**Práctica de Búsqueda Local**

Dominguez, Alejandro

Munuera, Adrià

Muñoz, Miguel Ángel

**Índice**

|  |  |
| --- | --- |
|  | **Página** |
| **Informe** | **3** |
| **Experimentos** | **4** |
| **Experimento 1** | **4** |
| **Experimento 2** | **5** |
| **Experimento 3** | **6** |
| **Experimento 4** | **7** |
| **Experimento 6** | **8** |
| **Conclusiones** | **9** |

**Informe**

Nuestro informe sobre las clases utilizadas y como han sido implementadas residen en un conjunto de páginas html creadas por javadoc.

Entre ellas se puede navegar para ver cada una de las funciones utilizadas y creadas para el propósito de este trabajo.

El archivo LEEME.txt indica como se ha de ejecutar y trabajar con el proyecto.

**Experimentos**

**Experimento 1**:

Determinar qué conjunto de operadores da mejores resultados para una función heurística que optimice el criterio de calidad del problema (3.2) con un escenario en el que el número de centros de datos es 4 y el de sensores 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. Pensad que con estas proporciones, se podrán transmitir todos los datos.

**Hipótesis del experimento:**

Nuestra hipótesis sobre el experimento es que el uso de ambos operadores resulta más beneficioso para encontrar el cable, puesto que nos devuelve un mayor número de sucesores posibles para mejor solución.

**Metodología:**

Realizaremos una serie de 10 casos aleatorios (con semillas aleatorias) en los cuales el número de sensores resulta de 100 mientras que son 4 centros de datos a la vez. Nuestros operadores son el **cambio de cable** desde un sensor hacia otro o un centro, y el **intercambio de cables** entre dos sensores que estaban conectados a dos elementos diferentes.

Hemos creado tres maneras diferentes de generar escenarios iniciales de los cuales, usaremos el tercer método que consta de conectar todos los mismos sensores que capturan 5 Mb/s a los diferentes centros de datos, los sensores que capturan 2 Mb/s a los sensores de 5 Mb/s y todos los sensores que capturan 1 Mb/s a los que capturan 5 Mb/s.

**Resultados:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Mejora en coste según el operador.** | | |
| **Nº de ejecución** | **Operador: changeCable** | **Operador: switchCable** | **Ambos operadores** |
| 1 | 1464296.0 | 1461848.0 | 1491795.0 |
| 2 | 1177974.0 | 1171443.0 | 1199479.0 |
| 3 | 1179161.0 | 1199378.0 | 1212231.0 |
| 4 | 1132903.0 | 1143862.0 | 1162084.0 |
| 5 | 970483.0 | 1005749.0 | 1022280.0 |
| 6 | 1392085.0 | 1404235.0 | 1421974.0 |
| 7 | 966229.0 | 995539.0 | 1011223.0 |
| 8 | 1148816.0 | 1157155.0 | 1174910.0 |
| 9 | 1276178.0 | 1314123.0 | 1340949.0 |
| 10 | 763346.0 | 812198.0 | 832942.0 |

Al ser la mejora que gana en coste total, respecto a la generación inicial que tomábamos como solución inicial, cuanto mayor sea, mejor resulta.

Para todas las ejecuciones, vemos que el uso de los dos operadores siempre es mucho mejor que solamente usar uno de ellos. Aparte, en la mayoría de casos, el uso del operador switchCables resulta más beneficioso en coste total del experimento, que el uso del operador changeCable.

Con lo cual vemos que nuestra hipótesis se cumple.

**Experimento 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.

**Hipótesis del experimento:**

Pensamos que la solución inicial creada por el tercer generador será la mejor aprovechada por el Hill Climbing.

**Metodología:**

Realizando 10 ejecuciones como anteriormente, hemos creado tres soluciones iniciales, una con cada generador, los cuales cada una consigue la misma semilla, y esta es diferente para cada iteración. Aplicamos Hill Climbing para conseguir el coste con cada solución inicial.

**Resultados:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Coste final según el generador usado.** | | |
| **Nº de ejecución** | **Generador de soluciones 1** | **Generador de soluciones 2** | **Generador de soluciones 3** |
| 1 | 181999.0 | 181999.0 | 76997.0 |
| 2 | 257444.0 | 257444.0 | 87856.0 |
| 3 | 207570.0 | 207570.0 | 79238.0 |
| 4 | 163679.0 | 163679.0 | 70241.0 |
| 5 | 395537.0 | 395537.0 | 72238.0 |
| 6 | 392638.0 | 392638.0 | 87804.0 |
| 7 | 329006.0 | 329006.0 | 77066.0 |
| 8 | 271662.0 | 271662.0 | 64186.0 |
| 9 | 288446.0 | 288446.0 | 76083.0 |
| 10 | 271437.0 | 271437.0 | 76635.0 |

Con el coste final, nos fijamos que el generador 1 y el generador 2, sabiendo que tienen una solución inicial diferente, devuelven el mismo resultado en coste final. Por otra parte, vemos que el tercer generador reduce enormemente el coste final respecto a los demás.

Lo que implica, que el tercer generador, da lugar a mejores soluciones.

**Experimento 3:**

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

**Hipótesis del experimento:**

Sabiendo el funcionamiento de Simulated Annealing, suponemos que con una alta temperatura, el rendimiento del algoritmo será mucho mejor que si esta temperatura es mucho menor.

**Metodología:**

Hemos realizado una pequeña búsqueda de experimentos realizados por Simulated Annealing donde el resultado parecía bastante correcto, y por tanto lo dábamos por válido (claramente después de comparar los algoritmos usados con los dados por las transparencias).

Nos topamos con un experimento que daba como temperatura 10000 y un cooling rate de 0.003. Así pues, después de documentarnos un poco más, vimos que según el departamento de matemáticas de la Universidad de Buenos Aires, la temperatura debe ser proporcional a la raíz cuadrada de los nodos. Así pues nuestra consideración ha sido de colocar una temperatura media según esta fórmula donde *nsens* representa el número de sensores y *ncent* el número de centros.

Para el factor de enfriamiento (cooling rate) hemos tenido algunos problemas. Primeramente, decidimos usar un factor que sea proporcional al usado por el experimento que vimos con anterioridad. con lo cual, usamos un cooling rate de 0.075. Este factor terminaba siendo tedioso el tiempo que necesitaba para realizar el experimento. Pensamos en multiplicar por diez el factor, pero no aprovecharíamos todo el potencial de Simulated Annealing. Así pues, optamos por la media de estos dos valores y terminamos usando el cooling rate de 0.0375 que trata de .

**Resultados:**

|  |  |  |
| --- | --- | --- |
| **Nº de la prueba** | **Temperatura** | **Coste final** |
| 1 | 12000 | 137687.0 |
| 2 | 15000 | 134920.0 |
| 3 | 18000 | 128982.0 |
| 4 | 22000 | 125422.0 |
| 5 | 25000 | 124553.0 |
| 6 | 28000 | 118893.0 |
| 7 | 31000 | 115639.0 |
| 8 | 34000 | 111217.0 |
| 9 | 37000 | 108234.0 |
| 10 | 40000 | 106982.0 |

**Experimento 4:**

Dado el escenario de los apartados anteriores, estudiad como evoluciona el tiempo de ejecución para hallar la solución para valores crecientes de los parámetros siguendo la proporción 4:100. Comenzad con 4 centros de datos e incrementad el número de 2 en 2 hasta que se vea la tendencia. Usad el algoritmo de Hill Climbing y la misma función heurística que antes.

**Hipótesis del experimento:**

**H:**  El tiempo evoluciona creciendo de manera exponencial o casi exponencial.

**H0:** El tiempo evoluciona decreciendo o manteniendose.

**Metodología:**

Se ha programado el experimento 4 para ir ejecutando el algoritmo de Hill Climbing con centros y sensores en proporción 1:25. Se ha programado un bucle donde en cada iteración aumentaba el número de centros en 2 y de sensores en 50.

**Resultados:**

Se ha comprobado en 3 ordenadores diferentes que el factor de aumento de tiempo entre una iteración y otra es de tiempo\_anterior x 10. Por lo tanto, la evolución del tiempo es exponencial. Para uno de nuestros ordenadores, la primera iteración tarda 49s, la segunda 362s, y la tercera cercana a la hora.

|  |  |
| --- | --- |
| **Nº de iteración** | **Tiempo de ejecución** |
| 1 | 49 segundos |
| 2 | 6 minutos y 2 segundos |
| 3 | cercana a la hora. |

**Experimento 6:**

Suponiendo que los centros de datos no sean costosos, podríamos estimar como afecta el añadir más centros al coste de la red. Fijando el numero de sensores en 100, realizad experimentos aumentando el número de centros de datos de dos en dos hasta 10 y medid el coste de la red de conexión, el número de centros de datos usados y el coste temporal para hallar la solución. Usad el algoritmo de Hill y el de Simulated Annealing. Climbing.

**Hipótesis del experimento:**

Sabiendo que nuestros operadores sobre todo hacen cambios entre sensores, suponemos que el añadir centros tendría que ser extremadamente barato.

La hipótesis nula por tanto consistiría en que añadir centros tiene un coste alto.

**Metodología:**

Hemos creado una nueva clase que implementa todo lo necesario para llevar a cabo el experimento. Hemos recogido los datos después de la ejecución del programa y hemos sacado conclusiones

**Resultados:**

Se confirma nuestra hipótesis totalmente: añadir centros es barato.

En el documento *EXP6\_RESULTS.txt* se pueden visualizar los resultados obtenidos.

Como conclusiones, decir que prácticamente el tiempo de ejecución de cada iteración es similar. También vemos una muy lógica subida del volumen de datos que nos llega y una bajada del precio total del circuito de cables.

**Conclusiones**

La búsqueda local usa algoritmos que dados todos los sucesores posibles que se pueden generar con los operadores que se han creado previamente, realiza una inspección de estos y elige entre ellos cual es el supuesto mejor estado sucesor.

Entre estos algoritmos, practicamos Hill Climbing y Simulated Annealing.

En el primero realiza una inspección entre todos los posibles estados sucesores y elige entre ellos el mejor. Una apuesta segura, pero más tardía que el algoritmo siguiente.

En Simulated Annealing, dados dos valores, uno de temperatura y otro de refrigeración, consigue un porcentaje, con el que calculará si se quedará el supuesto estado sucesor o no. Repetirá el proceso con todo estado hasta que la temperatura llegue a un nivel inferior a uno.

Hemos pasado por una lista de varios operadores, por ejemplo, al inicio de la práctica, usábamos operadores como conectar cables entre sensores, cable entre sensor y centro, entre otros. Pero el espacio de sucesores se volvería exageradamente grande. Por ello, terminamos consiguiendo reducir el número de operadores a dos. Cambiar cable entre estructuras, y intercambiar cables. Esto nos da una amplia gama de sucesores y el espacio de estos permite trabajar cómodamente.

Hemos llegado a tener algunos problemas en la realización de los algoritmos de Hill Climbing y Simulated Annealing al no haber trabajado antes con ellos, por otra parte, otro problema surgió a la hora del tratamiento entre datos como son los sensores y los centros. Solucionados todos ellos, hemos realizado experimentos con pocos problemas.