Práctica 1 de Inteligencia Artificial

Búsqueda Local

Curso 2021/2022 2Q

Joan Almagro

Francesc Salar

Andreu Antúnez

Índice

Descripción del problema

Descripción del estado

Nuestros estados corresponden a cómo están distribuidos los diferentes grupos a rescatar para cada uno de los helicópteros que tenemos, es decir que helicóptero rescata a cada grupo.
Con más detalle, tenemos una lista de Centros, que a su vez contiene una lista de los helicópteros en ese centro. Ésta lista a la vez contiene otra lista de lo que llamamos Rescates, que corresponde a un trayecto completo del helicóptero en el que rescata hasta 3 grupos (sin superar el máximo de personas permitido).

Descripción estrategias estado inicial

Hemos implementado un total de tres estrategias para el estado inicial:

- *ESTUPIDO:* esta implementación asigna únicamente al Helicóptero 0 del Centro 0 todos los grupos a rescatar, pero solamente pudiendo rescatar un grupo por viaje.
- *LLENA_RESCATES:* esta estrategia, como la anterior, solamente utiliza el Helicóptero 0 del Centro 0 para el rescate de todos los grupos, a diferencia que en esta solución sí que puede llevar más de un grupo.
- *ASIGNA_CENTRO_EQUIT:* esta solución final la hicimos más eficiente, ya que se reparten equitativamente los grupos en cada Helicóptero disponible.

Hemos implementado estas tres estrategias iniciales para ver cómo evoluciona la solución final a raíz de si utilizamos una u otra. Creemos que es interesante utilizar desde una estrategia “estúpida” hasta una más eficiente para ver cómo cambia al final y si realmente tener una solución inicial más eficaz para nosotros también es lo óptimo cuando se trata con estos algoritmos.

Descripción operadores

Hemos implementado estos operadores:

- *avanzaRescate:* asigna un grupo a un Rescate anterior en la lista de Rescates del mismo helicóptero del mismo centro que tenía asignado. Éste operador ha sido desestimado por no contemplar cambios en los helicópteros asignados a los grupos.
- *cambiaHelicoptero:* asigna un grupo a otro helicóptero del mismo centro. Éste operador también ha sido desestimado por no contemplar cambios en los centros asignados a los grupos.
- *reasignaGrupos:* operador general con el que hemos desarrollado toda la práctica. Dado un grupo, dos centros y dos helicópteros, uno de cada centro respectivamente, si en nuestro estado el grupo es rescatado por el helicóptero dado del centro dado, devuelve un nuevo estado en el que éste grupo pasa a ser rescatado por el otro helicóptero del otro centro dado. Si el grupo no es rescatado por el helicóptero del centro devuelve null.

Descripción Heurísticas

Hemos implementado dos funciones heurísticas:

- *DesastresHeuristicFunction1:* devuelve el tiempo total que tardan los helicópteros en rescatar a todos los grupos, suponiendo que nada se hace en paralelo.
- *DesastresHeuristicFunction2:* devuelve el producto del tiempo total que tardan los helicópteros en rescatar a todos los grupos por el tiempo total que tardan los helicópteros en rescatar a todos los grupos de prioridad 1.

Experimentos

1. **Determinar qué conjunto de operadores da mejores resultados para una función heurística que optimice el primer criterio de calidad del problema (3.1) con un escenario en el que el número de centros de rescate es 5, hay un helicóptero en cada centro y el número de grupos a rescatar es 100. Deberéis usar el algoritmo de Hill Climbing. Escoged una de las estrategias de inicialización de entre las que proponéis. A**

partir de estos resultados deberéis fijar los operadores para el resto de experimentos.

Observación: Podemos tener conjuntos de operadores mejores que otros, respecto al tiempo de rescate y/o al tiempo de ejecución del programa.
Planteamiento: Elegimos distintos operadores usados y observamos los resultados.
Hipótesis:
Método:

- Elegimos 10 seed aleatoriamente.
- Ejecutamos un experimento con cada semilla y conjunto de operadores (avanzaRescate y cambiaHelicoptero, reasignaGrupos)
- Experimento con 100 grupos a rescatar, donde hay 5 centros de rescate con 1 helicóptero cada uno.
- Usamos el algoritmo de Hill Climbing
- Mediremos el tiempo de rescate y el tiempo de ejecución.

Réplica	avanza y cambia		reasignaGrupos	
	TRescate(min)	TEjecución(ms)	TRescate(min)	TEjecución(ms)
1	2732.42	1857	2642.64	1645
2	3306.61	1763	3143.21	1569
3	2665.36	1915	2707.84	1494
4	3103.46	1643	3015.85	1245
5	3202.70	1592	3052.83	1480
6	3090.23	1906	3174.49	1255
7	2882.64	1981	2947.70	1388
8	3025.12	1829	2991.76	1586

9	3396.02	1928	3212.92	1333
10	2598.46	1670	2631.17	1492
Media	3057.67	1843	3003.81	1486

En los experimentos hechos para ver qué operador escoger podemos ver que el tiempo de rescate cuando usamos el operador de *reasignaGrupos* es ligeramente menor, pero donde sí podemos ver diferencia es en el tiempo de ejecución, ya que el operador de *reasignaGrupos* es la combinación más eficiente de *avanzaRescate* y *cambiaHelicoptero*. Por lo tanto, utilizaremos el operador de *reasignaGrupos*.

2. **Determinar qué estrategia de generación de la solución inicial da mejores resultados para la función heurística usada en el apartado anterior, con el escenario del apartado anterior y usando el algoritmo de Hill Climbing. A partir de estos resultados deberéis fijar también la estrategia de generación de la solución inicial para el resto de experimentos.**

Observación: Uno de los estados iniciales es más eficiente que el otro.
Planteamiento: Generamos soluciones de los diferentes estados iniciales y compararemos los resultados.
Hipótesis: Un estado inicial será mejor que el otro.
Método:

- Elegimos 10 seed aleatoriamente.
- Ejecutamos un experimento con cada semilla comparando los dos estados iniciales
- Experimento con 100 grupos a rescatar, donde hay 5 centros de rescate con 1 helicóptero cada uno.
- Usamos el algoritmo de Hill Climbing
- Mediremos el tiempo de rescate y el tiempo de ejecución.

Réplica	ESTUPIDO	LLENA_RESCATES	ASIGNA_CENTRO_EQ

					UIT	
	TRescate (min)	TEjecución(ms)	TRescate (min)	TEjecución(ms)	TRescate(min)	TEjecución (ms)
1	3001.61	2508	2642.64	1645	2793.18	2402
2	3291.99	2871	3143.21	1569	3545.98	1996
3	2743.70	2932	2707.84	1494	3012.82	1835
4	3003.32	2469	3015.85	1245	3112.66	1978
5	3528.11	2482	3052.83	1480	3119.12	2406
6	3060.66	3107	3174.49	1255	3048.52	2198
7	3142.26	2714	2947.70	1388	3012.46	2059
8	3142.26	2644	2991.76	1586	3004.55	2155
9	3148.98	2512	3212.92	1333	3296.68	2081
10	2810.22	2312	2631.17	1492	2579.8	2181
Media	3101.46	2578	3003.81	1486	3030.67	2118

Como podemos observar la función de generación de estados que nos proporciona una solución final mejor tanto en el tiempo de rescate como en el tiempo de ejecución es la de LLENA_RESCATES. Por lo que nos decantamos por esta para los futuros experimentos

3. **Determinar los parámetros que dan mejor resultado para Simulated Annealing con el mismo escenario, usando la misma función heurística, los operadores y la estrategia de generación de la solución inicial escogidos en los experimentos anteriores.**

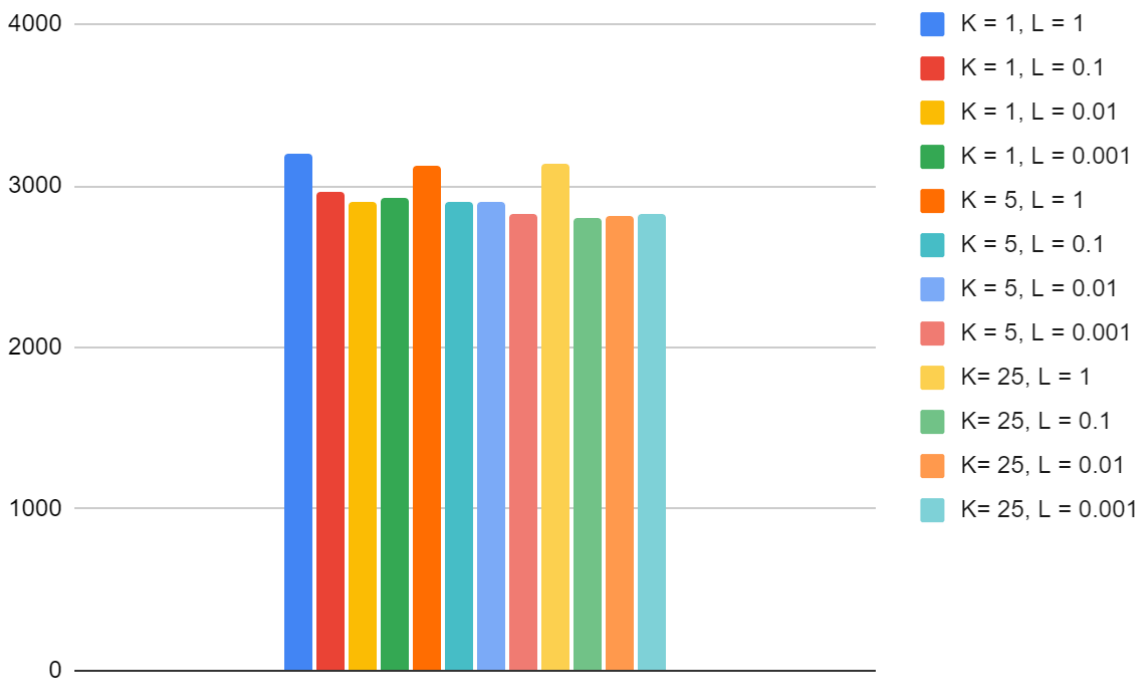
Observación: Una combinación del Simulated Annealing debería dar mejores resultados que otra.

Planteamiento: Generamos soluciones con diferentes valores de K y Lambda para ir acotando estos.

Hipótesis: Se podría encontrar una combinación que nos de valores iguales o mejores al Hill Climbing.

Método:

- Se establecerán valores para Lambda de 1, 0.1, 0.01 y 0.001.
- Se establecerán valores para K de 1, 5, 10 y 25.
- Elegiremos como valor predeterminado 10000 steps y 100 stiters, ya que es un número de iteraciones razonable.
- Elegimos 10 seed aleatoriamente.



Como vemos en la gráfica de la parte superior, los tiempos están bastante parejos , pero cogeremos la K = 25 y Lambda = 0.1 ya que son los valores que nos dan mejores resultados de media.

Usando este algoritmo con la configuración dicha anteriormente, vemos que nos da un tiempoRescate medio más bajo que con el Hill Climbing, pero también el tiempoEjecución es bastante más elevado.

4. **Dado el escenario de los apartados anteriores, estudiad cómo evoluciona el tiempo de ejecución para hallar la solución para valores crecientes de los parámetros del problema siguiendo la proporción 5:100 para centros y grupos. Comenzad con 5 centros de distribución e incrementad el número de 5 en 5 hasta que se vea la tendencia. Usad el algoritmo de Hill Climbing y Simulated Annealing y la misma función heurística que antes. Comprobad que los parámetros para Simulated Annealing que habéis hallado en el apartado anterior siguen dando buenos resultados al aumentar el tamaño del problema.**

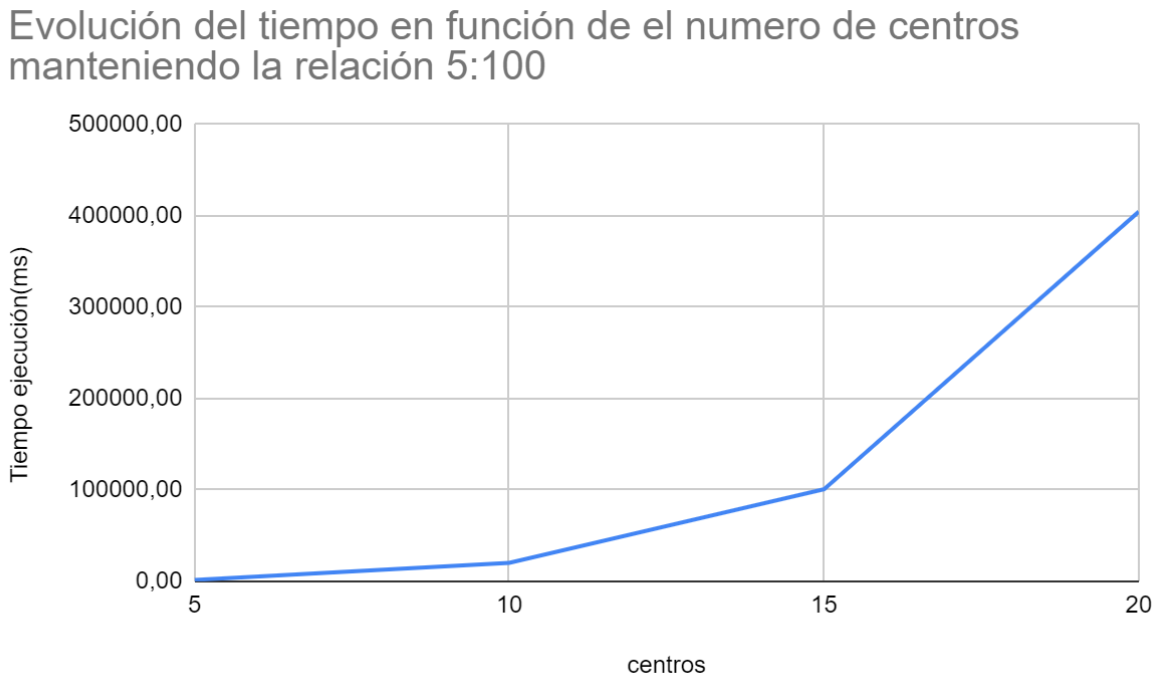
Observación: Al aumentar el número de centros y grupos manteniendo la proporción 5:100, aumentará el tiempo de ejecución

Planteamiento: Generamos soluciones para cada proporción y cada tipo de algoritmo

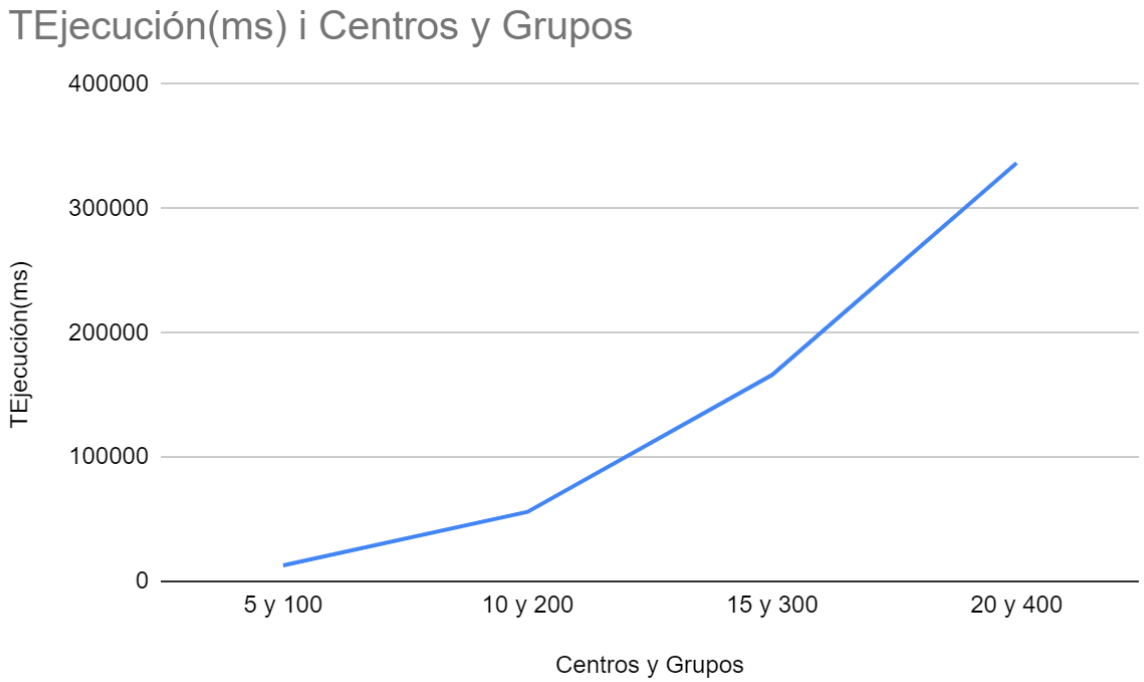
Hipótesis: El tiempo de ejecución depende de del número de centros y grupos

Método:

- Con los dos algoritmos ejecutar 10 seeds con 5 centros y 100 grupos y hacer la media
- Con los dos algoritmos ejecutar 10 seeds con 10 centros y 200 grupos y hacer la media
- Con los dos algoritmos ejecutar 10 seeds con 15 centros y 300 grupos y hacer la media
- Con los dos algoritmos ejecutar 10 seeds con 20 centros y 400 grupos y hacer la media



Para el Hill Climbing como podemos observar a medida que aumentamos el número de centros y de grupos el tiempo de ejecución crece exponencialmente



Con el algoritmo Simulated Annealing se puede notar un gran incremento exponencial en los tiempos de ejecución. Sorprendentemente, la curva es menor en este algoritmo que en el Hill climbing. Aunque los tiempos con 5 centros eran muy

bajos en el Hill Climbing y más altos en en Simulated Annealing, con 20 centros el tiempo medio de ejecución del Hill Climbing es bastante más alto que el de este anterior.

5. **Dado el escenario de los apartados anteriores, estudiad cómo evoluciona el tiempo de ejecución para hallar la solución para valores crecientes de los parámetros del problema por separado. Un primer escenario en el que aumenta el número de grupos a rescatar partiendo de 5 centros y 100 grupos, aumentando los grupos de 50 en 50 hasta que se vea la tendencia. Un segundo escenario en el que aumenta el número de centros de rescate partiendo de 5 centros y 100 grupos, aumentando los centros de 5 en 5. Usad solamente el algoritmo de Hill Climbing.**

Primer escenario: Aumentar grupos

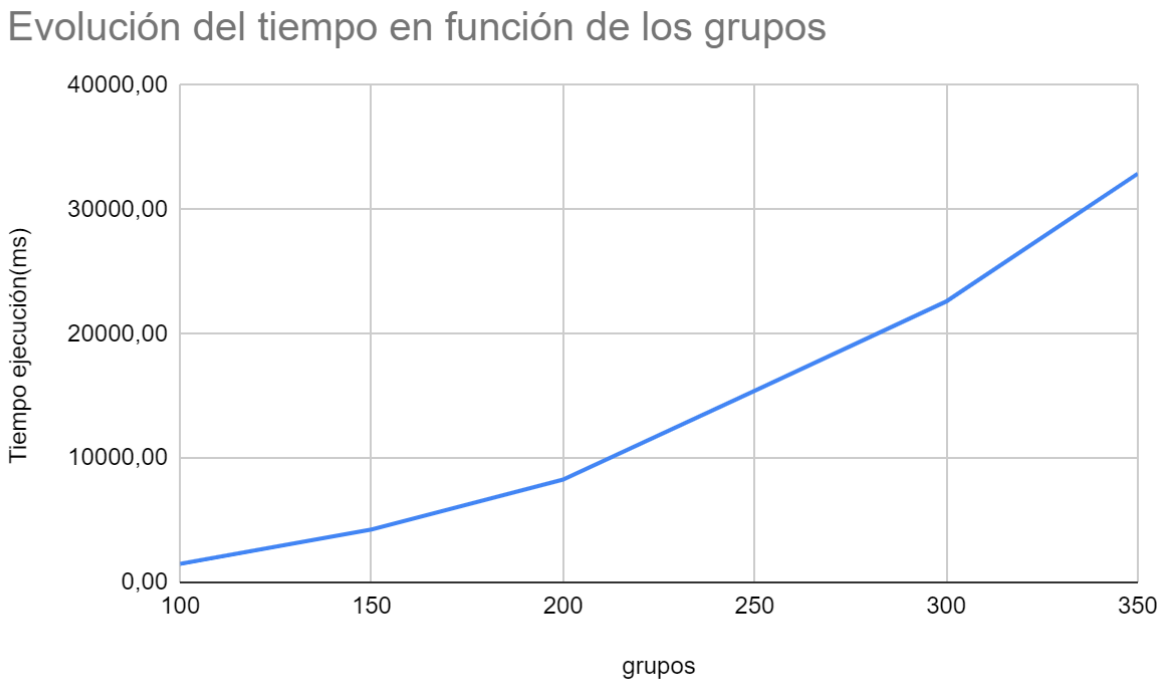
Observación: Al aumentar el número de grupos por separado el tiempo de ejecución aumenta

Planteamiento: Generamos soluciones variando el número de grupos

Hipótesis: El tiempo de ejecución depende del número de grupos

Método:

- Con grupos igual a 100, 150, 200, 250, 300, 350
- Ejecutamos 10 seeds aleatorias



Podemos observar que el tiempo de ejecución depende directamente del número de grupos a rescatar. Como podemos ver el tiempo de ejecución aumenta más o menos linealmente a medida que aumenta el número de grupos

Segundo escenario: Aumentar centros

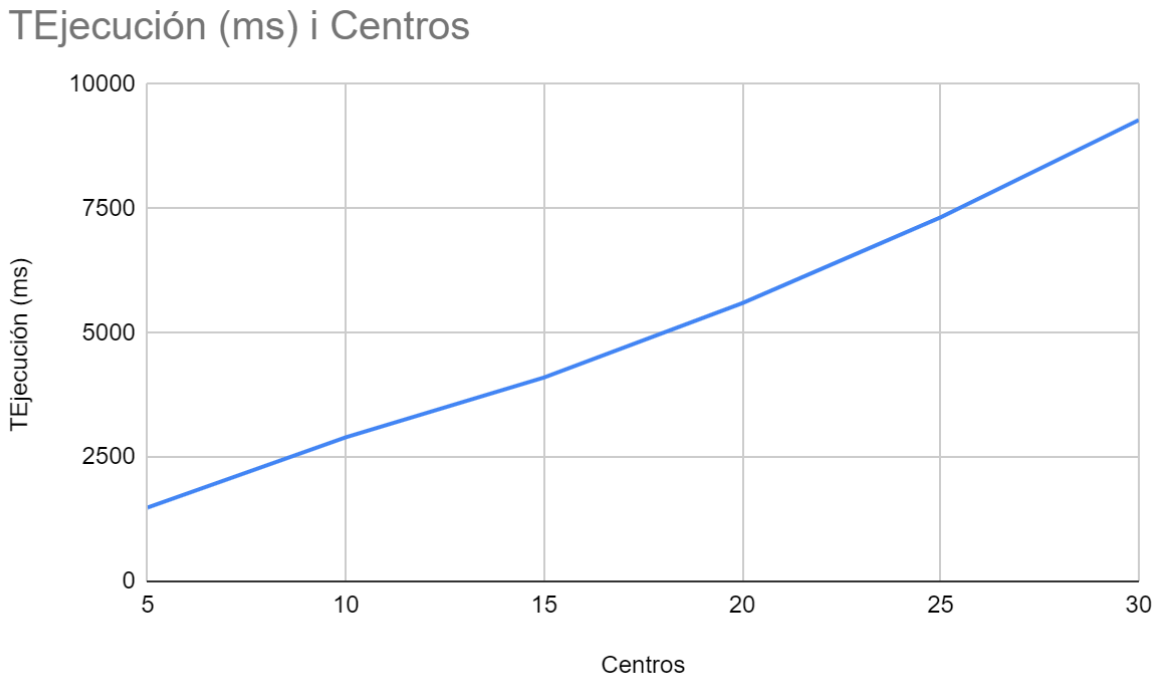
Observación: Al aumentar el número de centros por separado el tiempo de ejecución aumenta

Planteamiento: Generamos soluciones variando el número de centros

Hipótesis: El tiempo de ejecución depende del número de centros

Método:

- Con centros igual a 5, 10, 15, 20, 25, 30
- Ejecutamos 10 seeds aleatorias



En este caso se puede apreciar más que el incremento del número de centros hace que el tiempo de ejecución incremente de forma lineal. Hemos escogido hasta 30 centros, cada prueba haciendo la media con 10 seeds diferentes, y hemos visto que esto era suficiente para ver que el tiempo de ejecución depende directamente del número de centros.

6. **En los experimentos hasta ahora solo había un helicóptero por centro de rescate. Antes de hacer más experimentos ¿Cómo creéis que afectará aumentar el número de helicópteros por centro a la calidad de la solución? ¿Y al tiempo para hallar la solución? ¿Es lo mismo que aumentar el número de centros con un solo helicóptero? Dad una explicación detallada de vuestras suposiciones e intentad confirmarlas experimentalmente usando las condiciones del primer apartado y el algoritmo de Hill-Climbing.**

En el caso de aumentar los helicópteros creemos que la solución mejorará al aumentar el número de helicópteros por centro ya que así el tiempo de rescate de algunos grupos será menor y como la solución viene dada por la suma de todos los tiempos de rescate. En cambio el tiempo de ejecución será más grande ya que el factor de ramificación del operador aumentará. Pasa algo más o menos como con

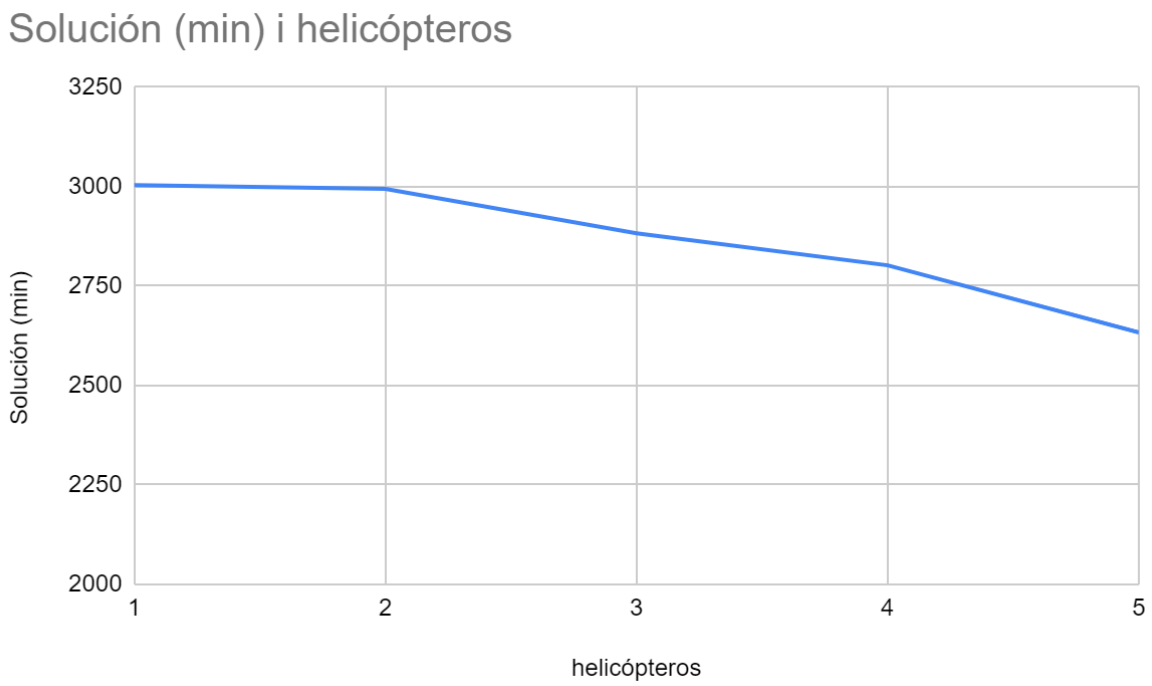
los centros, ya que a más centros más helicópteros. Para confirmar nuestras suposiciones haremos un experimento

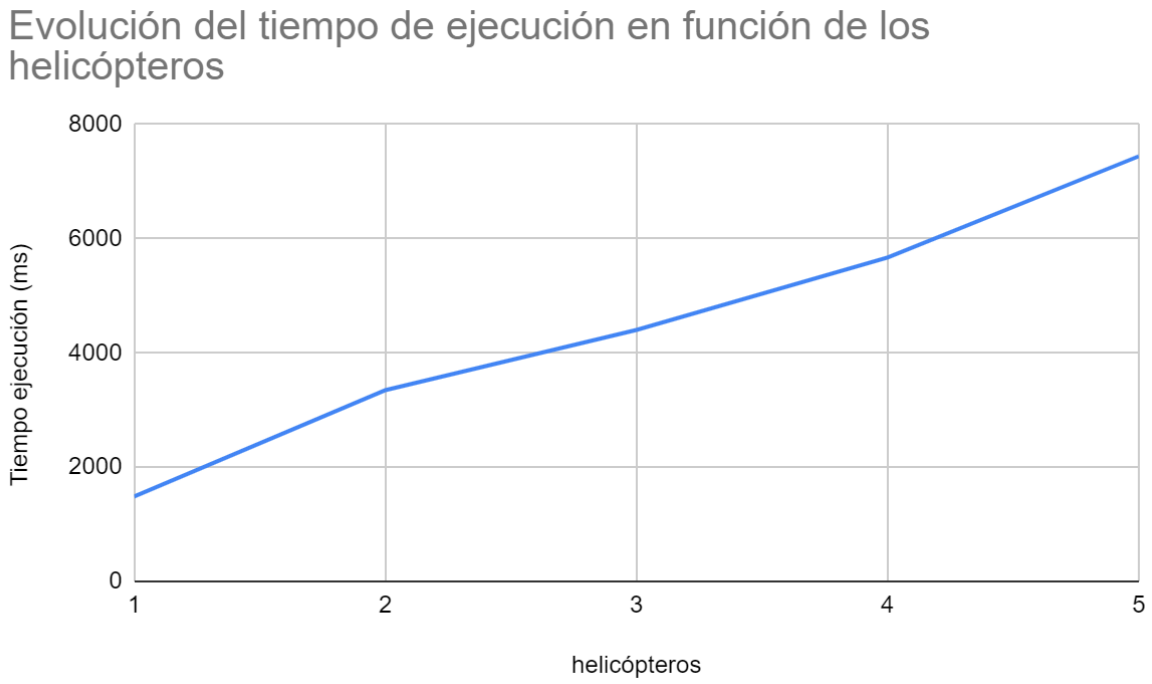
Planteamiento: Generamos soluciones variando el número de helicópteros

Hipótesis: El tiempo de ejecución y la solución dependen del número de helicópteros

Método:

- Con helicópteros igual a 1, 2, 3, 4, 5
- Ejecutamos 10 seeds aleatorias





Como podemos observar las suposiciones que habíamos hecho se cumplen y el tiempo de ejecución aumenta de manera lineal a medida que aumenta el número de helicópteros por centro, mientras que la solución obtenida disminuya a medida que aumenta el número de helicópteros de cada centro.

7. **La segunda función heurística introduce un segundo criterio que parece similar al primero pero que cambia la forma en la que se da importancia a los tiempos de rescate ¿Explicad la diferencia entre los dos criterios? Realizad experimentos con una segunda función heurística que sea la suma de los dos criterios y observad como cambia la suma total del tiempo de rescate. Usad las condiciones del primer apartado usando los algoritmos de Hill-Climbing y Simulated Annealing. Repetid los experimentos añadiendo una ponderación a este segundo criterio y observad como cambia la suma total del tiempo de rescate al cambiar la ponderación. Podéis comenzar doblando el peso del segundo criterio y continuar doblándolo hasta ver la tendencia.**

Con la primera heurística no se tiene en cuenta la prioridad de rescatar a los heridos antes, pero esto cambia con la segunda. Con esta heurística que usaremos en adelante se le da prioridad a los grupos en que haya al menos una persona herida

para rescatarla lo antes posible. Compararemos los tiempos en que todos los grupos de prioridad són rescatados y el tiempo total de rescate para cada algoritmo.

Planteamiento: Generamos soluciones midiendo el tiempo hasta que todos los grupos de prioridad son rescatados y el tiempo total de rescate.
Hipótesis: Los rescates de prioridad acaban mucho antes que todos los rescates.
Método:

- Ejecutamos 10 seeds aleatorias
- Utilizaremos los valores iniciales de 100 grupos, 5 centros y 1 helicóptero por cada centro.
- Utilizaremos los algoritmos Hill Climbing y Simulated Annealing para observar las diferencias.

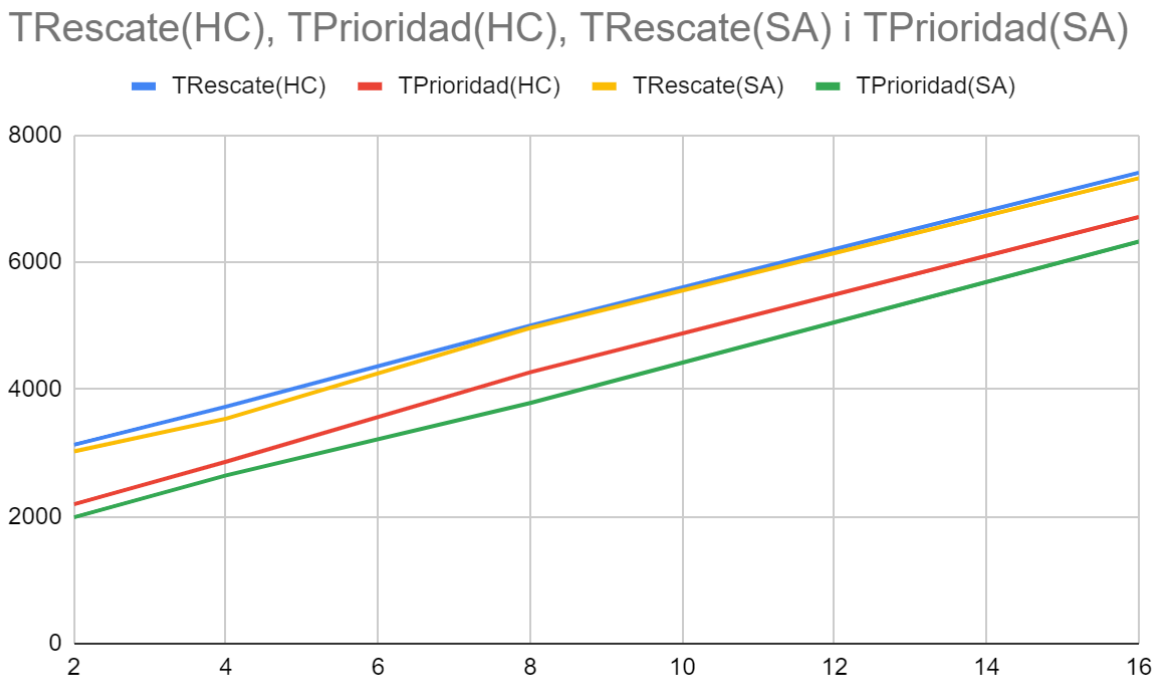
Réplica	Hill Climbing		Simulated Annealing	
	TRescate(min)	TPrio(min)	TRescate(min)	TPrio(min)
1	2736.27	1943.34	2946.22	1837.98
2	3541.73	2627.25	3742.81	2733.03
3	2775.71	2005.72	3023.48	1986.16
4	3129.29	2191.90	2911.73	1843.62
5	3587.24	2910.83	3756.57	2694.00
6	3093.19	2431.88	2961.96	1937.77
7	2979.86	2205.94	2916.30	2130.15
8	3219.65	2113.57	3363.05	1925.04
9	3197.97	2158.16	3225.14	2183.16
10	2743.85	1984.24	2741.24	2010.98
Media	3129.29	2191.9	3023.48	1986.16

En la tabla anterior podemos ver que los tiempos en que todos los grupos de prioridad son rescatados son bastante inferiores al total. También, otra curiosidad que nos encontramos, es que la media en el Simulated Annealing es más baja que en el algoritmo Hill Climbing.

Ahora añadiremos una ponderación a ambos algoritmos para ver cómo evoluciona el tiempo total de rescate y el tiempo de rescate a los grupos prioritarios.

Planteamiento: Generamos soluciones midiendo el tiempo hasta que todos los grupos de prioridad son rescatados y el tiempo total de rescate, añadiendo una ponderación cada vez mayor, multiplicando el tiempo de rescate a los grupos de prioridad por dos.
Hipótesis: Tanto el tiempo total de rescate como el tiempo de rescate a grupos de prioridad aumentarán de forma equitativa.
Método:

- Ejecutamos 10 seeds aleatorias
- Utilizaremos los valores iniciales de 100 grupos, 5 centros y 1 helicóptero por cada centro.
- Multiplicamos por dos el tiempo de rescate a los grupos de prioridad.
- Utilizaremos los algoritmos Hill Climbing y Simulated Annealing para observar las diferencias.



Como podemos ver en la gráfica, donde hemos recogido datos multiplicando el tiempo de rescate por dos hasta un total de 16 minutos, los tiempos de rescate total y a los grupos de prioridad incrementan linealmente. En este caso también, como hemos comentado antes, el tiempo del Simulated Annealing es inferior al del Hill Climbing, aunque este último tenga un tiempo de ejecución bastante inferior.