Pràctica de Cerca Local
Intel·ligència Artificial

Implementación del estado
Operadores elegidos
Estrategias para hallar la solución inicial
Funciones heurísticas

/**********************/

Experimentos
Comparación HC vs SA
Preguntas del enunciado

Implementación del estado

La clase del estado cuenta con un seguido de atributos que almacenan la información necesaria para definir una solución del problema (no necesariamente optimizada). Estos parámetros son: managedCentre, G, C, groupsAdjM, centresAdjM, distance, rescT1, rescT2.

managedCentre es una ArrayList que representa propiamente la situación final. Cada elemento representa un Centro, que a su vez es un ArrayList que contiene los diferentes Paths que se hacen des de ese centro. Path es un struct que representa los grupos que un helicóptero recoge al realizar una salida des de un centro. Hemos considerado no diferenciar qué helicóptero realiza el recorrido pues lo que buscamos minimizar es el tiempo de vuelo de los helicópteros, es decir, la cantidad de gasolina que se consume. Por lo tanto no nos importa qué helicóptero realize el recorrido, pues el consumo será el mismo. Además, gracias a esta estructura podemos almacenar la solución en una estructura de coste espacial Θ(|G|).

G y ***C*** representan respectivamente los grupos y los centros del estado actual. Es donde se guarda propiamente la información de todos los grupos y centros.

groupsAdjM y ***centresAdjM*** son dos matrices que representan la distancia (Double) entre los propios grupos y entre centros y grupos, respectivamente. La usamos al inicializar el estado para no tener que calcular dinámicamente la distancia entre centros y grupos, y así poder acceder a ellas al calcular el tiempo de una solución.

distance, **extraR*escueTime1*** y **extraR*escueTime2*** nos sirven para agilizar el cómputo del tiempo total de vuelo en el heurístico. Distance representa la distancia total que realizan todos los helicópteros en un estado concreto y extraRescueTime1 y extraRescueTime2 representan el tiempo de recogida de los grupos (extraRescueTime2 considera la diferencia de prioridad entre los grupos a recoger). Gracias al uso de éstos parámetros, en el heurístico solo se tiene que pasar distance a tiempo (por medio de la velocidad de los helicópteros) y sumar al tiempo de extraRescueTime1 o extraRescueTime2, dependiendo del heurístico que usemos.

Hace falta añadir que estos tres últimos parámetros son correctamente modificados en los distintos operadores que usamos, descritos a continuación.

Operadores

Para la realización de esta práctica hemos definido cuatro operadores distintos: swap grupos, swap centros, move i move nou.

Swap grupos intercambia dos grupos distintos cualesquiera, tanto dentro y fuera de un Path como dentro y fuera de un centro. De éste modo consideramos también la diferencia de tiempo al realizar un Path de distintas formas. Considerando que este cambio se realiza entre todos los grupos tenemos en el caso peor un coste temporal de O(G*G).

Swap paths intercambia dos paths distintos cualesquiera, tanto dentro como fuera de uncentro. De éste modo consideramos la diferencia de tiempo al realizar un Path desde distintos centros. Considerando que este cambio se realiza entre todos los paths, tenemos en el caso peor un coste temporal de O(G/3*G/3) si los grupos son de 3, O(G/2*G/2) si són de 2… con lo que se trata de O(G*G).

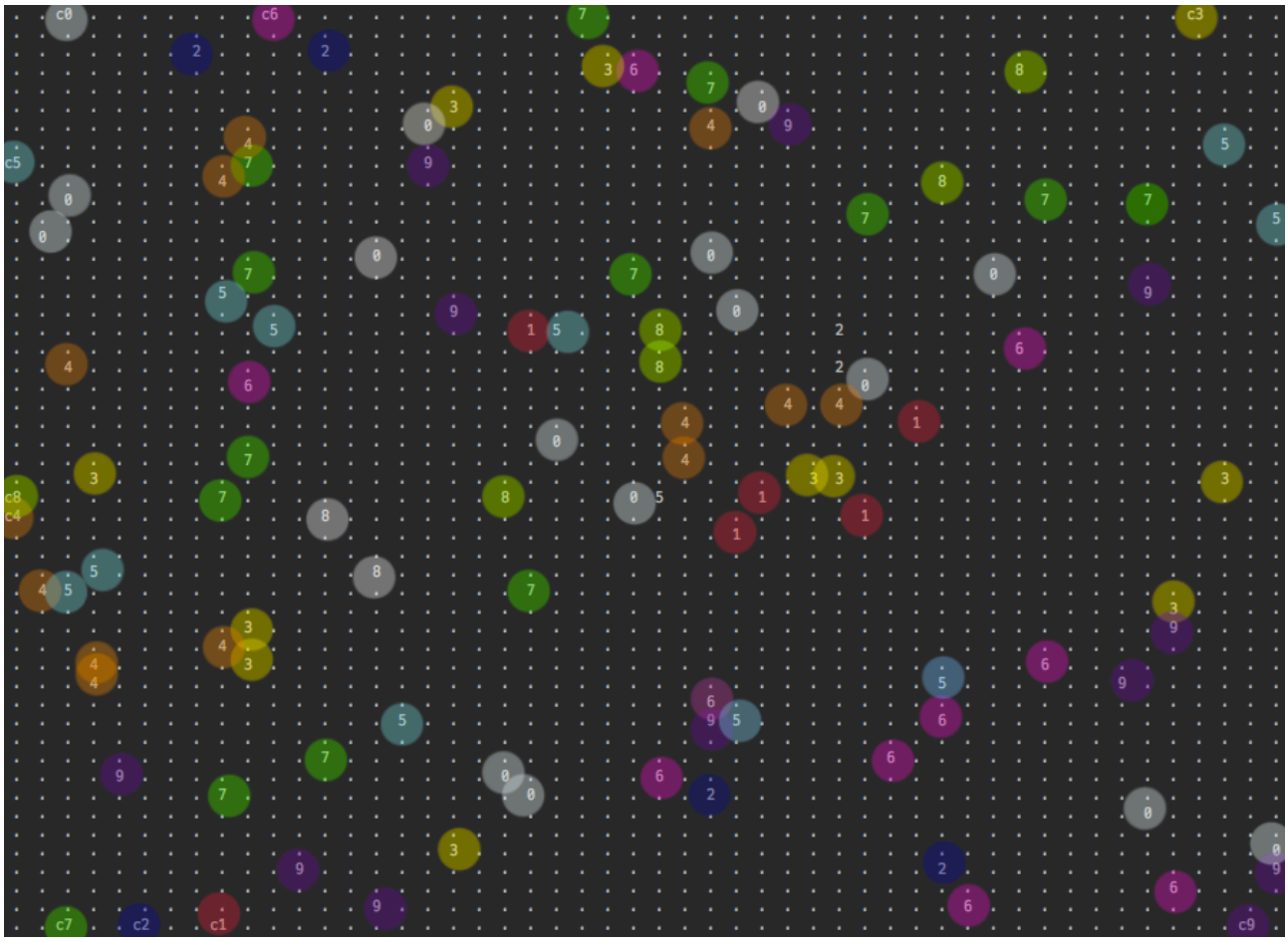
Move añade a un Path existente y no completo un grupo perteneciente a otro Path. El peor caso es aquel en el que cada grupo se encuentra en un Path no completo distinto, con lo que el coste temporal sería otra vez O(G*G). Sin embargo este caso no se dará nunca pues las distintas formas que tenemos de generar la solución inicial satura los Paths, es decir, fuerza que estén completamente llenos.

Move nou selecciona un grupo de un Path, lo elimina de dicho Path y crea un nuevo Path con este grupo. Seguidamente lo añade a un centro cualquiera, incluído el centro origen del grupo. En este caso el coste temporal es O(G*C), pues asignamos cada grupo a cada centro para generar los distintos sucesores.

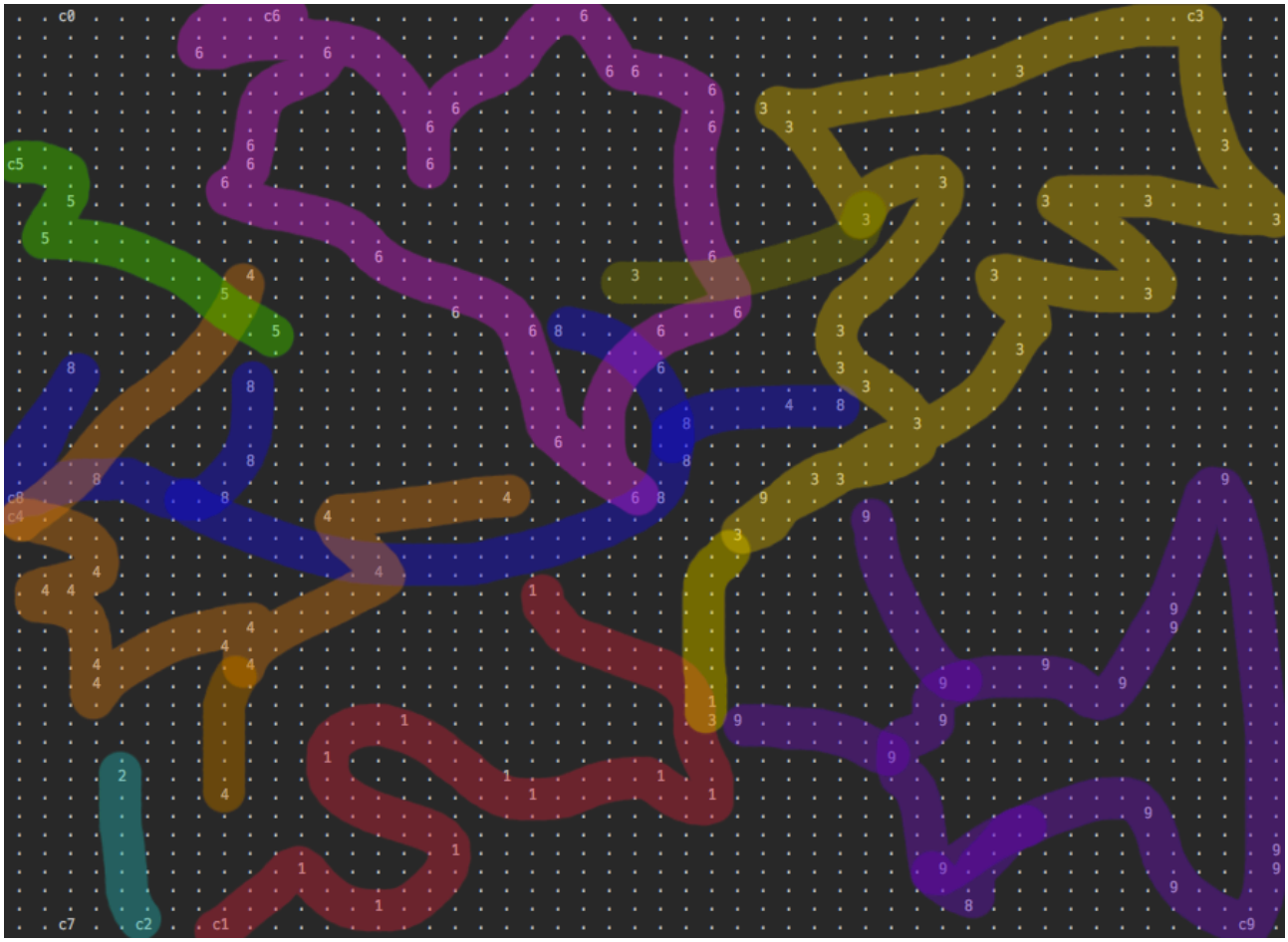
Estos operadores estan pensados para ser usados por AIMA para optimizar la solución descrita en el siguiente apartado.

Solución inicial

Hemos realizado dos versiones distintas para generar la solución inicial: Dummy y Efficient. Ambos tienen un aspecto común, y es que los grupos de personas se agrupan de forma que cumplan las dos restricciones principales: solo podemos agrupar 3 o menos grupos y estas agrupaciones contienen 15 o menos personas. De esta forma nos ahorramos comprovar el cumplimiento de estas restricciones durante todo el proceso, exceptuando los casos en los que debamos comprobar si podemos aplicar un operador. La diferencia fundamental entre los dos algoritmos es la asignación de los Paths a los centros. Mientras Dummy lo hace de forma aleatoria, Efficient calcula el centro más propero a dos de los elementos del grupo (exceptuando los casos en los que el Path solo tiene un grupo, que busca el centro más cercano a ese grupo). Nuestra hipótesis es que dadas estas tres situaciones como base, Efficient va a tardar un tiempo menor a su predecesor pues va a ahorrarnos un buen número de iteraciones de los operadores.



Ejemplo de situación incicial generada por Dummy



Ejemplo de situación incicial generada por Efficient

Las dos capturas a continuación muestran la situación inicial devuelta por Good y Efficient, respectivamente. Se puede apreciar claramente la mejora en la asignación de Paths a centre.
(***cX*** representa el centro x, y ***X*** representa un grupo asignado a un path del centro x).

Hemos considerado únicamente estas dos posibilidades pues creemos que son suficientes para mostrar el funcionamiento de AIMA según la calidad de la solución inicial. Por un lado como fita inferior tendríamos el caso de asignación aleatoria (Dummy). Las restricciones se cumplen, se busca una proximidad entre grupos pero se ignora la distancia entre el centro y los grupos. Por otro lado Efficient busca dar una solución con cierta calidad, pues se centra en minimizar todas las distancias. Realizar un algorismo inicial más cercano a la solución óptima hubiése sido una pérdida de tiempo computacional, y es preferible dejar a AIMA realizar el trabajo de optimización por medio de los operadores.

Función heurística

Como hemos comentado con anterioridad, hemos buscado liberar de trabajo la función heurística calculando dinámicamente el coste de cada solución mientras la generamos/modificamos. De esta manera, la única operación que se debe realizar en la función heurística es transformar la distancia recorrida por los helicópteros a tiempo (sabemos que los helicópteros van a una velocidad de 100 km/h, con lo que se trata de un cálculo simple), y sumarlo al tiempo de recogida de personas. Diferenciamos por lo tanto dos funciones heurísticas, una para el caso sin prioridades y una para el caso con prioridades.

Experimentos

Para cada experimento deberéis hacer como mínimo 10 repeticiones y calcular valores medios. Podéishacer gráficas y estadísticas que ilustren vuestras conclusiones, obviamente los que lo hagáis tendréis una mejor valoración

Experimento 1

Condiciones del experimento: *Análisis de la calidad de los operadores mediante* Hill Climbing , sobre un escenario con 5 centros, 100 grupos y un helicóptero por centro. La solución inicial viene dada por Dummy.

Resultados del experimento:

Resultados esperados y resultados obtenidos: Con este experimento buscamos encontrar los operadores que dan mejor resultado. Para verlo con más claridad hemos decidido usar el generador Dummy para así notar con más facilidad las diferencias entre operadores.

Consideramos que los operadores Move y Move nou son operadores necesarios en cualquier escenario para conseguir el valor óptimo, ya que al ser el combustible el parámetro a minimizar nos interesa crear nuevos paths con grupos cercanos a un centro, pese a que así no saturemos los Paths. Solo consideraríamos saturar los Paths en una situación en la que lo que interesase fuese minimizar el tiempo de recogida en paralelo de los helicópteros. Por lo tanto en este experimento queremos demostrar la necesidad de estos dos operadores y ver qué es más útil, el swap de grupos o de paths.

Conclusiones:

Experimento 2

Condiciones del experimento: Análisis de la calidad del generador de solución inicial mediante Hill Climbing, sobre un escenario con 5 centros, 100 grupos y un helicóptero por centro.

Resultados del experimento:

Resultados esperados y resultados obtenidos: *Con este experimento buscamos ver que el generador Efficient es mejor que Dummy. Tal y como hemos visto con anterioridad ya las primeras pruebas independientes muestran que así es, sin embargo hemos realizado el experimento con la integración de AIMA para asegurar que la mejora en la solución inicial tiene un impacto a posteriori.*

Conclusiones:

Experimento 3

Condiciones del experimento: Mismo escanerio que en el experimento cambiando el algorismo de búsqueda por Simulated Annealing.

Resultados del experimento:
Resultados esperados y resultados obtenidos:
Conclusiones:

Experimento 4

Condiciones del experimento: Análisis de la evaluación del tiempo de ejecución de los dos algorismos anteriores, sobre el mismo escenario y aumentando progresivamente el tamaño de la entrada siguiendo una tendencia 5:100 y aumentando los centros de 5 en 5.

Resultados del experimento:
Resultados esperados y resultados obtenidos:
Conclusiones:

Experimento 5

Condiciones del experimento: *Análisis de la evaluación del tiempo de ejecución de los dos algorismos anteriores y sobre el mismo escenario. Evaluaremos dos progresiones distintas a partir de 5 centros y 100 grupos, por un lado aumentando de 50 en 50 los grupos y por el otro aumentando los centros de 5 en 5.*

Resultados del experimento:
Resultados esperados y resultados obtenidos:
Conclusiones:

Experimento 6

Condiciones del experimento: *Análisis de la variación del número de helicópteros por centro en la situación anterior.*

Resultados del experimento: - -
Resultados esperados y resultados obtenidos: - -
Conclusiones: Tal y como hemos planteado el estado este experimento no altera para nada los resultados de las situaciones anteriores. Como hemos comentado con anterioridad el objetivo es minimizar el consumo de combustible de los helicópteros, con lo que no nos importa cual sea el helicóptero que realice el vuelo. No existe propiamente la estructura de datos helicóptero, los centros almacenan únicamente los recorridos (paths) que se deben realizar des de cada centro.

Experimento 7

Condiciones del experimento:
Resultados del experimento:
Resultados esperados y resultados obtenidos:
Conclusiones:

Hill Climbing vs Simulated Annealing

Preguntas del enunciado