Resolucion del probema de recoleccion de basura de los contenedores utilizando algoritmos evolutivos multiobjetivo

Gimena Bernadet
CI: 4.503.539-1
Curso de Algoritmos Evolutivos
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
gimebernadet@gmail.com

Cristiano Coelho
CI: 4.298.259-3
Curso de Algoritmos Evolutivos
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
cristianoocca@hotmail.com

Abstracto—**Este documento propone la resolución del problema histórico de la recolección de basura en Montevideo. Se utilizará un algoritmo evolutivo multi objetivo para obtener un conjunto de soluciones satisfactorias, y se evaluará el mismo comparando con otros métodos de resolución**

I. DESCRTIPCION DEL PROBLEMA

La propuesta se trata de utilizar la idea del ahora Intendente de Montevideo, Ing. Daniel Martínez, de colocar sensores en los contenedores de basura para saber qué tan llenos están, y de esta manera organizar los trayectos de los recolectores de manera de optimizarlos.
Además de los datos de los sensores, se puede estimar la velocidad de llenado de un contenedor, y hacer una planificación de camiones recolectores acorde utilizando ambos datos, para minimizar la cantidad de basura que quedara en la ciudad una vez que los camiones terminen de recoger, como también minimizar la trayectoria de los camiones.
La cantidad de basura que quedara en la ciudad una vez que los camiones terminen de recoger, fue definida como una métrica de calidad de servicio (quallity of service): QoS. Además se hizo énfasis en el porcentaje de llenado de los contenedores al finalizar el recorrido (por ejemplo, se consideró peor no recolectar un contenedor que está lleno en un 80% que 4 contenedores con un 20%).

A. Consideraciones

Por un lado se tiene la información directa de los sensores, que indica cuán lleno está un contenedor (en %), este dato fue el utilizado al ejecutar cada instancia del algoritmo (la entrada de éste).

Por otro lado, se manejaron los siguientes datos estáticos (calculados una única vez, iguales para todas las instancias):

- Lista de contenedores
  - Con coordenadas e identificador
  - Velocidad de llenado de cada uno, obtenido mediante análisis de los datos de los sensores
    - La velocidad de llenado se definirá en % / h (porcentaje / hora)
  - Tiempo que se demora en recoger un contenedor (esto es global y no por contenedor)
- Matriz de distancias y tiempos de contenedores
- Cantidad de Camiones
- Capacidad de camiones, cantidad de contenedores completos que puede recolectar un camión antes de tener que descargar.
- Función de calidad de servicio (QoS) que multiplica por algún factor determinado según el porcentaje de llenado del contenedor al finalizar el recorrido.
- Definimos que un camión solo hace un trayecto, o sea, que una vez que sale del origen, y llega al destino, no puede volver a salir.
- Tiempo que demora un camión en recolectar un contenedor

B. Formulación Matemática

El problema es claramente multiobjetivo. Por un lado se busca minimizar el recorrido de los camiones de recolección y por el otro maximizar la calidad del servicio brindado.
Para poder definir entonces las dos funciones objetivo se definen las siguientes constantes:

$CANT\_CONT$ = Cantidad de contenedores
$CANT\_CAM$ = Cantidad de camiones
Todas las variables anteriores deben ser > 0.

$C = \{1,2,\dots,CANT\_CONT\}$ = Lista de contenedores

$CAM = \{1,2,\dots,CANT\_CAM\}$ = Lista de camiones

D (x,y) = Función que indica la distancia entre dos puntos, x e y.

Sea $S = \{ s_{11}, s_{12}, \dots , s_{21}, s_{22}, \dots, s_{ij}, \dots \}$ el vector solución, de largo N donde $s_{ij} \in C \cup \{0\}, i \in CAM, j = 1..M, M \leq CANT\_CONT$, esto es, s_{ij} indica el contenedor a recoger por el camión i en la posición j o volver al origen si $s_{ij} = 0$.
Este vector se puede ver como una matriz donde cada uno indica la planificación de cada camión, indicando que contenedores debe recoger y en qué orden.

Dado lo anterior, definimos la primera función objetiva:

$$F1\ (S) = min \sum_{i=1}^{CANT\_CAM} \left(D(0, s_{i1}) \sum_{j=2}^{M} \left(D(s_{i\ j-1}, s_{ij})\right) + D(s_{i\ M}, 0) \right)$$

Lo que quiere decir, que se intenta minimizar la trayectoria total de los camiones.
Esto es, por cada camión, se suma la distancia entre el origen (0) y el primer contenedor visitado, luego todas las distancias intermedias entre contenedor y contenedor, y finalmente la distancia entre el último contenedor y el origen nuevamente, para sumar la distancia requerida para volver.

El cálculo de la segunda función objetivo, para medir la calidad de servicio, es bastante compleja ya que para poder calcularla, necesitamos saber el tiempo de recolección de cada contenedor.

Para esto, definimos lo siguiente:

Se define una función P(x) que para un contenedor, calcula el % de llenado (esto es la basura que quedo en el contenedor una vez finalizado el proceso de recolección) y retorna el puntaje asociado a este (más adelante se define la escala de puntajes que se utilizará en la implementación).

Finalmente, se define F2 de la forma:

$$F2\ (S) = min \sum_{i=1}^{CANT\_CONT} (\ P(i))$$

Es decir que por cada contenedor se suma un valor de calidad que se relaciona directamente con el porcentaje de llenado que tenga cuando se termina la recolección.

Finalmente, se tienen ciertas restricciones del vector solución:

- La cantidad de basura recogida por cada camión no puede exceder su capacidad real, para esto se utiliza el % de basura recogido de cada contenedor y se valida que el total no supere su capacidad.
- Para cada camión, luego que se tiene un 0 en su solución, todos los restantes valores a la derecha también deberán ser 0, para hacer cumplir la restricción de que un camión solo puede hacer un trayecto.

II. Relevación de Datos

Los datos mencionados anteriormente y necesarios para resolver el problema fueron recolectados de diversos medios:

1. **Contenedores**: La ubicación de contenedores se encuentra disponible en la página de la IMM: (1)

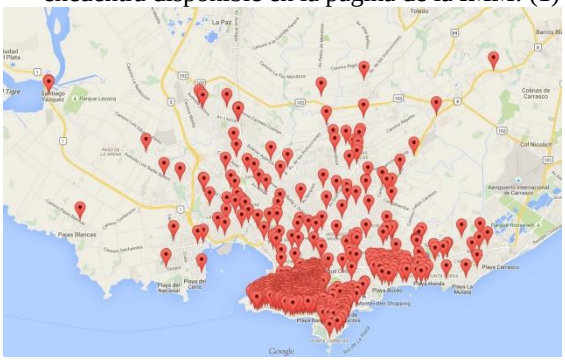


Figura 1 – Contenedores de Montevideo

Figura 2 – Contenedores del Centro y Ciudad Vieja

2. **Vertedero** (o lugar donde salen y al que regresan los camiones) (2)

Figura 3 – Vertedero de donde salen/llegan los camiones recolectores en Camino Felipe Cardoso (vista desde Google Maps)

3. **Distancias y tiempos entre contenedores**: Para todas las combinaciones entre contenedores, fue necesario tener la distancia y el tiempo necesario para llegar de uno a otro. Además se incluyó el punto de origen (de donde se sale) y destino (donde se descarga). Por simplicidad, tomamos origen = destino. Estos tiempos y distancias se obtuvieron mediante el servicio de mapas de Google Maps. (3)

4. **Tiempo que demora un camión en recolectar un contenedor:** Para obtener este dato tuvimos que cronometrar a un camión en acción.

Figura 4 – Camión recolector

5. **Cantidad de camiones:** Según (4) la intendencia dispone de:
   i) 3 camiones recolectores compactadores de carga lateral
   ii) 2 camiones recolectores compactadores de carga trasera
   iii) 1 camión lava contenedores de carga lateral

   Por lo que asumimos que tenemos 6 camiones en total.

6. **Capacidad de un camión:** Además según (4) hay 850 contenedores de capacidades 2,4 m3 y 3,2 m3. Nosotros manejamos 804 contenedores de 2,4m3 0 3,2m3.
   Además según (5)la densidad de residuos sólidos sin compactar es de: 200kg/m3 o 500kg/m3 compactados. O sea que un contenedor de 2.4m3 pesa 480kg y un contenedor de 3.2m3 pesa 640kg. En promedio: 560kgs por contenedor.
   Según la ficha técnica de los camiones Scania (6), que son los mismos utilizados en Mvdeo., la carga máxima del camión es de 26000kg. Con esto, podríamos estimar que cada camión puede recolectar unos 26000/560 contenedores completos.
   Esto es, 47 contenedores.

III. JUSTIFICACION DEL USO DE AE

Estamos ante un problema de optimización, similar al de TSP (Traveling Salesman Problem) o VRP (Vehicle Routing Problem), con varias diferencias, como por ejemplo, se tiene más de un vehículo (o más de un viajero) y además no necesariamente se deberán recoger todos los contenedores (el viajero no tiene que recorrer todos los lugares) sino solo aquellos que realmente importen.
Hasta el momento no hay forma de resolver este tipo de problema con un algoritmo exacto, y en tiempo razonable, por lo que se experimentará con el uso algoritmos evolutivos, que prometen dar muy buenos resultados si se implementan correctamente, y en tiempos razonables que no crecen exponencialmente a medida que aumentamos la cantidad de datos del problema.
Por último, los AE pueden resolver problemas multi objetivo optimizando ambas funciones al mismo tiempo, lo cual también es usualmente difícil.

IV. ESTRATEGIA DE RESOLUCION

Se utilizó la librería jMetal, para desarrollar un algoritmo evolutivo multiobjetivo que resuelva el problema planteado.

A. Representacion de la solución

Lista de enteros, como se muestra en la Figura 5. Tal que se divide en sub listas, donde cada una representa los contenedores que debe visitar el camión (y en ese orden). Además se tiene la posición especial “0” para indicar que debe volver al origen/destino.

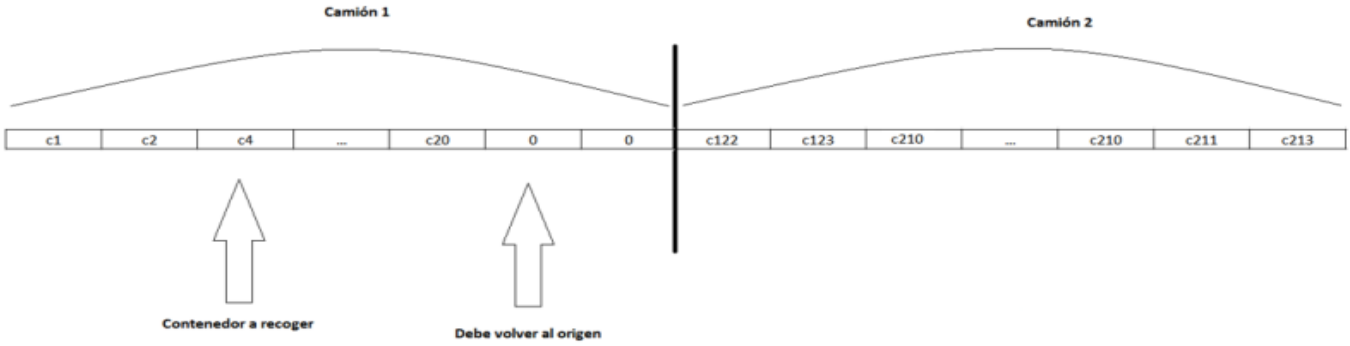


Figura 5 – Representación de la solución

Luego, el largo de cada sub lista va a estar dado por el valor de $\boldsymbol{Capacidad\ de\ Camiones} + (\boldsymbol{Capacidad\ de\ camiones} / \mathbf{2})$
Debido a que un contenedor no necesariamente va a estar completamente lleno al momento de ser recogido, de esta forma permitimos un cierto margen de error, ya que si no la única forma en que un camión vuelva con su capacidad llena, es si todos los contenedores que recogió estuvieran al 100%.

Entonces, hay que detallar que un camión puede o no llenar su capacidad, y puede tener a partir de cierto punto todo 0's para indicar que vuelve al destino sin recoger todos los contenedores que la representación le permite.
Se da por implícito que siempre se sale y llega al origen/destino, por lo que no hay que utilizar 0's para indicar el inicio y fin (en caso en que todos los valores de la representación sean distinto de 0)

<u>Además, una solución es factible únicamente si:</u>

- Una vez haya un 0 en la lista de un camión, todos los siguientes valores deben ser 0, ya que no puede realizar más de un trayecto.
    - Para corregir estas soluciones no factibles, se mueven todos los valores distintos de 0 hacia la izquierda (para un camión dado) para "rellenar" los 0's que no deberían estar. En este punto la solución podría no ser factible aún y será corregida en el siguiente paso.
- Un camión no puede superar su capacidad
    - Para esto se debe validar que la suma de todos los % de llenado de los contenedores recogidos por el camión no supere su capacidad de contenedores
    - En caso de ser inválido, se procederá a poner 0's de derecha a izquierda hasta que la solución sea válida.
- No pueden haber contenedores repetidos, únicamente el cero se puede repetir. Esto se logró implementando una representación de permutaciones que permite ceros duplicados y cuyos valores pueden ser mayores al largo de la permutación.

Luego, la trayectoria total se hace simplemente recorriendo la lista y sumando distancias, como se definió en el modelo matemático. Esta es la primera función de fitness y se minimizará este valor.

Para calcular el QoS, primero se define la siguiente escala de puntajes:

- Si el porcentaje de llenado del contenedor es menor a 40 % se deja el mismo porcentaje como valor.
- Si el porcentaje es mayor a 40 pero menor que 80 se multiplica por 2.
- Finalmente si el porcentaje es mayor que 80 se multiplica por 4.

Luego por cada contenedor se obtiene el porcentaje de llenado a partir del tiempo que paso desde el tiempo inicial, hasta que llegó un camión a recogerlo multiplicado por su velocidad de llenado más la basura que tiene el contenedor inicialmente.
Este valor es el que se suma al camión para controlar que nunca recoja más de lo que su capacidad le permite.

En caso de que un contenedor no es recogido, su porcentaje de llenado se obtiene sumando su basura inicial, más el tiempo transcurrido desde el tiempo inicial, hasta que vuelve el último camión multiplicado su velocidad de llenado, y con este valor se obtiene el puntaje de calidad de servicio, según la escala definida anteriormente.

B. Inicializacion de la Población

La población se inicializa con el algoritmo Greedy mencionado más adelante y con todas las combinaciones de parámetros mencionadas para este. Además, a cada resultado del algoritmo, se lo perturba mediante mutación (dejando siempre uno sin perturbar) para aumentar la diversidad.

Se tomó esta decisión ya que el algoritmo por si solo requería una cantidad muy grande de generaciones para alcanzar resultados similares al de Greedy, debido a la complejidad del problema.

C. Operadores

1) De Cruzamiento

PMX (modificado para que funcione con nuestra representación de permutaciones con 0's duplicados) con probabilidades: 0.75, 0.8, 0.9

2) Mutación

Mutación: mezcla de Bit Flip Mutation y Swap mutation (modificado para que funcione con nuestra representación de permutaciones) con probabilidades de: 0.05, 0.1, 0.15.
Donde con una probabilidad 0.5 se realiza bit flip mutation y 0.5 swap mutation. Se utilizan dos operadores de mutación ya que solo con swap mutation, una solución no podría mutar de forma de obtener "nuevos ceros" o valores distintos a los que actualmente almacena.

3) Torneo Binario

Se utiliza torneo binario como operador de selección.

4) Asignación de fitness

NSGA-II: Utiliza Torneo Binario como selección, modificado para el algoritmo NSGA-II (además de dominancia, tiene en cuenta el valor de 'crowding distance')
SPEA-II: Este tiene un parámetro de configuración más: tamaño de archivo, el cual tomamos como tamaño de población / 2.

5) Generaciones:

Para evaluación de configuración: 1000.
Para comparación de algoritmos (evaluación experimental): 10.000 y 20.000.

6) Tamaño población:

100 y 200

D. Evaluación

Métricas disponibles que consideramos:

- Hipervolumen relativo
- Distancia generacional
- Tiempo de ejecución

V. Instancias y Plataforma de ejecución

Para las pruebas, se definieron 10 instancias, que representan la lectura de los sensores en un momento dado, donde los datos se generan aleatoriamente entre 5 y 100, representando así el % de llenado de cada contenedor. Además, globalmente se definen las velocidades de llenado de cada contenedor de forma aleatoria entre 1% y 2%, teniendo en cuenta que actualmente los contenedores se recogen aproximadamente cada 2 y 3 días.

Por otro lado, siempre se ejcutan de a 3 instancias al mismo tiempo en paralelo, mediante scripts que invocan al algoritmo evolutivo con los parámetros especificados que obtienen y almacenan los datos resultados para su posterior análisis.

Las pruebas fueron ejecutadas en una notebook con las siguientes especificaciones:

- CPU: Intel i7 4810MQ (Haswell) de 4 cores, con 8 cores virtuales mediante tecnología HT, y una velocidad de 2.8Ghz a 3.8Ghz mediante turbo boost. Sin embargo la velocidad de ejecución máxima es de 3.5Ghz al tener los 4 cores en uso intensivo.
- 16 GB Ram
- Disco de Estado Sólido 512GB
- Sistema Operativo Windows 8.1

VI. Evaluación de configuración

En esta etapa debemos evaluar los posibles parámetros de la configuración (probabilidad de mutación y cruzamiento, tamaño de población, generaciones, etc) para cada uno de los algoritmos elegidos. Para esto, realizamos un análisis estadístico.

1. Definimos 3 instancias de prueba.
2. Para cada instancia
    a. Para cada algoritmo
        i. Para cada combinación de parámetros.
            1. Realizamos 30 ejecuciones independientes
    b. Se estima el mejor frente de pareto luego de realizar todas las ejecuciones
    c. Se calculan métricas de RHV y GD para cada instancia y cada combinación paramétrica.
    d. Se obtienen medias y desviaciones estándar para cada métrica
    e. Se realiza un test de normalidad para los datos obtenidos (Se realizaron los tests de Kolmogorov Smirnov y Shapiro Wilks)
    f. Se analizan los resultados mediante tablas
    g. Nos quedamos con la mejor combinación paramétrica de cada algoritmo.

A. Resultados

Nota: Ver tablas en apéndice.

Para definir que combinación paramétrica es mejor, ya que se tiene una gran cantidad de posibles combinaciones, se optó por quedarnos con aquellas combinaciones que presenten mejor media en las métricas de Hipervolumen Relativo y Distancia Generacional, donde si ambas métricas coinciden en la misma combinación, se elige esa, y si estas no coinciden, pero presentan

distribución normal, se "desempata" mediante el criterio informal: `|med(A) – med(B)| > max (std(A), std(B))`. En caso de que no se presente distribución normal, se elige la que tenga mejor media en Hipervolumen Relativo, ya que a nuestro criterio es la métrica más importante, por evaluar varios aspectos de la solución en una misma métrica.
Además, si al menos un test de normalidad (de los dos elegidos) falla, se asume que no hay distribución normal.

1) Resultados para NSGA-II:

En las 3 instancias, todos los candidatos a mejor combinación (aquellos con mejor medias en ambas métricas) no presentan evidencia suficiente para demostrar que no tienen una distribución normal (no se puede rechazar la hipótesis nula), por lo que se asume normalidad. Además, en 2 instancias, se coincide en cuál es la mejor combinación, mientras que en la instancia 1, hay dos posibles combinaciones, donde no se puede desempatar mediante el criterio informal, por lo que se toma la combinación que da mejores resultados en RHV, que a su vez coincide con la misma combinación que en las otras dos instancias.

2) Resultados para SPEA-II:

Para este algoritmo, en la mayoría de los casos se obtuvieron distribuciones NO normales, donde además fue un poco más difícil elegir la mejor combinación ya que no se coincide en que combinación es mejor como en el caso de NSGA-II. Sin embargo, entre las 3 instancias, se coincide en una combinación paramétrica, y se elige esa como la mejor.

Para ambos algoritmos, NSGA-II y SPEA-II, la mejor combinación paramétrica fue la siguiente:
Tamaño de población: 200
Probabilidad de cruzamiento: 0.8
Probabilidad de mutación: 0.15

Por otro lado, NSGA-II presentó resultados mejores (menores) en tiempos de ejecución en todas las instancias, por lo que esto ya es un punto negativo para SPEA-II.

VII. EVALUACIÓN EXPERIMENTAL

En esta evaluación, se comparan ambos algoritmos con su mejor combinación paramétrica (la obtenida en la parte anterior) en dos etapas:

En una primera etapa, se utilizan 10.000 generaciones y 3 instancias distintas a la de la parte anterior, y se realiza lo siguiente:

- Se procede de la misma forma que en la parte anterior, realizando 30 ejecuciones por cada instancia y cada algoritmo, para luego estimar el mejor frente de Pareto y obtener métricas, medias y desviaciones estándar.
- Se realizan tests de normalidad (Kolmogorov Smirnov y Shapiro Wilks)
- Se realiza un test de significancia estadística para comparar las medias de las métricas de ambos algoritmos.
    - Se utiliza el test T-Student
- Nos quedamos con el algoritmo que tenga mejores medias (si pasa el test de student) en más métricas y más instancias.

En una segunda etapa, se repite lo definido anteriormente, pero con otras 3 instancias y además 20.000 generaciones.

Finalmente, se muestran de forma gráfica los mejores frentes de Pareto obtenidos en esta última etapa de cada algoritmo, para las 3 instancias y además se muestran los resultados obtenidos mediante un algoritmo Greedy que se define a continuación:

La estrategia del algoritmo será obtener para cada camión, el contenedor más cercano, y además, solo se lo recogerá si tiene una cantidad de basura mayor a un porcentaje definido (parámetro del algoritmo). Luego se procede al siguiente contenedor más cercano de la misma forma, y así hasta llenar la capacidad del camión. Se procede de la misma forma para los camiones restantes.
El algoritmo se ejecutará con el parámetro de llenado mínimo con varios valores distintos: 0%, 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45% y 50%.

A. Resultados

Nota: Ver tablas en apéndice.

En ambas etapas, los resultados obtenidos fueron similares. La mayoría de las distribuciones se pueden asumir normales ya que no se puede rechazar la hipótesis nula en ninguno de los dos tests paramétricos.

Por otro lado, en todos los casos (donde hubo distribución normal), a excepción de uno, el test de Student demostró diferencias significativas en las medias de todas las métricas.

Luego, el algoritmo NSGA-II obtuvo mejores medias en todas las métricas (confirmado en la mayoría de los casos con el test de student), como también mejores tiempos de ejecución (con SPEA-II demorando unas 3 veces más).

Como en la mayoría de los casos se obtuvieron distribuciones normales y diferencias significativas en las medias, y estas fueron mejores para NSGA-II, **se decide que este algoritmo es el que obtuvo mejores resultados.**

Luego, por cada instancia, comparamos gráficamente los mejores frentes de ambos algoritmos, y además los resultados del algoritmo Greedy.

En el eje X se tiene la distancia total recorrida, mientras que en el eje Y se tiene el valor de calidad de servicio, donde para ambos casos, cuanto menor, mejor.

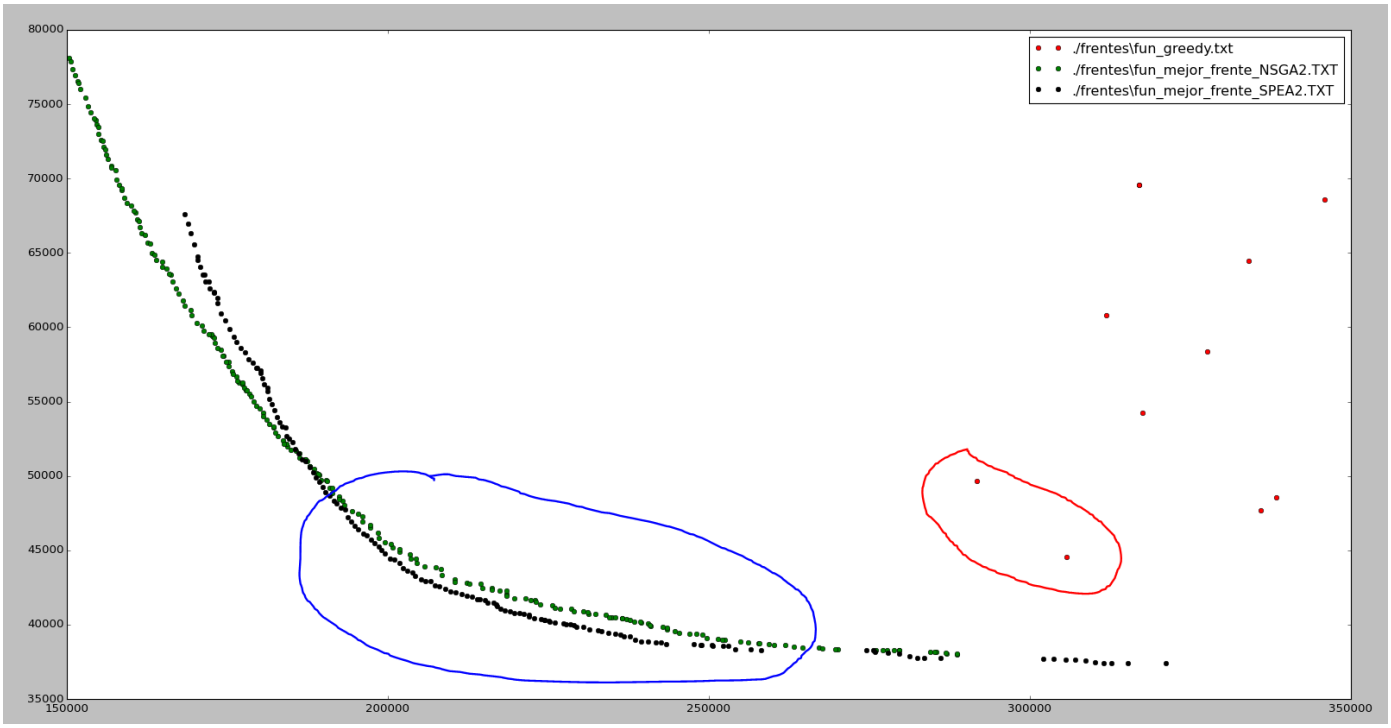
1) *Instancia 7*

Resaltado en rojo, se tiene el "frente" obtenido por el algoritmo Greedy.
Se puede observar que NSGA-II devuelve mejores resultados en una parte del frente, pero sin embargo, levemente peores en otra parte del frente (resaltado en azul). Se podría decir que la parte resaltada en azul es la parte más importante del frente, ya que los extremos no son realmente importantes porque queremos balancear ambas funciones objetivos.

La diferencia máxima para el eje Y en la zona resaltada con azul es de un 2% aproximadamente, mientras que para el eje X es de para un mismo valor de Y, es de un 5% aprox.

Con esto las conclusiones de que NSGA-II es mejor que SPEA-II (según métricas obtenidas) para este problema podrían no ser del todo válidas, ya que el motivo de que NSGA-II siempre de un mejor valor en RHV, puede ser por el extremo de más a la izquierda, el cuál SPEA-II nunca llega del todo, que puede o no ser de importancia.
De todas formas, solo estamos viendo el mejor frente obtenido por cada algoritmo, y no el caso promedio, el cuál puede ser distinto.

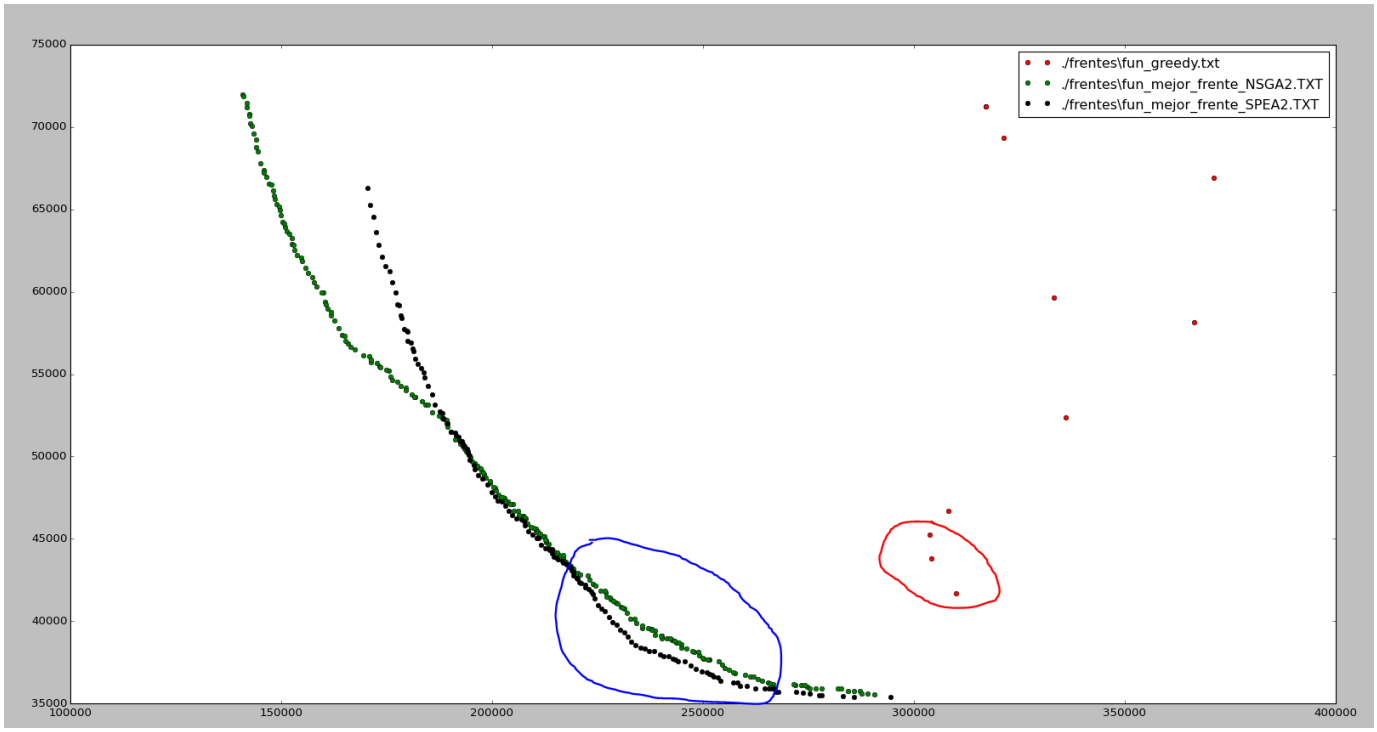
Luego, en comparación con Greedy, las mejoras son evidentes.
Para el eje Y, se tiene una mejora de un 16% con el mismo valor de X. Mientras que para un mismo valor de Y, la mejora respecto al eje X es de 34%. Esto quiere decir, que se puede obtener la misma calidad de servicio, recorriendo un 34% menos de distancia, lo cual anualmente se puede traducir a un ahorro de combustible (y dinero) considerable.



2) *Instancia 8*

Nuevamente, en esta instancia se tiene el mismo caso que en la instancia anterior, donde NSGA-II solo es mejor por el extremo izquierdo del frente, sin embargo en la zona "Más importante", SPEA-II es superior.

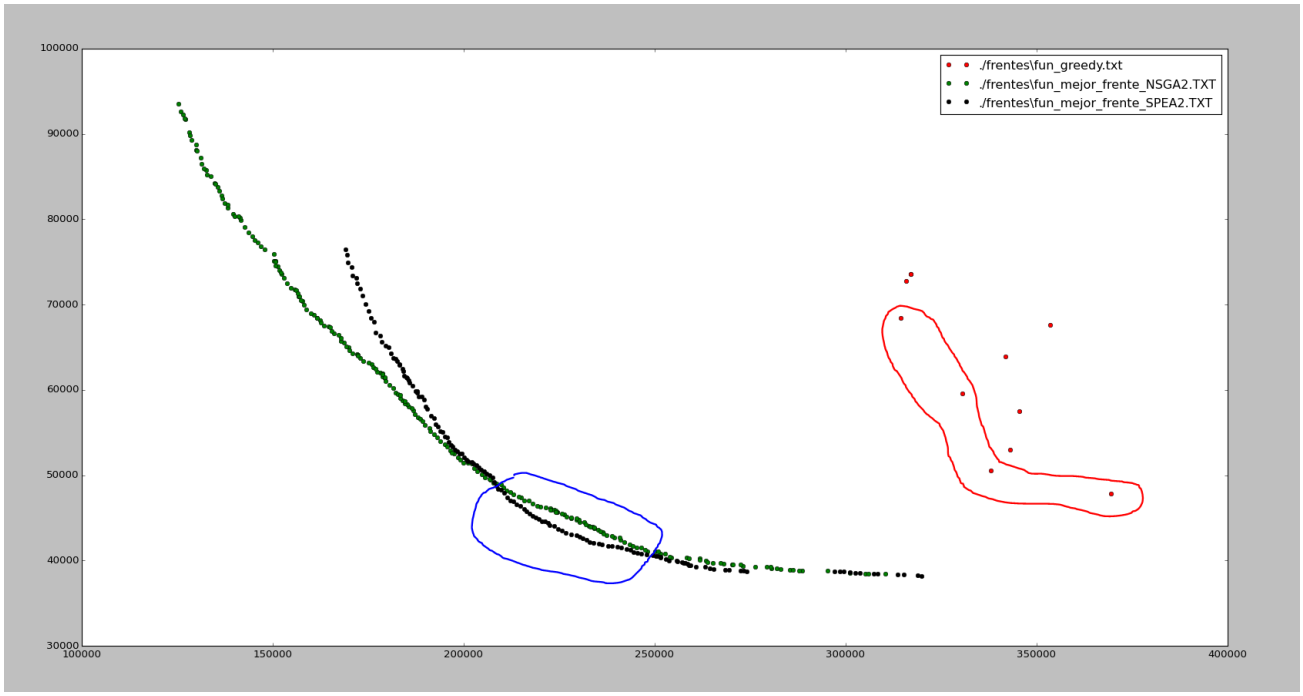
En este caso, la mejora respecto a Greedy en el eje Y es de un 18%, tomando el extremo derecho.
Luego, la mejora respecto al eje X, para un mismo valor de Y, es de un 27-29% (según algoritmo)



3) Instancia 9

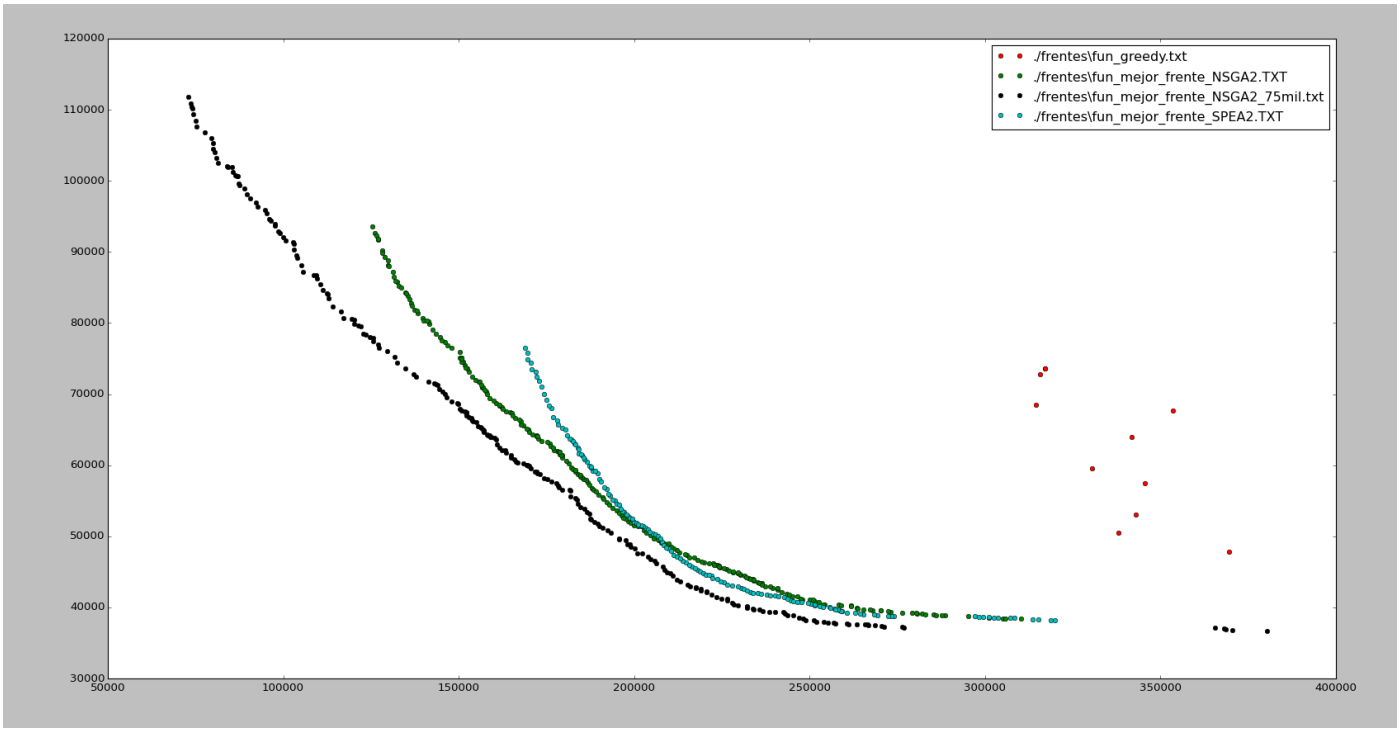
En esta instancia también sucede lo mismo al comparar ambos algoritmos.

Luego, la mejora respecto a Greedy respecto al eje Y, es de un 21% (además de una diferencia significativa también en el valor X), mientras que respecto al eje X para un mismo valor de Y, es de 38-39%.



Se realizó una tercera prueba, aumentando la cantidad de generaciones del algoritmo NSGA-II, de forma de que necesite el mismo tiempo de ejecución que SPEA-II, para intentar comparar si logra mejores resultados en todo el frente de Pareto. Para esto, se utilizó la misma instancia 9 y se realizaron 30 ejecuciones independientes, con 75000 generaciones.

Se obtuvo un promedio de 343s (comparado con los 345s de SPEA-II con 20000 generaciones), y graficando los resultados, se puede observar que el mejor frente con estos parámetros de NSGA-II (puntos en negro) es superior en todas los puntos del frente.



Está claro que esto es solo el "mejor caso" del algoritmo, pero como ya se demostró que NSGA-II es mejor que SPEA-II mediante un análisis estadístico, esto solo hace más fuerte esta afirmación.

Por lo que esto concluye que NSGA-II es realmente superior a SPEA-II en este problema.

VIII. Conclusiones y Trabajo Futuro

Con los análisis estadísticos, se puede concluir que el algoritmo NSGA-II para el problema planteado es superior a SPEA-II.

Por otro lado, NSGA-II es significativamente mejor en tiempos de ejecución, por lo que se demostró que si se aumenta la cantidad de generaciones de forma de igualar los tiempos de ejecución de SPEA-II, se obtienen mejores resultados en todo el frente.
Otra alternativa, podría ser combinar ambos algoritmos de forma de obtener lo mejor de cada uno.

Como trabajo a futuro, se podría mejorar el tiempo de ejecución del AE paralelizando mediante un sistema de islas o alguna solución similar.

Por otro lado, no hay duda de que ambos algoritmos son superiores (y por un alto margen) a los resultados obtenidos por un algoritmo Greedy, (dejando de lado tiempos de ejecución).
Estas diferencias, pueden significar tanto un ahorro importante al disminuir la distancia recorrida, como también una mejora en la calidad del servicio.
Además, los AE dan la posibilidad de elegir entre muchas soluciones posibles al problema, debido a que devuelve una parte importante del frente de Pareto, mientras que Greedy se limita a una solución (o varias) particular.

IX. REFERENCIAS

1. **[En línea] https://catalogodatos.gub.uy/dataset/contenedores-residuos-secos-domicialiarios.**

2. [En línea] https://www.google.com.uy/maps/place/Vertedero+Felipe+Cardoso/@-34.8560997,-56.1009707,15z/data=!4m2!3m1!1s0x0:0x623564b06e603efb.

3. [En línea] https://developers.google.com/maps/documentation/javascript/.

4. [En línea] http://www.teyma.com/web/es/teymamedioambiente/servicios/higiene/index.html.

5. [En línea] http://www.bvsde.paho.org/eswww/proyecto/repidisc/publica/hdt/hdt017.html.

6. [En línea] http://www.scania.es/Images/DB6x2b4M_091_tcm64-161645.pdf.

APÉNDICE

Configuración paramétrica:

Instancia 1

NSGA-II

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	100	1000	0.75	0.05	2,8	0,66417987	0,041173056	0,8863099	0,507654548	0,0115154	0,00197937	0,79546387	0,65682065
NSGA2	100	1000	0.75	0.1	3	0,72686171	0,028440703	0,551055	0,117450029	0,0091315	0,00119691	0,67570763	0,05641271
NSGA2	100	1000	0.75	0.15	3,03333333	0,76185243	0,024734416	0,67361083	0,268210322	0,00887537	0,00215115	0,25267773	0,00018032
NSGA2	100	1000	0.8	0.05	3	0,66657975	0,039731305	0,99075647	0,586500466	0,01124471	0,00304583	0,24240146	0,00066616
NSGA2	100	1000	0.8	0.1	3	0,73286785	0,041740027	0,94454481	0,577034593	0,00892566	0,0018433	0,28540913	0,00943997
NSGA2	100	1000	0.8	0.15	3	0,76977775	0,032641731	0,91546115	0,679165542	0,00830495	0,00199488	0,92566701	0,06592272
NSGA2	100	1000	0.9	0.05	3	0,66952019	0,038548021	0,20207939	0,223902062	0,01125364	0,0028371	0,05422907	1,4468E-05
NSGA2	100	1000	0.9	0.1	3	0,71435183	0,036782305	0,94274607	0,670429528	0,01008915	0,0024919	0,19518351	0,00627771
NSGA2	100	1000	0.9	0.15	3,36666667	0,76093877	0,050686963	0,80205741	0,344600886	0,00840596	0,00246013	0,31709419	0,00265523
NSGA2	200	1000	0.75	0.05	6,33333333	0,75080856	0,034521897	0,70804246	0,204349414	0,00525432	0,00117769	0,33677967	0,00029603
NSGA2	200	1000	0.75	0.1	6,7	0,79793067	0,029640241	0,89208743	0,560152054	0,00466954	0,00112491	0,99720945	0,98458093
NSGA2	200	1000	0.75	0.15	6,9	0,83908808	0,026461366	0,90655755	0,070304573	0,00349481	0,00082305	0,88163437	0,1386795
NSGA2	200	1000	0.8	0.05	6,86666667	0,74200502	0,031529915	0,88637191	0,699690759	0,00576001	0,00129525	0,54955035	0,00190327
NSGA2	200	1000	0.8	0.1	6,96666667	0,80574667	0,041342619	0,9491993	0,701783657	0,00486148	0,00158786	0,33174325	0,02745166
NSGA2	200	1000	0.8	0.15	7,13333333	0,84740393	0,029776699	0,90743768	0,69616431	0,00357924	0,00088515	0,95405368	0,60074663
NSGA2	200	1000	0.9	0.05	7,56666667	0,75106564	0,038746288	0,97276162	0,888200641	0,00593104	0,0017691	0,16981783	0,0072468
NSGA2	200	1000	0.9	0.1	6,96666667	0,79661717	0,035593117	0,53967168	0,243951023	0,00498062	0,00148101	0,29028531	0,0920623
NSGA2	200	1000	0.9	0.15	6,9	0,82266254	0,041153764	0,31754394	0,231590793	0,00414791	0,00148623	0,98540363	0,30650479

SPEA-II

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
SPEA2	100	1000	0.75	0.05	6,06666667	0,62477342	0,046926978	0,71159024	0,22302179	0,01518006	0,00216242	0,99050158	0,51893961
SPEA2	100	1000	0.75	0.1	6,06666667	0,6856887	0,040952989	0,89276321	0,903857291	0,01326378	0,00332958	0,17833694	1,9932E-06
SPEA2	100	1000	0.75	0.15	5,63333333	0,71393581	0,04533764	0,98667705	0,822703123	0,01157457	0,00224982	0,33427723	0,0349284
SPEA2	100	1000	0.8	0.05	6,2	0,63931951	0,046590876	0,76438734	0,086299337	0,0160831	0,00353494	0,1900702	7,6326E-05
SPEA2	100	1000	0.8	0.1	5,9	0,68260731	0,035656451	0,59466832	0,708553255	0,01319473	0,00251372	0,32976256	0,00258003
SPEA2	100	1000	0.8	0.15	6,03333333	0,70720999	0,041085731	0,83361909	0,252747566	0,01243065	0,0020873	0,81107668	0,12034315
SPEA2	100	1000	0.9	0.05	5,83333333	0,62113364	0,053985819	0,99998091	0,991883099	0,01528551	0,00270803	0,98940461	0,21197227
SPEA2	100	1000	0.9	0.1	5,36666667	0,67289025	0,048762272	0,94519373	0,348925918	0,01360025	0,00197773	0,94657383	0,64954418
SPEA2	100	1000	0.9	0.15	5,06666667	0,71772514	0,043014479	0,99356775	0,476319194	0,01273866	0,00341948	0,65398439	0,00360379
SPEA2	200	1000	0.75	0.05	27,9	0,73404549	0,045525258	0,40756235	0,126821533	0,00758351	0,00158704	0,97621847	0,53436381
SPEA2	200	1000	0.75	0.1	24,2	0,78476357	0,040830592	0,7181702	0,050554797	0,00562949	0,00156802	0,94044397	0,33825007
SPEA2	200	1000	0.75	0.15	22,0666667	0,80244115	0,045842516	0,19453528	0,009081055	0,00541119	0,00133507	0,98963281	0,50236613
SPEA2	200	1000	0.8	0.05	26,2333333	0,71967215	0,034939336	0,97912659	0,884220123	0,00802965	0,0020534	0,05179117	1,088E-06
SPEA2	200	1000	0.8	0.1	22,4	0,77577416	0,035082105	0,97965519	0,886555552	0,0060429	0,00159721	0,23785055	5,9333E-05
SPEA2	200	1000	0.8	0.15	21,1	0,81231981	0,030380158	0,98889447	0,51762408	0,00528082	0,0016406	0,06606449	9,6963E-06
SPEA2	200	1000	0.9	0.05	22,5666667	0,73238713	0,032933528	0,73531722	0,385360897	0,00794937	0,00140561	0,64789078	0,06654569
SPEA2	200	1000	0.9	0.1	20,5333333	0,76159653	0,041804243	0,85159227	0,213292673	0,00634385	0,0016249	0,97155359	0,8547352
SPEA2	200	1000	0.9	0.15	18,2666667	0,81044758	0,037287915	0,8202819	0,572023273	0,00547038	0,00125929	0,78897657	0,1193445

Instancia 2

NSGA-II

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	100	1000	0.75	0.05	2,83333333	0,61461908	0,03566003	0,85132416	0,094011329	0,01647807	0,00155789	0,47153319	0,03318059
NSGA2	100	1000	0.75	0.1	3,06666667	0,67087168	0,03683268	0,98478619	0,735465527	0,01467612	0,00182819	0,66911943	0,43661234
NSGA2	100	1000	0.75	0.15	2,96666667	0,69847663	0,03220855	0,96629635	0,437041342	0,01377838	0,00178373	0,97574037	0,84125406
NSGA2	100	1000	0.8	0.05	2,96666667	0,61492087	0,04250899	0,62865649	0,019794084	0,01676054	0,00203221	0,43972219	0,0427321
NSGA2	100	1000	0.8	0.1	3	0,66856802	0,02527203	0,84178453	0,360598713	0,01484772	0,00132028	0,76088269	0,27567595
NSGA2	100	1000	0.8	0.15	3	0,70640114	0,03765499	0,91323824	0,565929532	0,01317852	0,00180507	0,99933795	0,82394385
NSGA2	100	1000	0.9	0.05	3	0,61018507	0,03165929	0,58944817	0,025160946	0,01672637	0,00157542	0,54444516	0,37203604
NSGA2	100	1000	0.9	0.1	3	0,65427232	0,02800451	0,9197522	0,566923082	0,01523866	0,00153203	0,97165142	0,11862136
NSGA2	100	1000	0.9	0.15	3,33333333	0,68780092	0,02100253	0,87660968	0,615156054	0,0140354	0,00123905	0,91053938	0,55963206
NSGA2	200	1000	0.75	0.05	6,7	0,6962325	0,03002418	0,96186625	0,877307475	0,00900958	0,00118933	0,994767	0,94960254
NSGA2	200	1000	0.75	0.1	6,73333333	0,75969957	0,03320626	0,41249475	0,199505687	0,00717991	0,00114541	0,98183503	0,43374461
NSGA2	200	1000	0.75	0.15	6,93333333	0,79275724	0,03142603	0,64490278	0,028158398	0,00648559	0,00097245	0,86049352	0,65287554
NSGA2	200	1000	0.8	0.05	6,9	0,68766781	0,03653074	0,93231444	0,453097671	0,00954936	0,00130078	0,99822812	0,93907863
NSGA2	200	1000	0.8	0.1	7,03333333	0,76200843	0,03371102	0,95482879	0,978882134	0,00722714	0,00145679	0,81522042	0,28959605
NSGA2	200	1000	0.8	0.15	7,16666667	0,80462548	0,02556597	0,98654324	0,570885599	0,00593958	0,00091196	0,76364008	0,58974487
NSGA2	200	1000	0.9	0.05	7,36666667	0,69188631	0,03340977	0,94744548	0,462266266	0,00909253	0,00119133	0,9828548	0,30044681
NSGA2	200	1000	0.9	0.1	6,93333333	0,74650219	0,03442317	0,81737952	0,53349179	0,00755291	0,00140641	0,90320676	0,51921028
NSGA2	200	1000	0.9	0.15	6,83333333	0,77703425	0,02712573	0,55675057	0,260020703	0,0068683	0,00088461	0,97332586	0,51953715

SPEA-II

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
SPEA2	100	1000	0.75	0.05	5,96666667	0,58125056	0,02627141	0,25220985	0,106514163	0,02420265	0,00213664	0,54676968	0,22184183
SPEA2	100	1000	0.75	0.1	6,16666667	0,62759666	0,02449362	0,96386467	0,632733285	0,02198871	0,00149578	0,89322885	0,2124164
SPEA2	100	1000	0.75	0.15	5,63333333	0,66235946	0,02820322	0,80292771	0,042988453	0,02019765	0,00252037	0,94050376	0,48373097
SPEA2	100	1000	0.8	0.05	5,86666667	0,5758183	0,02690516	0,95258943	0,361037642	0,02441004	0,0019086	0,84697447	0,66099066
SPEA2	100	1000	0.8	0.1	6	0,6169365	0,02348722	0,24396331	0,012377977	0,02207825	0,00158354	0,98831932	0,39018622
SPEA2	100	1000	0.8	0.15	5,73333333	0,65151846	0,02428471	0,84604064	0,262100786	0,02056473	0,00214492	0,86300691	0,74877429
SPEA2	100	1000	0.9	0.05	5,66666667	0,57156188	0,02978189	0,4267988	0,232801929	0,02423431	0,00217571	0,99431825	0,57142019
SPEA2	100	1000	0.9	0.1	5,33333333	0,61165608	0,02579357	0,63484433	0,019504145	0,02276238	0,00176586	0,57883897	0,002981
SPEA2	100	1000	0.9	0.15	5,03333333	0,64453705	0,03028402	0,98459679	0,861882687	0,0203231	0,00356564	0,12409452	5,1143E-06
SPEA2	200	1000	0.75	0.05	26,5	0,67217859	0,02795944	0,99910195	0,933349907	0,01290875	0,00172263	0,91001173	0,68420726
SPEA2	200	1000	0.75	0.1	23,1333333	0,72517919	0,02525643	0,39458763	0,262371093	0,0112224	0,00149198	0,78921802	0,26114357
SPEA2	200	1000	0.75	0.15	21,5333333	0,76428656	0,02105556	0,87404964	0,427976549	0,00973361	0,00115589	0,36343471	0,04440189
SPEA2	200	1000	0.8	0.05	24,9666667	0,67261979	0,02990195	0,9532324	0,594798982	0,01304232	0,00139145	0,90457868	0,18232568
SPEA2	200	1000	0.8	0.1	21,8666667	0,72066458	0,04020849	0,9956582	0,940663755	0,01162757	0,00182442	0,97132733	0,27210391
SPEA2	200	1000	0.8	0.15	20,8333333	0,75928782	0,04293771	0,67726289	0,105809823	0,00942484	0,00217895	0,82519761	0,30180973
SPEA2	200	1000	0.9	0.05	21,4666667	0,65950726	0,03135604	0,87374047	0,782253385	0,01351124	0,00188185	0,68507665	0,55073267
SPEA2	200	1000	0.9	0.1	20,5	0,70895792	0,03681151	0,69826145	0,000980765	0,0115886	0,002211	0,79309202	0,04720486
SPEA2	200	1000	0.9	0.15	19,3666667	0,74397299	0,02739161	0,9722836	0,599811256	0,01022982	0,0012469	0,98735962	0,97219574

Instancia 3

NSGA-II

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpv	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	100	1000	0.75	0.05	2,13333333	0,5139459	0,13439448	0,03357463	0,00085955	0,02421968	0,00900441	0,02240957	0,0004007
NSGA2	100	1000	0.75	0.1	2,43333333	0,58962404	0,11067827	0,07233671	0,00365399	0,02189067	0,00743475	0,036454	0,00032611
NSGA2	100	1000	0.75	0.15	2,3	0,65378979	0,10160984	0,0388384	0,00055006	0,01710708	0,00661934	0,00862475	4,4401E-05
NSGA2	100	1000	0.8	0.05	2,83333333	0,47347425	0,13629339	0,20203147	0,0128175	0,02825158	0,00970835	0,20775204	0,00779798
NSGA2	100	1000	0.8	0.1	2,76666667	0,58611758	0,09990775	0,15883236	0,00563412	0,02170623	0,00702683	0,05274131	0,00201689
NSGA2	100	1000	0.8	0.15	2,73333333	0,64894361	0,09425506	0,33042525	0,02050357	0,01697488	0,00622031	0,02940184	0,00010488
NSGA2	100	1000	0.9	0.05	2,76666667	0,55418337	0,11989929	0,06656957	0,00015913	0,02252053	0,00730827	0,05378457	7,0377E-05
NSGA2	100	1000	0.9	0.1	2,4	0,64767782	0,07327351	0,02054358	1,5659E-05	0,01670609	0,00479628	0,03350136	3,5968E-06
NSGA2	100	1000	0.9	0.15	2,9	0,6479734	0,09710769	0,14284163	0,00292124	0,0156101	0,00554876	0,02497269	1,6023E-06
NSGA2	200	1000	0.75	0.05	6,06666667	0,65985712	0,0856776	0,31040731	0,043168	0,01003462	0,00313145	0,45166041	0,00291205
NSGA2	200	1000	0.75	0.1	5,83333333	0,75558463	0,06343518	0,22028345	0,00199852	0,00792412	0,00292487	0,09268227	0,00043759
NSGA2	200	1000	0.75	0.15	5,73333333	0,79303051	0,07713804	0,05315878	0,00027402	0,00648055	0,00272272	0,0391745	8,0425E-06
NSGA2	200	1000	0.8	0.05	5,8	0,65155863	0,09255597	0,26181516	0,0602154	0,01172385	0,00461048	0,23006708	0,00431353
NSGA2	200	1000	0.8	0.1	5,86666667	0,7453423	0,08381557	0,04506252	3,591E-05	0,0077874	0,00365702	0,00260396	1,9185E-08
NSGA2	200	1000	0.8	0.15	6	0,80558035	0,04585295	0,72921008	0,37500909	0,00570811	0,0015385	0,63553354	0,45390186
NSGA2	200	1000	0.9	0.05	6,1	0,68316216	0,0731775	0,0448493	0,00019037	0,0109355	0,00320108	0,03707702	1,4657E-05
NSGA2	200	1000	0.9	0.1	6,46666667	0,72010892	0,09753608	0,23485242	0,00447804	0,00802848	0,00321408	0,01372838	6,3109E-07
NSGA2	200	1000	0.9	0.15	5,9	0,78258974	0,06714553	0,42871282	0,11127835	0,00599512	0,00146661	0,83962171	0,70357084

SPEA-II

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpv	medGD	stdGD	ksGDpval	shapiroGDpval
SPEA2	100	1000	0.75	0.05	6,43333333	0,4487311	0,11897156	0,14096744	0,00193225	0,03893332	0,01358787	0,04937846	0,00046573
SPEA2	100	1000	0.75	0.1	5,96666667	0,54101608	0,10278199	0,57791519	0,01912463	0,03081796	0,01216554	0,09688282	0,00531542
SPEA2	100	1000	0.75	0.15	5,7	0,58446562	0,10132371	0,27693403	0,00394458	0,02603808	0,00887556	0,13889402	0,00279645
SPEA2	100	1000	0.8	0.05	5,96666667	0,4767952	0,10306283	0,16972634	0,00575206	0,0363918	0,01052569	0,11059229	0,00097092
SPEA2	100	1000	0.8	0.1	5,63333333	0,53182419	0,11575574	0,58312912	0,05224784	0,03169566	0,01246764	0,16302016	0,00233728
SPEA2	100	1000	0.8	0.15	5,4	0,57188071	0,11023422	0,22901907	0,01684249	0,02655912	0,01192384	0,02729038	8,1505E-05
SPEA2	100	1000	0.9	0.05	6,2	0,45708656	0,12992529	0,2253261	0,00467956	0,03835658	0,01337744	0,29102278	0,01293046
SPEA2	100	1000	0.9	0.1	5,33333333	0,59026826	0,10483677	0,22886594	0,00284523	0,02711386	0,00876662	0,04294208	0,00041758
SPEA2	100	1000	0.9	0.15	4,96666667	0,62063537	0,08699843	0,32881887	0,00421911	0,02213716	0,00747069	0,03222825	0,00120132
SPEA2	200	1000	0.75	0.05	27,5333333	0,60688022	0,11384456	0,28084093	0,00854182	0,01795067	0,00744488	0,03177871	0,00036288
SPEA2	200	1000	0.75	0.1	23,8	0,71273558	0,09448525	0,01629456	8,8068E-05	0,01303013	0,00610254	0,03812635	0,00046984
SPEA2	200	1000	0.75	0.15	23,0666667	0,71943291	0,08127492	0,71166522	0,17707469	0,00980078	0,00447826	0,07280828	0,00067295
SPEA2	200	1000	0.8	0.05	25,5333333	0,61903273	0,09936818	0,11897307	0,00483446	0,01686187	0,00587998	0,12068805	0,00393955
SPEA2	200	1000	0.8	0.1	22,6333333	0,71482016	0,07100621	0,82048625	0,34233481	0,01162308	0,00364381	0,51762016	0,06721565
SPEA2	200	1000	0.8	0.15	21,2666667	0,73745298	0,08335271	0,02680691	0,00300406	0,00988727	0,00404397	0,01316049	1,498E-06
SPEA2	200	1000	0.9	0.05	23,6666667	0,63790539	0,07937971	0,09325861	0,00016425	0,01630747	0,00433146	0,30790141	0,00084764
SPEA2	200	1000	0.9	0.1	20,9666667	0,68150724	0,08729721	0,27523624	0,01521307	0,01329451	0,00618882	0,0454808	3,0392E-05
SPEA2	200	1000	0.9	0.15	20,2	0,72860542	0,0878647	0,49202373	0,00736466	0,0092301	0,00324123	0,63601785	0,00111531

Comparación de algoritmos, parte 1

Instancia 4

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	200	10000	0.8	0.15	45,2	0,863254373	0,03914096	0,99989262	0,997535765	0,00441429	0,00120562	0,93550672	0,863649011
SPEA2	200	10000	0.8	0.15	189,6	0,678773138	0,05320999	0,77555605	0,042163249	0,00581819	0,0019945	0,78897503	0,703694046

tstudent	tstudent
8,06317E-21	0,00215625
Alta probabilidad de que las medias son distintas	Alta probabilidad de que las medias son distintas
Ojo: La distribución de RHV de SPEA2 puede no ser normal ya que falla uno de los dos tests	Gana NSGA2 por una mejor media

Instancia 5

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	200	10000	0.8	0.15	47,23333333	0,915929676	0,0261636	0,63810638	0,128897503	0,00312059	0,000917	0,82183572	0,448540568
SPEA2	200	10000	0.8	0.15	181,5333333	0,838024005	0,02549742	0,93959502	0,469445407	0,004554598	0,00132551	0,94535656	0,603280723

tstudent	tstudent
1,4532E-16	1,44291E-05
Alta probabilidad de que las medias son distintas	Alta probabilidad de que las medias son distintas
Gana NSGA2 por una mejor media	Gana NSGA2 por una mejor media

Instancia 6

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	200	10000	0.8	0.15	49,9	0,873327655	0,03023259	0,43751055	0,160831317	0,004245768	0,00071534	0,69858293	0,074846603
SPEA2	200	10000	0.8	0.15	177,0333333	0,79159631	0,02834129	0,98926245	0,692014873	0,006536421	0,00195424	0,39143356	0,204455122

tstudent	tstudent
3,34126E-15	8,15523E-07
Alta probabilidad de que las medias son distintas	Alta probabilidad de que las medias son distintas
Gana NSGA2 por una mejor media	Gana NSGA2 por una mejor media

Comparación de algoritmos, parte 2

Instancia 7

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVpval	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	200	20000	0.8	0.15	95,2	0,927331707	0,01923336	0,37879542	0,881585181	0,00285628	0,00073178	0,93048188	0,700703442
SPEA2	200	20000	0.8	0.15	338,9666667	0,886688896	0,02946699	0,44568228	0,111678258	0,004151887	0,00144198	0,86242041	0,475901216

tstudent	tstudent
1,00158E-07	9,19255E-05
Alta probabilidad de que las medias son distintas	Alta probabilidad de que las medias son distintas
Gana NSGA2 por una mejor media	Gana NSGA2 por una mejor media

Instancia 8

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVp	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	200	20000	0.8	0.15	93,06666667	0,889244766	0,01631898	0,86636019	0,05569156	0,00465348	0,00087956	0,20131948	0,002840105
SPEA2	200	20000	0.8	0.15	359,3666667	0,837519389	0,02581058	0,99111478	0,99165243	0,00533514	0,00090769	0,87674307	0,244690388

tstudent	3,89144E-12
	Alta probabilidad de que las medias son distintas
	Gana NSGA2 por una mejor media

tstudent	0,0052015
	No puedo decir que las medias son distintas
	Ojo: La distribución de GD de NSGA2 puede no ser normal ya que falla uno de los dos tests

Instancia 9

poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVp	medGD	stdGD	ksGDpval	shapiroGDpval
200	20000	0.8	0.15	104,2	0,86312819	0,03316057	0,84600126	0,3794775	0,00436913	0,00113734	0,88240722	0,926060915
200	20000	0.8	0.15	345,6333333	0,796709223	0,02768984	0,9134265	0,3352896	0,00578849	0,00154868	0,90702977	0,44238925

tstudent	2,61652E-11
	Alta probabilidad de que las medias son distintas
	Gana NSGA2 por una mejor media

tstudent	0,00021132
	Alta probabilidad de que las medias son distintas
	Gana NSGA2 por una mejor media

Instancia 9 con 75000 generaciones para NSGA-II.

algoritmo	poblacion	gens	cross	mut	medTiempo	medRHV	stdRHV	ksRHVpval	shapiroRHVp	medGD	stdGD	ksGDpval	shapiroGDpval
NSGA2	200	75000	0.8	0.15	343,933333	0,91833846	0,02679289	0,54401044	0,42852256	0,00319467	0,00095455	0,74395429	0,251870424