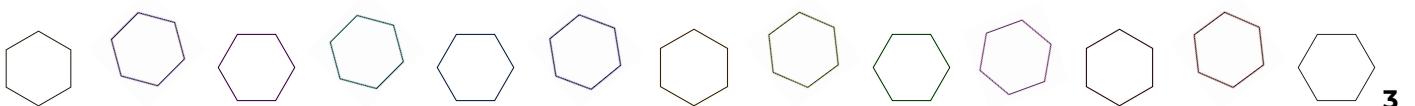
Observamos que a pesar de tener el contorno inicial de la función ZDT3, existen varios de estos puntos que coinciden con los máximos de la función.

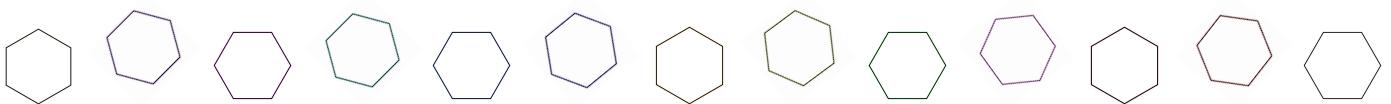
Es decir, de cierta manera, cada vez que se han actualizado los vecinos, se estaban modificando otros vecinos con los pesos asignados a otros vecinos, ya que parece que se estaba trabajando con otros pesos, ocasionando que se desplazaran, sin embargo no se ha sabido cómo solventar este problema.

Al principio se ha creido que se debía a un problema a la hora de la reproducción, ya que se comprobó que tanto el listado de **B(i)** como la lista de fenotipos, siempre, en ambos problemas, se ha mantenido dentro del margen de sus valores establecidos, y las ventanas se encontraban conformadas por los subproblemas correspondientes y en el orden esperado.

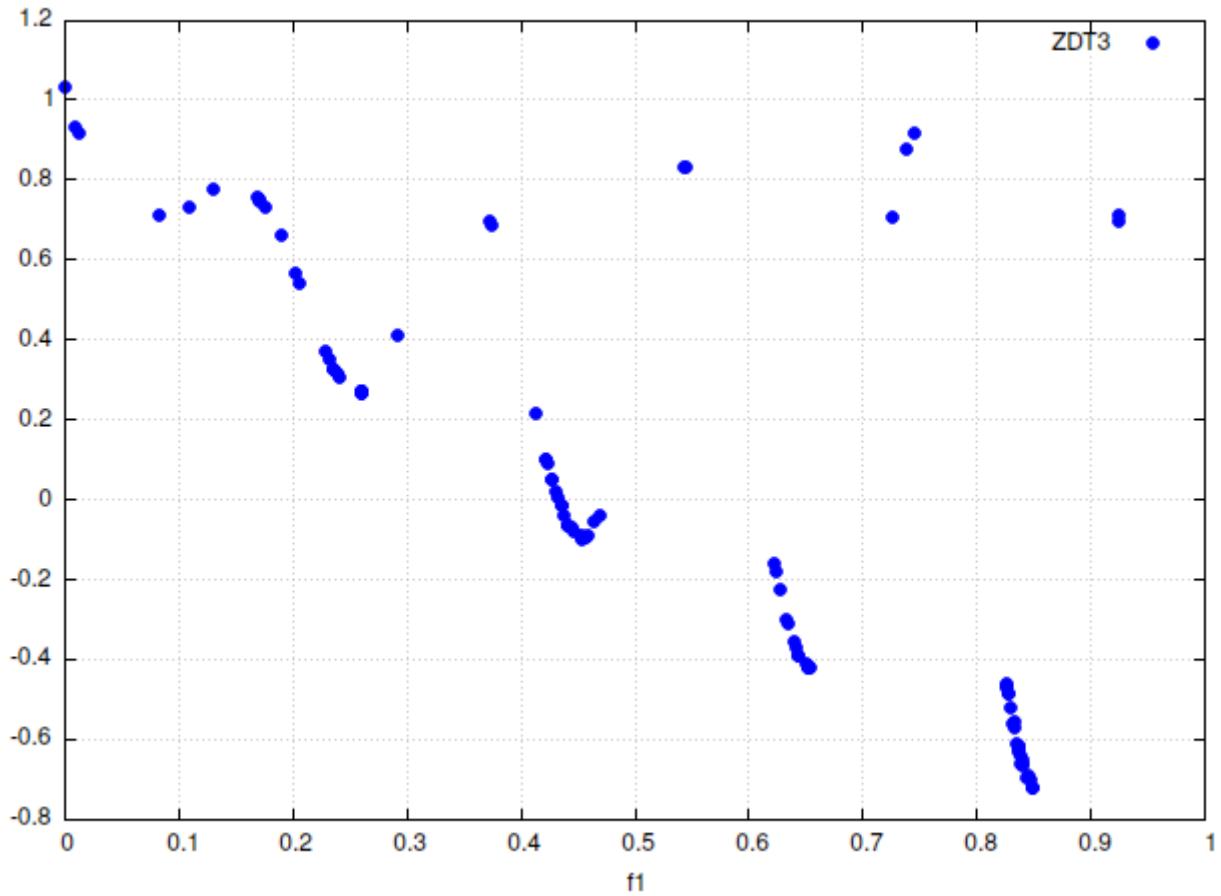
Se estuvo probando con una variación de parámetros tanto en el índice de cruce, como en el de mutación y el parámetro de vecindad **T**.

El parámetro que más llegó a afectar de entre todos, fue sin duda la vecindad, el número de individuos por subproblema, se estuvo probando con valores entre [10%,30%], dando mejores resultados con menos vecinos que con más.





Real time graph



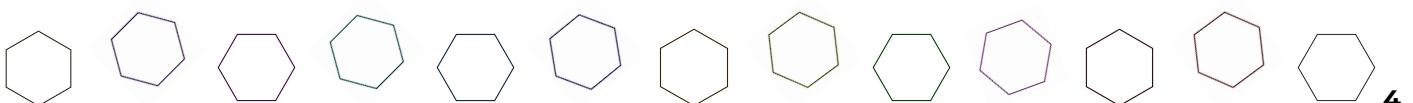
Esta conclusión sacada a partir de numerosos experimentos, donde, a pesar de que se formaba el frente de soluciones, siempre se acumulaban un cierto número de individuos en los máximos de la función.

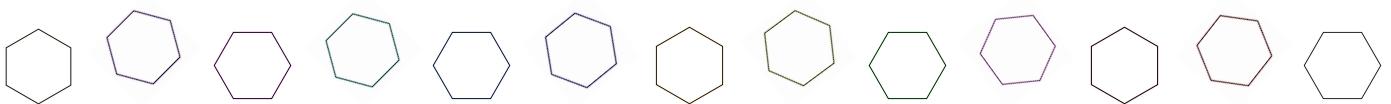
Esto se ha considerado un error completo, por el motivo anterior, y cuando mayor ha sido el número de generaciones, más notorio se ha vuelto, mostrando cada vez más el contorno de toda la función.

ZDT3

Con el análisis preliminar realizado en el apartado anterior continuaremos con las evaluaciones sobre este problema.

Estudiando las prestaciones para 4K, y 10K, partiendo del error anterior, se ha notado que a más prestaciones, y menor vecindad, era posible llegar a una solución más cercana al Frente de Pareto. A partir de ahora las métricas y gráficas usadas poseerán un 10% de vecindad y un 50% de Cruce, 50% de Factor de Mutación, y 10 experimentos.





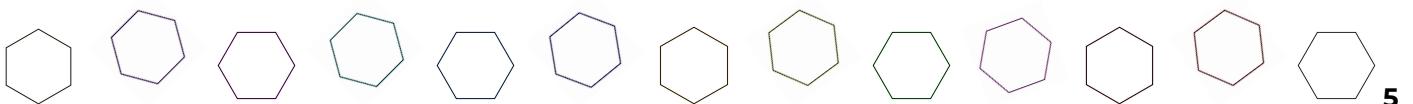
10.000 PRESTACIONES

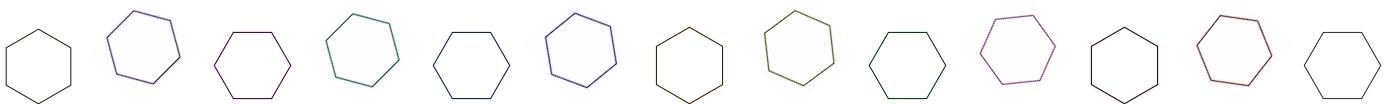
Se ha analizado los resultados más aceptables usando el software de métricas de forma masiva, comparando 10 ejecuciones de NSGA II y mi solución (100 Generaciones y 100 Individuos). A título informativo, cuando pone UNKNOWN, es debido a que se modificó lo mínimo e imprescindible un script de análisis para poder trabajar con los datos.

```
Front 1 hypervolume: 1.4338652413
Front 1 spacing: 0.0047575538
Configuration UNKNOWN p100g100: Hypervolume mean: 1.1673
Configuration UNKNOWN p100g100: Hypervolume standard deviation: 0.221192
Configuration NSGAIID p100g100: Hypervolume mean: 1.44209
Configuration NSGAIID p100g100: Hypervolume standard deviation: 0.00786969
Configuration UNKNOWN p100g100: Spacing mean: 0.0310956
Configuration UNKNOWN p100g100: Spacing standard deviation: 0.0128854
Configuration NSGAIID p100g100: Spacing mean: 0.00488778
Configuration NSGAIID p100g100: Spacing standard deviation: 0.000611439
```

Observamos distintos datos que son de utilidad a la hora de justificar el problema encontrado. Las métricas de Espaciado se busca que tengan un valor reducido, respecto a las del NSGAII para que sean mejores, pero el hecho de que sean bastante más grandes, implica que las soluciones abarcan un mayor rango de valores dentro del espacio de búsqueda, y tardaría más en converger. Algo que veremos al analizar con menos prestaciones.

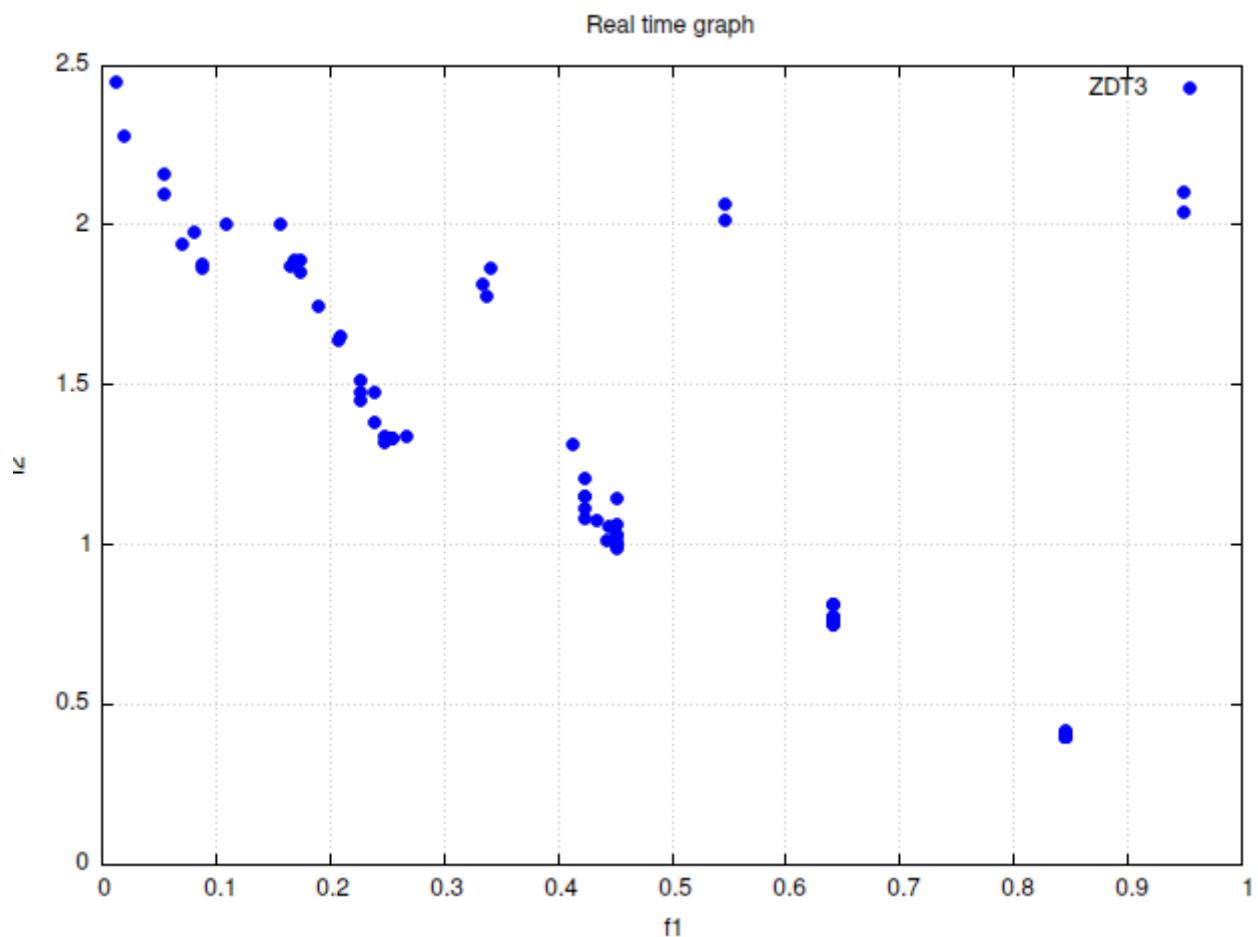
En el caso del hipervolumen pasa exactamente lo mismo, buscamos aumentar el espacio de soluciones no dominadas con el hipervolumen, a pesar de que gráficamente pueda implicar que la solución no es mala, al haber una diferencia tan grande, esto implica que estamos perdiendo un gran número de soluciones no dominadas en el problema.





4.000 PRESTACIONES

En este caso, observamos que la solución converge con menor sentido a la esperada.



Las soluciones llegan a converger pero no son tan precisas que si se tuvieran más prestaciones.

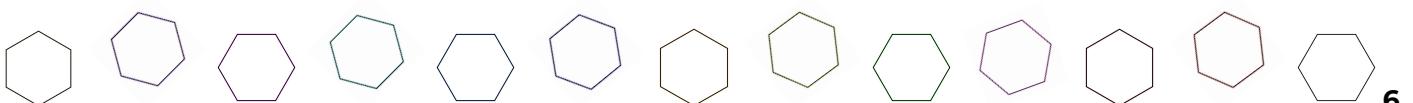
```

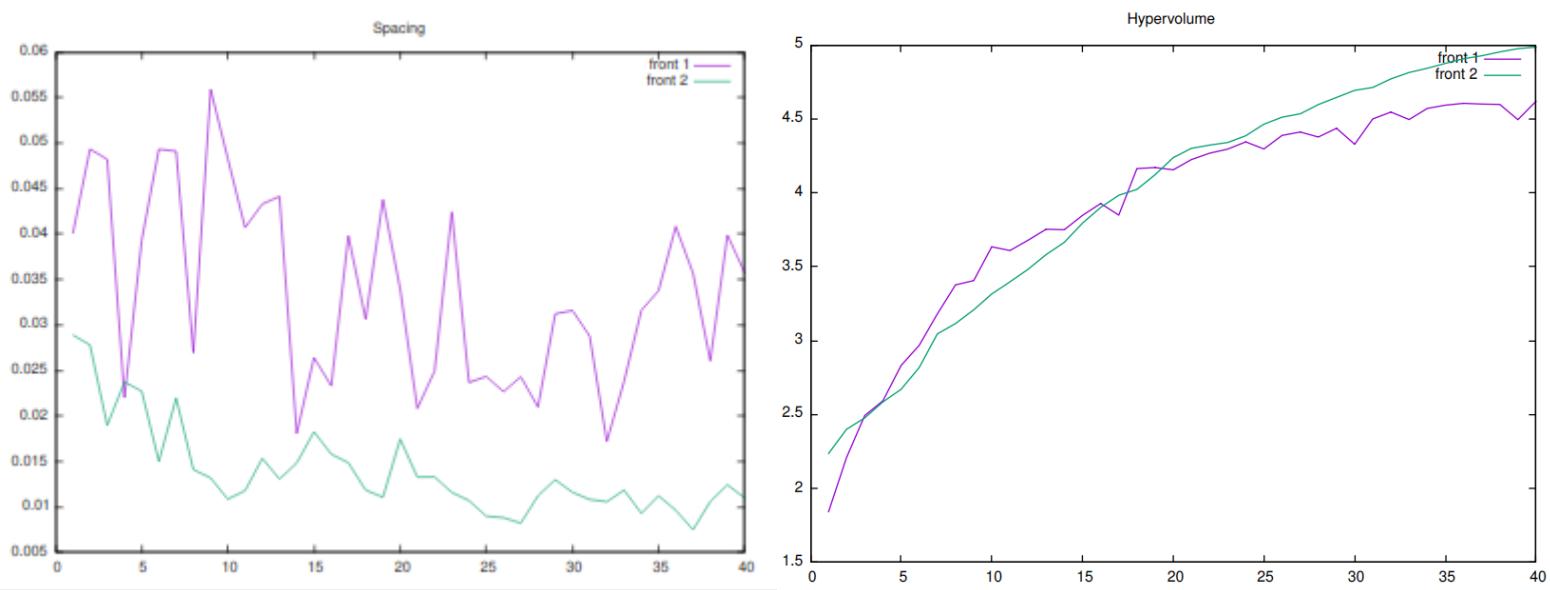
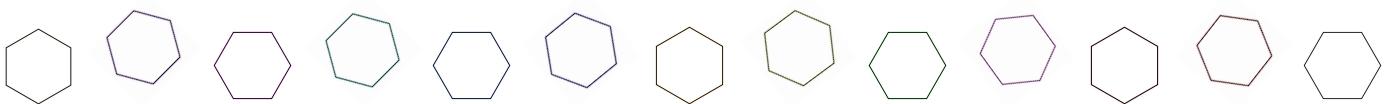
Front 1 hypervolume: -0.1065473539
Front 1 spacing: 0.0000000000
Configuration UNKNOWN p100g40: Hypervolume mean: 0.0946735
Configuration UNKNOWN p100g40: Hypervolume standard deviation: 0.094835
Configuration NSGAIIf p100g40: Hypervolume mean: 0.128874
Configuration NSGAIIf p100g40: Hypervolume standard deviation: 0.130326
Configuration UNKNOWN p100g40: Spacing mean: 0
Configuration UNKNOWN p100g40: Spacing standard deviation: 0
Configuration NSGAIIf p100g40: Spacing mean: 0
Configuration NSGAIIf p100g40: Spacing standard deviation: 0

```

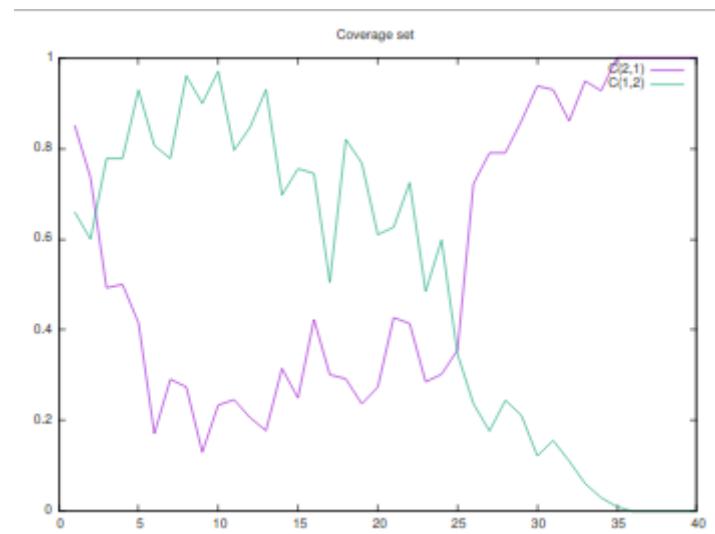
En cuanto a las métricas, dado que los resultados son especialmente peores no podemos compararlos de ninguna manera en cuanto a la diferencia entre los 2 resultados respecto al espaciado de las soluciones.

Podemos observar especialmente que el espacio de búsqueda se ha reducido pero que el rango de valores que se han probado ha aumentado, aumentando la imprecisión.

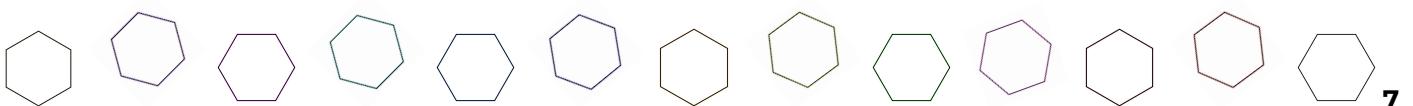


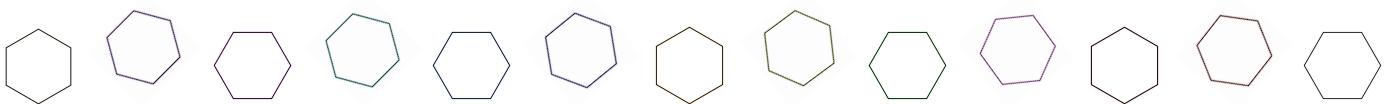


Si comparamos los valores del frente 1 respecto al del 2(solución propuesta o solución NSGA II), como ya se ha visto con los resultados numéricos, estos varían a lo largo del tiempo, considerando demasiados valores, haciéndolo menos preciso y variable. Es decir, está probando y actualizando valores que empeoran el resultado final.



Observamos como el frente 1 sobre el 2 llega a una solución más precisa, o al menos no se estanca con rapidez, mientras que el frente 2 sobre el 1 parece llegar a un punto donde pierde muchisima diversidad.





CF6

Este problema para funcionar con mayor precisión requiere de un control de restricciones. Se estuvo estudiando 2 alternativas:

- Un control de restricciones con pesos estáticos, que variaría el valor de la función fitness en caso de que se violase una restricción.
- El uso de una función peso, la cual asignará un peso variable(menos al principio del programa y mayor cuanto más tiempo lleven). De esta forma podría ser capaz de explorar más opciones y restringir los valores al final de una ejecución

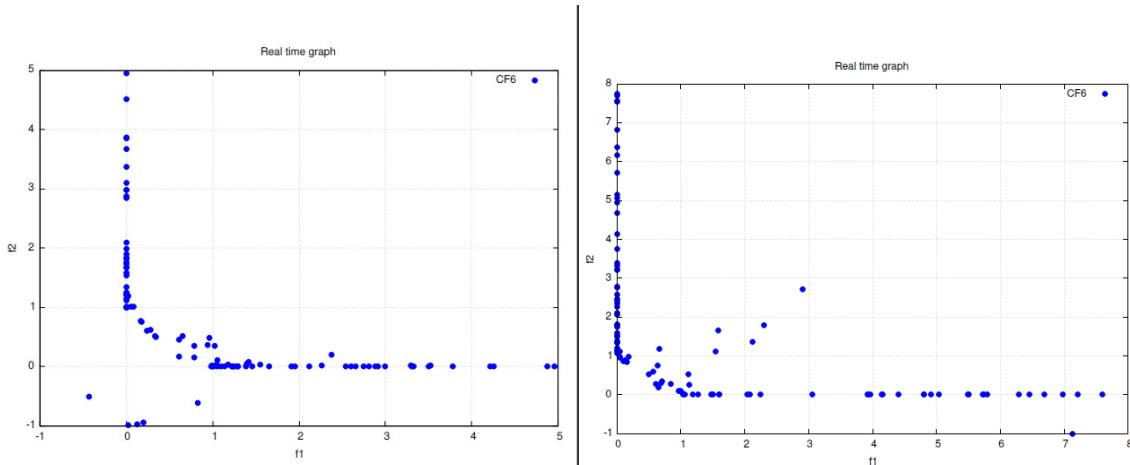
De entre las 2 se tomó por la segunda, tras una breve prueba, que se ha visto que llevaría mucho tiempo probar con muchos valores.

La idea a probar fue la siguiente.

Consideraremos dos funciones:

g(x) y **w(t)**, la primera de ambas indicará con un 0 o un 1 si el individuo viola las restricciones, y se multiplicará por la segunda función, la cual se ha asumido como un valor entre 0 y 1, similar a un porcentaje. Para esto se ha creado una función sigmoidal que se encarga de esto, con un factor de crecimiento.

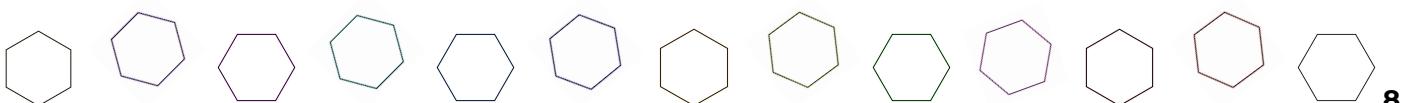
Aún tomando esta decisión, los resultados son peores a los que se esperarían, debido a la actualización de vecinos. Notándose en ambos casos, 4 y 16 dimensiones, que se calcula un frente de máximos que oculta la solución verdadera.

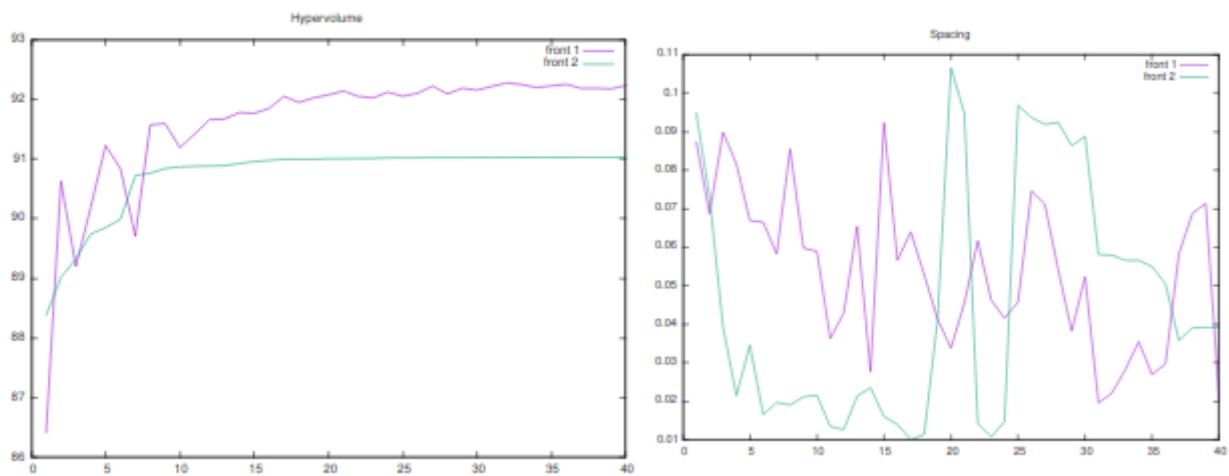
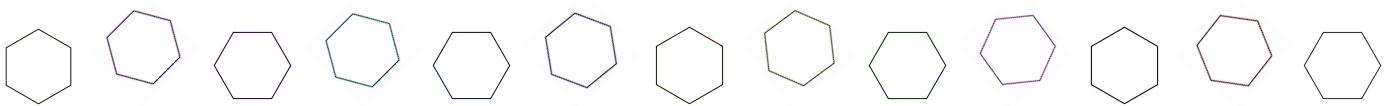


La imagen de la izquierda representa un caso de 100 individuos y 40 generaciones, 4 dimensiones y la derecha una de 100 generaciones y 16 dimensiones.

4 DIMENSIONES

Analizaremos primero con 4000 prestaciones, las cuales nos muestran unos resultados como los enseñados anteriormente





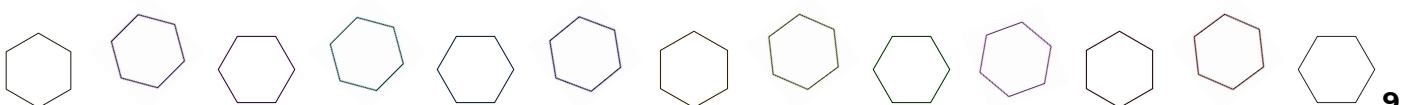
Observamos que entre las dos métricas, los resultados del primer frente, respecto al segundo abarcan un espacio mayor de búsqueda y se mueven sobre el mismo espacio de soluciones, no saliéndose de ese rango, llegando a parecer mejores que los propuestos por la asignatura. Se ha llegado a observar una clara mejoría en las métricas cuando el número de prestaciones aumenta.

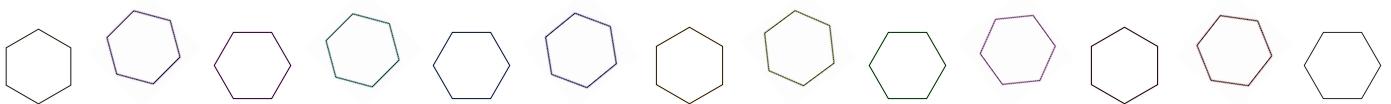
```
Front 1 hypervolume: 15.7154784331
Front 1 spacing: 0.0205152255
Front 2 hypervolume: 15.5142404473
Front 2 spacing: 0.0392330039
C(front2,front1)=0.3100000000
C(front1,front2)=1.0000000000
```

Según las métricas observamos que el espacio de búsqueda es ligeramente más bajo en el dataset original, igual que el espaciado es ligeramente inferior en el dataset. Esto se puede comprender como un aumento en la precisión y espacio de trabajo del algoritmo, exceptuando por la actualización de vecinos, lo cual se puede observar gráficamente cuando los datos generan un frente ad-infinitum paralelo al eje **F2**.

```
Front 1 hypervolume: 32.7641375831
Front 1 spacing: 0.0231296802
Front 2 hypervolume: 31.9939186867
Front 2 spacing: 0.0420950940
C(front2,front1)=0.1600000000
C(front1,front2)=1.0000000000
```

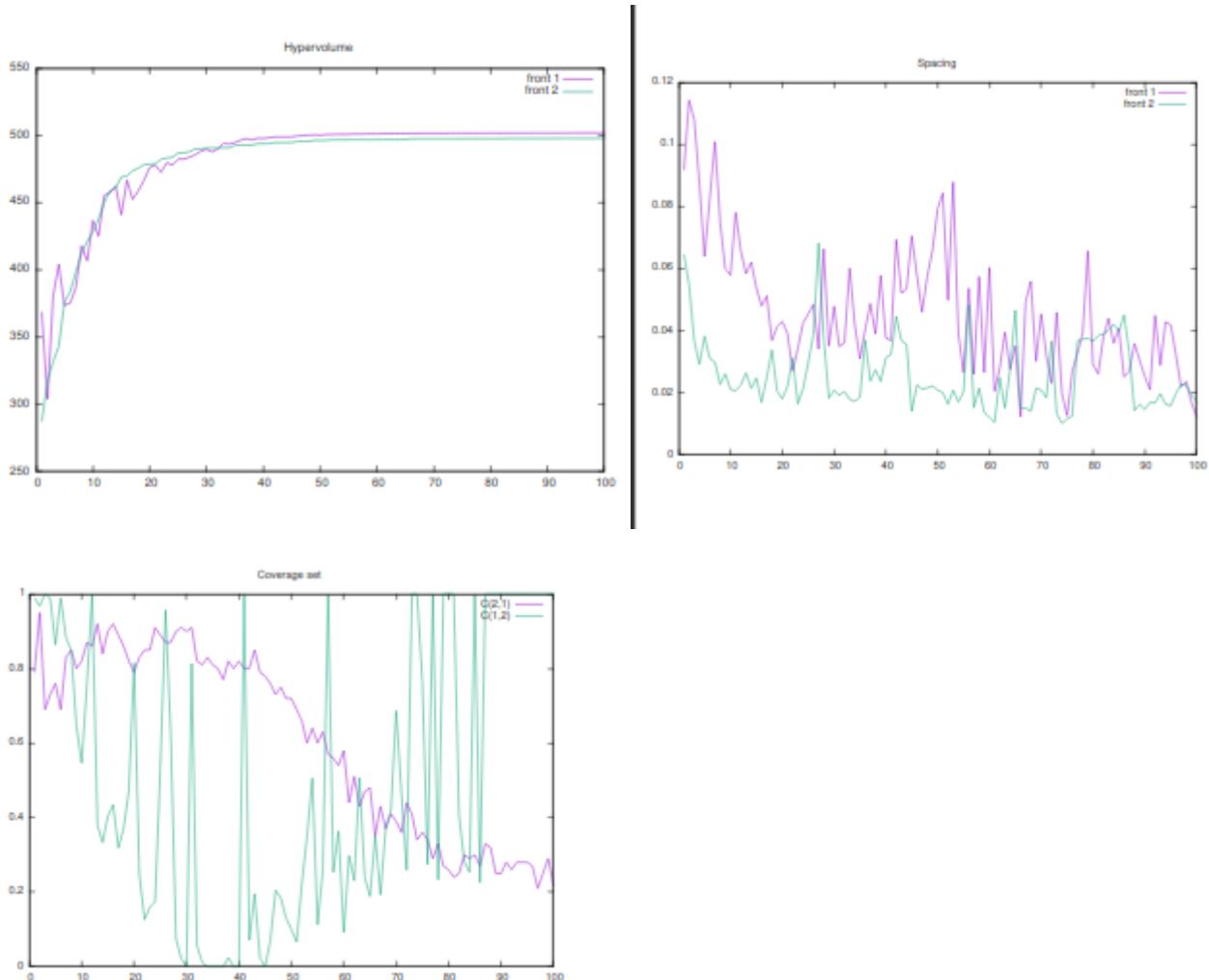
Con más prestaciones (10K) los resultados mejoran gratamente. Si observamos la comparación del coverage set del frente 2 sobre el 1, en ambos casos parece tener un bajo valor, lo que puede indicar que los resultados del frente 1 son más prometedores que los del 2.





16 DIMENSIONES

En este caso, es más interesante el estudio de las métricas en un caso con un gran número de generaciones que de individuos, al ser este un problema más complejo los resultados no convergen con tanta precisión como en el caso anterior.

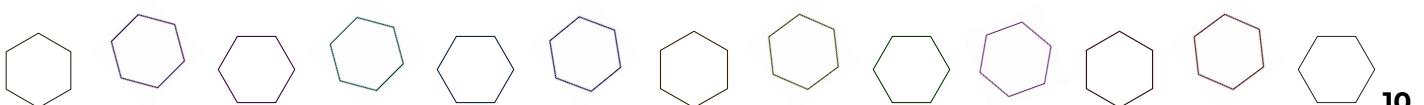


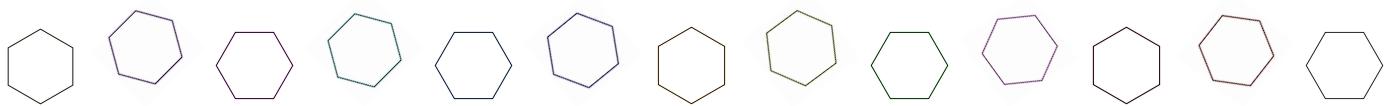
Siendo el frente 1 sobre el 2, la solución propuesta por el alumno sobre la del dataset, observamos que ambos resultados muestran un espaciado y un hipervolumen bastante similar, sucede algo muy parecido al caso de 4 Dimensiones.

```

Front 1 hypervolume: 93.4805092514
Front 1 spacing: 0.0123200040
Front 2 hypervolume: 91.3403685743
Front 2 spacing: 0.0171999264
C(front2,front1)=0.2200000000
C(front1,front2)=1.0000000000
  
```

Los resultados parecen cubrir mejor las soluciones, aunque las soluciones están más espaciadas.





Conclusiones

Podemos decir que los resultados no son los esperados claramente, el frente no se llega a cubrir el frente de pareto y la actualización de vecinos parece mover individuos hacia los máximos, de alguna forma que a nivel de implementación no se ha llegado descubrir, aunque generando una solución que es posible estudiar.

