Metaheurística híbrida Tabú Genético para el problema p-hub

Pedro Jiménez Latorre[1], César Hervás Martínez[1], A. Tallón[2]

Resumen—**Presentamos un algoritmo de búsqueda tabú (*Tabu Search*) y una hibridación tabú-genético para encontrar soluciones al problema p-hub mediano sin restricciones de capacidad y con asignación simple. Este problema de localización está considerado NP-duro y tiene múltiples aplicaciones en campos como las telecomunicaciones, el transporte, la mensajería etc. El algoritmo propuesto presenta unos parámetros determinados a partir de diseños experimentales. Los resultados obtenidos son competitivos con los resultados de la bibliografía reciente. Finalmente aportamos un valor para el problema con 200 nodos y 8 hubs, mejor que cualquiera de los que hemos tenido acceso en la literatura.**

Palabras clave—**Localización de hub, búsqueda tabú, algoritmo genético, diseño de experimentos.**

I. INTRODUCCIÓN

El problema p-hub mediano con asignación simple, parte de *n* nodos interconectados entre si, entre los que se produce un determinado flujo que origina unos costes.

Este problema tiene múltiples aplicaciones en campos de tanta actualidad como son las telecomunicaciones [4] y [21], telefonía móvil [19], el transporte aéreo o de cualquier tipo [3], [13], [16] y [23], la mensajería [6], [7] y [14], los servicios de emergencia [4], etc.

Para su solución se localizan *p* hubs ($p<n$). Esto es, se encuentran *p* nodos que servirán de enlace entre los demás. Los nodos hub estarán conectados entre si, pero un nodo no hub solamente estará conectado con uno y solo un hub. Este es un problema de localización, conexión y asignación, que encuentra los hubs idóneos y asigna a cada hub un conjunto de nodos. Los conjuntos de nodos asignados a cada hub deben ser disjuntos.

En la literatura encontramos diversas aproximaciones al problema, o a problemas similares, que utilizan diversas técnicas de computación. Las más usadas para afrontar este tipo de problemas son las técnicas de *Branch-and-Bound* [6], [7], [11] y [22], los algoritmos *Greedy* [2] y la búsqueda tabú [1], [12] y [23]. Otras técnicas menos utilizadas son la búsqueda dispersa [9], GRASP [12], el recocido simulado [7] y redes neuronales de tipo Hopfield [25]. También aparecen algoritmos genéticos para resolver este problema que en general son algoritmos híbridos como en [18], que utiliza un algoritmo genético con búsqueda multiarranque y en [1] que aparece un algoritmo híbrido utilizando técnicas de búsqueda tabú.

En [10] se describe un algoritmo genético sin hibridaciones y con parámetros, que lo adaptan para resolver este problema, en cuyas ideas nos hemos basado.

Nuestro objetivo era obtener un algoritmo de búsqueda tabú que permitiera encontrar soluciones de calidad en un tiempo razonable para el problema p-hub mediano en problemas de gran tamaño. Para ello realizamos diversas pruebas estadísticas sobre los problemas con 50 nodos y 3,4 y 5 hubs respectivamente, para los que es conocido el óptimo. Con los resultados obtenidos ajustamos los tamaños de la *lista tabú* y los parámetros de la estrategia de selección *élite-plus* de nuestro algoritmo de búsqueda tabú. Con estos parámetros ejecutamos el algoritmo para un problema de gran tamaño del que aún no sabemos que se haya publicado el óptimo, como es el problema con 200 nodos y 8 hubs, y aportamos un valor mejor que los que anteriormente han sido publicados en la literatura a la que hemos tenido acceso.

En la sección II se describe el problema y se presenta la formulación matemática utilizada para resolverlo. La sección III presenta el algoritmo utilizado. En la sección IV se muestran las pruebas realizadas. Las dos últimas secciones están dedicadas a las conclusiones obtenidas.

II. DESCRIPCIÓN DEL PROBLEMA

Los problemas de localización aparecen cuando los encargados de tomar decisiones deben seleccionar donde se ubicarán una o varias instalaciones o servicios [3]. Las instalaciones pueden ser de muy diverso tipo, como hospitales, industrias, aeropuertos, plantas generadoras de electricidad, mercados, gasolineras etc. Las decisiones, en estos problemas, se toman bajo una serie de criterios preestablecidos, cumpliendo unas determinadas restricciones de forma que se satisfagan de forma óptima las necesidades demandadas por los usuarios. Uno de los criterios determinantes de estos problemas es decidir que norma se utilizará para estimar las distancias [20].

Cuando el problema trata de una localización dentro de una ciudad, se suele usar una norma rectilínea. Que se adapta mejor a la estructura de las

[1] Departamento de Informática y Análisis Numérico. Universidad de Córdoba. Campus Universitario de Rabanales, Edificio C-2.e-mail: {pjimenez, chervas}@uco.es

[2] Departamento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla. E-mail: atallon@us.es

calles que hay que recorrer, para unir dos puntos. Esta norma mide la distancia entre dos puntos en función del camino más corto que hay entre ellos. Cada camino está formado por un conjunto encadenado de segmentos que parten de uno de ellos y llega hasta el otro.

Cuando se trata de problemas en zonas rurales o en el espacio aéreo, se suele utilizar la norma euclídea, que dice que la distancia entre dos puntos viene dada por la longitud del segmento que los une.

Los criterios usados, relacionados de forma directa con el tipo de instalaciones o servicios a localizar, van desde el que precisa que los servicios estén tan próximos a los puntos de demanda como sea posible, hasta el que pretende lo opuesto. Las restricciones pueden imponer que el número de instalaciones a localizar esté prefijado, que los servicios dispongan de ciertas capacidades que no puedan ser sobrepasadas, etc. En función de las combinaciones de criterios, restricciones, entorno sobre el que se desarrolla el problema y cualquier otro elemento que intervenga en la determinación óptima de las instalaciones, aparecen diferentes modelos que son resueltos de manera diferente [3].

El problema del p-hub [16] y [17], es un problema de localización, conexión y asignación, en el que el conjunto de localizaciones coincide con el de puntos de demanda. La asignación se consigue eligiendo los nodos hub de manera que cada nodo no hub esté conectado con uno o con varios nodos hub. Cada hub se conecta con cada uno de los puntos de demanda a los que da servicio, mediante un arco. Todos los hub están conectados mediante un subgrafo completo.

Cuando cada nodo se conecta con un único hub, al problema se le conoce como de asignación simple. El caso en el que los nodos puedan conectarse con más de un hub, se conoce como problema de asignación múltiple.

Las soluciones de estos problemas se pueden representar mediante un grafo $G=(V,E)$, en el que los vértices de G se corresponden con los puntos de demanda y localización. Los arcos de este grafo, representan a las asignaciones o conexiones.

Cuando $p=1$ el problema es trivial, ya que sólo hay un hub al que todos los nodos están conectados. El problema alcanza su máxima complejidad matemática cuando el número de hubs está en el intervalo

$$p \in \left[\frac{n}{2}-1, \frac{n}{2}+1\right]$$

Esto se debe a que el número de Stirling de clase II $S(n.p)$, toma sus mayores valores en ese intervalo.

Sin tener en cuenta las distintas posibilidades de asignación, hay

$\binom{n}{p} = \frac{n!}{p!(n-p)!}$ formas diferentes de escoger p hubs sobre un conjunto de n nodos. Pero cuando le añadimos la asignación entonces el número de soluciones es el número de Stirling de clase II

$$S(n,p) = \sum_{k=o}^{n} \frac{(-1)^k (n-k)^p}{k!(n-k)!}$$

Este es un problema NP-duro en el que es muy adecuada la utilización de metaheurísticas para intentar encontrar soluciones de calidad, aunque no siempre esté garantizado encontrar el óptimo.

La primera formulación del problema p-hub mediano sin restricciones de capacidad y con asignación simple corresponde a O'Kelly [16], En ella se describe al problema teniendo en cuenta que:

W_{ij} es el número de unidades de tráfico que se envían desde el punto i al j.

C_{ij} es el coste unitario por unidad de tráfico enviada sobre el arco (i,j).

Se supone que W_{ii} y C_{ii} son igual a 0, para todo i.

Si tanto i como j son hubs, el coste por unidad de tráfico sobre el arco (i,j) es igual a αC_{ij}. Siendo α el coeficiente de coste para transferencia entre dos hubs.

Se consideran también otros dos coeficientes β y γ que sirven para calcular el costo de "recogida" desde un punto origen hasta un hub, y de "distribución" desde un hub hasta un punto de destino. Estos últimos dos coeficientes se suelen tomar con valor igual a 1.

Las variables de decisión vienen definidas por

$$X_{ij} = \begin{cases} 1, \text{si el punto } i \text{ es asignado al hub } j \\ 0, \text{en otro caso} \end{cases}$$

$$Y_j = \begin{cases} 1, \text{si el punto } j \text{ es un hub} \\ 0, \text{en otro caso} \end{cases}$$

La formulación matemática de O'Kelly para el problema de asignación simple es

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}\left(\beta\sum_{k=1}^{n} X_{ik}C_{ik} + \gamma\sum_{m=1}^{n} X_{mj}C_{mj} + \right. \\ \left. + \alpha\sum_{k=1}^{n}\sum_{m=1}^{n} X_{ik}X_{jm}C_{km}\right) \quad (1)$$

sujeto a las siguientes restricciones

$$\sum_{j=1}^{n} X_{ij} = 1 \quad i = 1,..., n \qquad (2)$$

$$X_{ij} \le Y_j \qquad i, j = 1,..., n \qquad (3)$$

$$\sum_{j=1}^{n} Y_j = p \qquad (4)$$

$$X_{ij}, Y_j \in \{0,1\} \; i, j = 1,..., n \qquad (5)$$

La restricción (2) indica que cada nodo está asignado a uno y sólo un hub. La restricción (3) asegura que un nodo no hub está asignado a otro nodo *j*, solamente si el nodo *j* es un hub. La restricción (4) asegura que el número de hubs sea el adecuado.

Campbell [4] propuso una formulación para el problema p-hub mediano con asignación simple, cuyo número de variables es del orden $O(n^4)$. Para ello, define la fracción de flujo desde un nodo origen *i* hasta un nodo destino *j*, a través de los hubs *k* y *m*, como X_{ijkm}. El coste por unidad de flujo entre *i* y *j*, vía los hubs *k* y *m* es $C_{ijkm} = C_{ik} + \alpha C_{km} + C_{mj}$.

La formulación matemática propuesta por Campbell es:

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{m=1}^{n} W_{ij} C_{ijkm} X_{ijkm} \quad (6)$$

sujeto a

$$\sum_{k=1}^{n} Y_k = p$$

$$\sum_{k=1}^{n}\sum_{m=1}^{n} X_{ijkm} = 1, \; i = 1,..n, j = 1,...,n$$

$$Y_k \in \{0,1\}, \; k = 1,...,n$$

$$X_{jk} \in \{0,1\}, \; i = 1,...,n; \; k = 1,...,n$$

$$X_{ik} \le Y_k, \; i = 1,...,n; \; k = 1,...,n$$

$$\sum_{j=1}^{n}\sum_{m=1}^{n}\left(W_{ij} X_{ijkm} + W_{ji} X_{jimk}\right) = \sum_{j=1}^{n}\left(W_{ij} + W_{ji}\right) X_{ik},$$

$$i = 1,...,n; k = 1,...,n$$

$$X_{ijkm} \ge 0, \; i = 1,...,n; j = 1,...,n; k = 1,...,n; m = 1,...,n$$

Otras formulaciones de este problema se pueden ver en [15], [24] y en [5]. En [15] O´Kelly y otros, presentan una linearización de las formulaciones utilizadas en [24]. Jaime Ebery [5], desarrolla nuevas formulaciones para el problema que son estudiadas para problemas con dos o tres hub solamente.

En nuestro algoritmo hemos utilizado la formulación de O'Kelly (1), a pesar de ser la más compleja, ya que fue la utilizada en [10] y pretendemos comparar este algoritmo con aquel.

III. DESCRIPCIÓN DEL ALGORITMO

El problema que queremos abordar con nuestro algoritmo, se plantea de la siguiente forma: tenemos un conjunto de *n* nodos interconectados entre si, entre los que se produce un determinado flujo que origina unos costes. Para su solución se localizan *p* hubs ($p < n$). Esto es, se encuentran *p* nodos que servirán de enlace entre los demás. Los nodos hub estarán conectados entre si, pero un nodo no hub solamente estará conectado con uno y solo un hub.

A. *Representación del individuo*

Para representar a cada individuo se ha utilizado un esquema de representación entera similar al de [10] pero con una mejora importante en su interpretación.

Su estructura está definida en un vector de enteros de dimensión *n*, en el que los índices representan a cada uno de los nodos de la red y los valores de los mismos definen o distinguen al nodo en cuestión, como hub o como no hub. Los índices comienzan en *0* y acaban en *n-1*. Los valores permitidos para cada nodo, según que sea hub o no lo sea, son los siguientes:

- Los elementos del vector cuyo índice corresponde a un nodo no hub, sólo podrán tener valores comprendidos entre *[0,p-1]*.
- Los elementos del vector cuyo índice corresponde a un nodo hub, tendrán valores comprendidos entre *[p,2p-1]*.

En [10] cada valor asignado a un nodo no hub, indica a qué hub está conectado, teniendo en cuenta que los índices empiezan en 0. Más claramente, esto quiere decir que un valor 2, en uno de estos elementos indica que ese elemento está conectado con el hub con valor 2+*p*. Sin embargo en este trabajo, un valor 2 para un nodo, representa que ese nodo está asignado al tercer hub que aparezca en el vector, independientemente del valor que tenga. Esta nueva interpretación de la representación del individuo supone una mejora importante en el rendimiento de los algoritmos.

B. *Búsqueda tabú*

Las descripciones de esta metaheurística pueden verse en [8].

El algoritmo de búsqueda tabú diseñado en este trabajo utiliza estrategias basadas en memoria a corto plazo y en memoria a largo plazo.

En primer lugar, se selecciona, de forma aleatoria, una solución como *actual* y como *mejor*.

A continuación se realiza una búsqueda local. En cada una de sus iteraciones, se selecciona el mejor vecino de la solución actual. Esta solución *vecino* se convierte en la siguiente solución *actual*. Si, además, es más apta que cualquiera de las conocidas, se establece como *mejor*.

La búsqueda local continúa mientras no se cumpla la *condición de parada*: un estado donde se haya alcanzado el óptimo (cuando se conozca) o donde se haya ejecutado el número máximo de iteraciones. Finalmente, el algoritmo devuelve como *resultado* la solución *mejor* y la *iteración* en la que se encontró.

```
actual = seleccionar-solución-inicial
mejor = actual
i = 0
iteración-mejor = 0

mientras no es-óptimo(mejor) y i < máximo-iteraciones hacer
        actual = seleccionar-mejor-vecino(actual)
        si es-mejor(actual, mejor) entonces
                mejor = actual
                iteración-mejor = i
        fin-si
        i = i + 1
fin-mientras
devolver(mejor, iteración-mejor)
```

Fig. 1. Algoritmo de búsqueda tabú

Para la selección del mejor vecino se selecciona un conjunto de movimientos candidatos, mediante una estrategia *élite-plus*, esta estrategia toma como referencia una solución. Al inicio del algoritmo, es la *actual*; más adelante, serán las que se utilicen para continuar la búsqueda después de un óptimo local. A partir de la solución de referencia, se ejecutan todos los posibles movimientos, con lo que se obtiene un conjunto de nuevas soluciones.

Otra alternativa es usar sólo un subconjunto de esos movimientos. Se ordenan los movimientos en función de la aptitud de las correspondientes soluciones, situando primero los mejores. De esta forma se inicializa la estrategia de selección de candidatos *élite-plus*. Posteriormente, en la primera iteración, se toman los primeros *élite* movimientos y los *plus* movimientos siguientes a estos. En las iteraciones siguientes, se toman los mismos *élite* movimientos y los *plus* movimientos siguientes a los anteriores, volviendo al principio cuando se llegue al final de la lista de movimientos.

Se descartan los movimientos candidatos que sean tabú y no cumplan el criterio de aspiración. El algoritmo utiliza un criterio de *aspiración por objetivo* [14]. De este modo, aunque un movimiento sea tabú, si coincide con la mejor solución conocida, no se rechaza. De entre todos los movimientos que queden, se selecciona aquél que genere la mejor solución, y se registra en la *lista tabú*, donde permanecerá durante *n* iteraciones; siendo *n* un parámetro del algoritmo.

En el siguiente paso, se comprueba si la solución seleccionada es un óptimo local. En caso afirmativo, se ejecuta un algoritmo para continuar desde un óptimo local. Se actualiza la frecuencia de cada nodo como hub a partir de la configuración de la solución seleccionada. Esta información se tiene en cuenta en el algoritmo para continuar desde un óptimo local. Si la solución es lo suficientemente diversa respecto a las soluciones *élite* anteriores, se registra como solución *élite*. El criterio que se sigue para determinar esta condición es que la configuración de hubs sea distinta de la configuración de hubs de todas las soluciones *élite* ya registradas. Esta información también se utiliza en el algoritmo para continuar desde un óptimo local. Finalmente, se devuelve como resultado la solución *vecino*.

```
M = seleccionar-movimientos-candidatos
m = seleccionar-mejor-movimiento-no-tabú-o-si-aspi.(actual, M)
mejor-vecino = generar-solución(actual, m)
añadir-movimiento-a-lista-tabú(m)

si es-óptimo-local(mejor-vecino) entonces
    mejor-vecino = continuar-desde-óptimo-local
fin-si

registrar-frecuencias(mejor-vecino)
registrar-solución-si-diversa(mejor-vecino)

devolver mejor-vecino
```

Fig. 2. Algoritmo para seleccionar el mejor vecino

Cuando se llega a un óptimo local, el algoritmo de búsqueda tabú selecciona una solución distinta para continuar el proceso. El objetivo es encontrar una solución lo suficientemente diversa en cuanto a la configuración de hubs respecto a las ya utilizadas en exploraciones anteriores.

Como resultado de la ejecución del algoritmo para seleccionar el mejor vecino, se dispone de un conjunto de soluciones *élite*, diversas entre si, donde cada una tiene una configuración de hubs diferente. Para continuar desde un óptimo local selecciona la mejor de las soluciones *élite* que no se haya elegido anteriormente, y la devuelve como resultado. Para no prohibir movimientos que podrían ser interesantes con la nueva solución, se limpia la lista tabú, y, para evaluar de nuevo los movimientos, se inicializa la estrategia de selección de candidatos con la solución *élite* seleccionada como referencia.

Sin embargo, es posible que todavía no haya soluciones *élite* disponibles, o que todas ellas se hayan utilizado. En este caso, se examinan las frecuencias de los nodos como hub para generar una solución cuyos hubs hayan sido los menos usados desde que comenzó el algoritmo. De esta forma, también se obtiene una solución diversa. A continuación, y como se describió anteriormente, se inicializan la lista tabú y la estrategia de selección de candidatos.

Como último caso, si el algoritmo no encuentra una solución adecuada para continuar la búsqueda con los dos métodos anteriores, se genera una solución mediante una mutación por nodo no asociado del óptimo local, lo que realiza un cambio estructural importante en la configuración de nodos. Con esta nueva solución se pretende realizar un salto aleatorio para salir del óptimo local.

C. *Algoritmo híbrido*

En este algoritmo se ejecuta el genético durante un número de iteraciones fijado como parámetro. A continuación, se ejecuta el algoritmo de búsqueda tabú partiendo de la solución encontrada por el genético. Si se conoce el óptimo y el genético lo encuentra, no es necesario ejecutar también la búsqueda tabú.

El algoritmo genético utilizado es el que se describe en [10], escrito en java utilizando la librería JCLEC[3] [26], que ha sido reprogramado en C sobre Linux Suse 9.2, con algunas mejoras en su diseño. Estas mejoras están basadas en la nueva interpretación del individuo que se ha descrito al hablar de la representación del individuo.

El algoritmo híbrido se ha probado para problemas con 50 y 200 nodos.

IV. EXPERIMENTOS REALIZADOS

Las pruebas se realizaron en un ordenador con procesador AMD XP 2.4 GHz y 512 MB de RAM.

Los datos utilizados para probar el algoritmo, se pueden encontrar en internet[4].

Éstos son datos reales correspondientes al *Servicio Postal Australiano* y constan de una matriz de 200 nodos y 8 hubs. En el mismo fichero podemos encontrar también un pequeño programa en C que nos permite hacer subproblemas, con diferentes tamaños y número de hubs, partiendo del conjunto matriz. También nos proporcionan soluciones óptimas para varias combinaciones de ambos parámetros, en las cuales nos hemos basado para probar la eficiencia de nuestro algoritmo.

Para el afinamiento de los algoritmos híbrido y de búsqueda tabú hemos realizado diversos análisis de varianza que nos han ayudado a encontrar los valores más adecuados para optimizarlos.

Estos análisis han consistido en contrastes de Levene sobre igualdad de varianzas, pruebas de los factores inter-sujetos y tests de comparaciones múltiples de Tamhane realizados con el paquete estadístico SPSS v13.0.

Una vez recopilados los resultados, obtuvimos que para resolver los problemas de 50 nodos, el tamaño ideal de la *lista tabú* de nuestro algoritmo era 60 y que los parámetros *élite-plus* eran 188-57. Estos valores han sido los utilizados para afrontar el problema (200,8) en las pruebas realizadas tanto con el algoritmo híbrido como con el de búsqueda tabú.

Los algoritmos tabú y genético han sido probados para los problemas (20, 3)[5], (20, 4), (20,5), (25, 3), (25, 4), (25, 5), (50, 3), (50, 4) y (50, 5) y para el (200,8). El algoritmo híbrido sólo se ha probado para los casos con 50 y 200 nodos.

Para facilitar la lectura de las tablas debe tenerse en cuenta que **A.** significa algoritmo, **B.T.** búsqueda tabú, **G.C.** genético en C, **G.J.** genético en Java y **H.** híbrido.

La tabla I muestra los aciertos obtenidos sobre 30 ejecuciones. La primea columna nos presenta la complejidad del problema. La segunda el número de veces que se alcanzó el óptimo en esas ejecuciones con el algoritmo de búsqueda tabú. La tercera los aciertos para el algoritmo genético programado en C. La cuarta columna muestra los resultados obtenidos en las 30 ejecuciones con el algoritmo genético de [10].

TABLA I Número de aciertos hasta 25 nodos

Problema	**A.B.T.**	**A.G.C.**	**A.G.J.**
(20,3)	**30**	25	13
(20,4)	**23**	6	2
(20,5)	**15**	6	2
(25,3)	**30**	15	10
(25,4)	**19**	1	2
(25,5)	**23**	6	7

TABLA II Número de aciertos con 50 nodos

Problema	**A.H.**	**A.B.T.**	**A.G.C.**	**A.G.J.**
(50,3)	**30**	29	14	6
(50,4)	**25**	**25**	**25**	17
(50,5)	**11**	4	0	2

La tabla II muestra los resultados para los problemas con 50 nodos. En esta tabla se han añadido en la segunda columna los resultados del algoritmo híbrido, que sólo se probó para estos problemas. Como se observa el híbrido consigue al menos las mismas veces el óptimo que los demás. Asimismo se puede ver como el algoritmo de búsqueda tabú supera a los genéticos.

[3] *JCLEC* (Java Class Library for Evolution Computation) es una biblioteca de clases *Java* para Computación Evolutiva desarrollada por el grupo de investigación Aprendizaje y Redes Neuronales (AYRNA) de la Universidad de Córdoba.

[4] http://www.ms.ic.ac.uk/jeb/orlib/phubinfo.html.

[5] En cada pareja de números el primero indica números de nodos del problema y el segundo el número de hubs.

En la tabla III y en la tabla IV se pueden ver los errores medios obtenidos en los 30 ensayos realizados para cada uno de los problemas. Para calcular el error se obtiene la diferencia entre el valor obtenido y el óptimo conocido. Posteriormente se divide entre el óptimo y todo se multiplica por 100 para expresarlo en porcentaje.

error=100((valor_obtenido–valor_optimo)/valor_optimo)*

TABLA III Errores medios hasta 25 nodos

Problema	Error Medio A.B.T.	Error Medio A.G.C.	Error Medio A.G.J.
(20,3)	**0.0000**	0.6000	0.0100
(20,4)	**0.1500**	2.1300	2.8900
(20,5)	**0.4000**	1.5300	2.0900
(25,3)	**0.0000**	1.4000	1.9000
(25,4)	**0.0004**	0.0017	0.0174
(25,5)	**0.0031**	0.0220	0.0330

Como puede observarse el algoritmo de búsqueda tabú supera a los genéticos en todos los casos. También puede verse como el algoritmo híbrido supera al tabú en los casos en los que se experimentó.

TABLA IV Errores medios con 50 nodos

Probl.	Error Medio A.H.	Error Medio A.B.T.	Error Medio A.G.C.	Error Medio A.G.J.
(50,3)	**0.0000**	0.0030	1.0300	2.2100
(50,4)	**0.1300**	0.3000	0.5000	2.1400
(50,5)	**1.5200**	2.0800	4.1700	4.2000

TABLA V Tiempo en segundos hasta 25 nodos

Problema	A.B.T.	A.G.C.
(20,3)	**1,67**	2,52
(20,4)	**8,32**	20,23
(20,5)	**20,5**	40,36
(25,3)	**1,19**	7,74
(25,4)	**24,24**	29,16
(25,5)	**31,09**	49,48

La tabla V refleja la duración media en segundos de las 30 ejecuciones de los algoritmos de búsqueda tabú y del genético realizado en C. En estos problemas el tabú obtiene mejores tiempos que el genético en todos los casos.

La tabla VI muestra la duración de los algoritmos híbrido y tabú para los problemas con 50 nodos. Como se puede ver el algoritmo híbrido consume menos tiempo en los problemas mayores. Solamente en el más pequeño de estos problemas el tabú se realiza en un tiempo menor.

TABLA VI Tiempo en segundos con 50 nodos

Problema	A.H.	A.B.T.
(50,3)	125,96	**74,34**
(50,4)	**202,06**	221,69
(50,5)	**535,51**	684,63

Para el problema (200,8) no hemos encontrado el óptimo en la literatura que hemos consultado. En nuestros ensayos hemos obtenido diversos valores con los algoritmos de búsqueda tabú y con el híbrido que se pueden ver en la tabla VII.

Solamente hemos encontrado un valor para el problema (200,8) y está en el trabajo que realizamos en [9]. Ese valor es 125370,12, que es algo peor que el encontrado por el algoritmo híbrido.

TABLA VII Valores con 200 nodos y 8 hubs

Problema	Óptimo	Valor A.H.	Valor A.B.T.
(200,8)	?	**123856,33**	133976,89

V. CONCLUSIONES

En este trabajo hemos presentado varios algoritmos que resuelven el problema del p-hub con asignación simple y sin restricciones de capacidad. Para ello hemos partido de las ideas que desarrollamos en [10].

En aquel trabajo realizamos un algoritmo genético que hemos refinado y reprogramado en C, consiguiendo mejorar el anterior programado en java. Las mejoras del nuevo algoritmo genético han sido tanto en tiempo de ejecución como en resultados.

Hemos realizado un algoritmo de búsqueda tabú y un algoritmo híbrido tabú-genético. Para ello hemos sometido a los algoritmos a numerosas pruebas estadísticas que nos han permitido elegir los parámetros adecuados para su desarrollo.

En definitiva podemos señalar que hemos conseguido:

- Realizar tres algoritmos que proporcionan soluciones de calidad para el problema del p-hub con asignación simple y sin restricciones de capacidad, en un tiempo razonable.

- Mejorar las soluciones del algoritmo genético [10], sobre el que habíamos trabajado anteriormente para resolver este problema.

- Encontrar una solución a un problema de extraordinaria dificultad como es cuando el número de nodos es 200 y el de hubs 8.

- Obtener un algoritmo híbrido tabú-genético que mejora al tabú y al genético.

Estamos trabajando para mejor el algoritmo híbrido introduciendo nuevos elementos que lo mejoren. Unas de las líneas que estamos empezando es la programación paralela en un cluster de 8 hubs mediante programación con la librería MPI que puede obtenerse en internet[6].

REFERENCIAS

[1] S. Abdinnour-Helm. A hybrid heuristic for the uncapacited hub location problem. European Journal of Operational Research 106, pp.489-499. 1998,

[2] T. Aykin . The hub location and routing problem. European Journal of Operational Research 83, pp. 200-219. 1995.

[3] M. L. Brandeau and Samuel S. Chiu. An Overview of Representative Problems in Location Research. Management Science 35, pp. 645-674. 1989.

[4] J. F. Campbell. Integer Programming formulations of Discrete Hub Location Problems. European Journal of Operational Research 72, pp. 387-405. 1994.

[5] J. Ebery. Solving large single allocation p-hub problems with two or three hubs. European Journal of Operational Research 128, pp. 447-458. 2001.

[6] J. Ebery and M. Krishnammorthy, A.T. Ernst and N. Boland. The capacited multiple allocation hub location problem: Formulations and algorithms. European Journal of Operational Research 120, pp. 614-631. 2000.

[7] A. T. Ernst and M. Krishnamoorthy. Efficient algorithms for the uncapacitated single allocation p-hub median problem. Location Science 4, pp 139-154. 1996.

[8] F. Glover and M. Laguna. Tabu Search. Kluwer Academic Publishers 382pp. 1997.

[9] P. Jiménez, R. Castro, J. Mayoral y C. Hervás. Búsqueda Dispersa para el problema p-hub sin restricciones de capacidad. Actas del IV congreso español sobre metaheurísticas, algoritmos evolutivos y bioinspirados Maeb05, pp. 863-870. 2005.

[10] P. Jiménez, A. Barbancho y C. Hervás. Soluciones al problema p-hub usando algoritmos genéticos. Actas del II congreso español sobre metaheurísticas, algoritmos evolutivos y bioinspirados Maeb03, pp. 586-593. 2003.

[11] J. G. Klincewicz . A dual algorithm for the uncapacitated hub location problem. Location Science, 4(3):173-184. 1996.

[12] J.G. Klincewicz. A voiding local optima in the p-hub location problem using tabu search and grasp. Annals of Operations Research 40, pp. 283-302. 1992.

[13] J.G. Klincewicz. Heuristics for the p-hub location problem. European Journal of Operational Research 53, pp. 25-37. 1991.

[14] M.J. Kuby and R.G. Gray. The hub network design problem with stopover and feeders: The case of Federal Express. Transportation Research 27, pp. 1-12. 1993.

[15] M.E. O'kelly and D.L. Bryan, J. Skorin-Kapov and D. Skorin-kapov. Hub network design with single and multiple allocation: A computational study. Location Science 4, pp. 125-138. 1996.

[16] M.E. O'Kelly. A cuadratic integer program for the location of interacting hub facilities. European Journal of Operational Research 32, pp. 393-404. 1987.

[17] M.E. O'kelly. The location of interacting hub facilities. Transportation Science 20, pp. 92-106. 1986.

[18] M. Pérez, F. Almeida and J. M. Moreno-Vega. Genetic algorithm with multistart search for the p-hub median problem. Proceedings of the Euromicro Conference, pp. 702-707. 1998.

[19] S. Pierre and F. Houéto. A tabu search approach for assigning cells switches in cellular mobile networks. Computer Communications 25, pp. 464-477. 2002.

[20] J. Prawda. Métodos y modelos de investigación de operaciones, vol II. Editorial LIMUSA. 1981.

[21] J. Sarkis and R.P. Sundarraj . Hub location at Digital Equipment Corporation: A comprehensive analysis of qualitative and quantitative factors.European Journal of Operational Research 137, pp. 502-509. 2002.

[22] J. Schaffer, R. Caruana, L. Eshelman, and R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In J. Schaffer, editor, 3rd International Conference on Genetic Algorithms, pp. 51-60, San Mateo, CA. Morgan Kaufmann. 1989.

[23] D. Skorin-kapov, J. Skorin-Kapov and M. O'kelly. Tight linear programing relaxations of uncapacited p-hub median problem. European Journal of Operational Research 94, pp. 582-593. 1996.

[24] J. Skorin-Kapov and D. Skorin-kapov. On tabu search for the location of interacting hub facilities. European Journal of Operational Research., 73, pp. 502-509. 1994.

[25] K Smith, M. Krishnammorthy and M. Alaniswami. Neural Versus Traditional approaches to the location of interacting hub facilities.Location Science 4, pp. 155-171. 1996.

[26] S. Ventura, C. Hervas y D. Ortiz. Jclec: Una biblioteca de clases en java. Primer congreso iberoamericano de algoritmos evolutivos y bioinspirados. *AEB'02*. 2002.

[6] Message passing interface forum (www-unix.mcs.anl.gov/mpi/mpich/index.html). MPI: A message-passing interface standard. Ultima visita sep 2006