Métodos Multi-arranque

Rafael Martí *
Departamento de Estadística e Investigación Operativa
Facultad de Matemáticas
Universidad de Valencia
Dr. Moliner 50 46100 VALENCIA
rmarti@uv.es

J. Marcos Moreno Vega **
Departamento de Estadística, I.O. y Computación
Centro Superior de Informática
Universidad de La Laguna
Avda. Astrofísico Francisco Sánchez s/n 38271 SANTA CRUZ DE TENERIFE
jmmoreno@ull.es

Resumen

En los Métodos Multi-arranque se alternan una fase de generación de soluciones con otra de mejora de las mismas. El proceso se repite hasta que se cumpla un criterio de parada. Este esquema sencillo suministra un procedimiento que combina adecuadamente el poder de exploración de los métodos de construcción con el poder de explotación de los métodos de mejora. En el presente trabajo se enumeran las principales aportaciones y éxitos de los Métodos Multi-arranque, tanto para problemas combinatorios como para problemas de optimización global. Además, se describe un Método Multi-arranque para el problema del ancho de banda.

1. Introducción

El principal inconveniente de las Búsquedas Locales es que, en general, suministran soluciones localmente óptimas que pueden estar muy alejadas (en términos de valor objetivo) de la solución o soluciones óptimas globales. Una alternativa para solventar este inconveniente consiste en aplicar Búsquedas Locales desde varias soluciones de partida. La repetición de los procesos *Generar Solución Inicial* y *Búsqueda Local* constituye el primer Método Multi-arranque descrito en la literatura [4] [20]. Este esquema puede generalizarse para contemplar diferentes Métodos Multi-arranque que consisten en aplicar reiteradamente un optimizador o método de búsqueda desde diferentes soluciones iniciales.

La Figura 1 muestra el esquema general de un Método Multi-arranque. Típicamente podemos hablar de dos fases en cada paso i. En la primera, se construye una solución x_i y en la segunda se trata de mejorar mediante la aplicación de un método de búsqueda, obteniendo la solución $x_i^{'}$ (que eventualmente puede ser

*El primer autor ha sido subvencionado parcialmente por el Ministerio de Ciencia y Tecnología a través del proyecto TIC2000-1750-C06-01

**El segundo autor ha sido subvencionado parcialmente por el Ministerio de Ciencia y Tecnología a través del proyecto TIC2002-04242-C03-01. El 70 % de la ayuda de dicho proyecto proviene del Fondo Europeo de Desarrollo Regional (FEDER).

igual a x_i).

Hacer $i = 1$
while(Condición de Parada)
    Fase 1 (Generación)
        Construir solución x_i
    Fase 2 (Búsqueda)
        Aplicar método de búsqueda
            para mejorar x_i
        Sea $x_i^{'}$ la solución obtenida
    if ($x_i^{'}$ mejora a x_i)
        Actualizar la mejor solución
    $i = i + 1$

Figura 1: Método Multi-arranque

En algunas aplicaciones, la fase 1 se limita a la simple generación aleatoria de las soluciones, mientras que en otros ejemplos se emplean sofisticados métodos de construcción que consideran las características del problema de optimización para obtener soluciones iniciales de calidad. Algo similar ocurre con el método de búsqueda de la fase 2. Podemos encontrar algoritmos de *Descenso Simple* (Búsquedas Locales) que, a partir de la solución inicial, conducen al óptimo local más cercano mediante una serie de movimientos de mejora, o elaborados procedimientos metaheurísticos que realizan una búsqueda *inteligente* del espacio de soluciones y tratan de alcanzar la solución óptima del problema, evitando quedar atrapados en un óptimo local de baja calidad. En cuanto a la condición de parada, se han propuesto desde criterios simples, como el de parar después de un número dado de iteraciones, hasta criterios que analizan la evolución de la búsqueda y aseguran, en muchos casos, la convergencia asintótica al óptimo global del problema. Otra cuestión a considerar es si el método de búsqueda de la segunda fase debe aplicarse a todos los puntos generados o sólo a un subconjunto de dichos puntos. La combinación de todos estos elementos (construcción de soluciones iniciales, optimizador o método de mejora, criterio de parada, ...) suministra una gran variedad de procedimientos híbridos basados en el esquema Multi-arranque.

Martí [21] hace una revisión de los métodos multi-arranque y propone una clasificación basada en el uso o no de memoria, y en el grado de reconstrucción de la solución de inicio. Schoen [29] hace una revisión personal de tales métodos, dando una definición formal de los mismos y subrayando las características propias de estos métodos.

Los métodos multi-arranque han sido utilizados, principalmente, para resolver dos tipos de problemas de optimización: los problemas no lineales (que denominaremos optimización global) y los problemas combinatorios. En el campo de los problemas no lineales, se ha desarrollado principalmente la teoría sobre las condiciones de convergencia del método al óptimo global del problema. En el segundo caso se encuentran una gran cantidad de procedimientos heurísticos o aproximados para encontrar buenas soluciones en tiempos de computación relativamente pequeños.

En general, en los métodos multi-arranque la fase de mejora o búsqueda suele consistir en una búsqueda local que se aplica a partir de la solución generada. En problemas no lineales, la búsqueda local suele basarse en técnicas dependientes del gradiente que finalizan en un óptimo local del problema.

Para problemas combinatorios pueden diseñarse diferentes búsqueda locales. El éxito de éstas depende de diferentes factores, tales como la estructura de entorno, la estrategia empleada para examinar éste, la eficiencia en la evaluación de las soluciones y la solución inicial. La fase de construcción juega un papel fundamental respecto a proporcionar un punto inicial de calidad en la búsqueda. Entornos relativamente sencillos basados en intercambios o inserciones son los más comúnmente utilizados. Las dos estrategias típicas de selección de la solución en el entorno (movimiento) son la *best-improving* y la *first-improving*. En la primera se examina todo el entorno para determinar la mejor solución a la que moverse; en la segunda, en el momento en que se encuentra una solución que mejore a la actual, se finaliza la exploración y se realiza el movimiento sin explorar el resto del entorno. La mayor parte de la experimentación realizada indica que una de las claves de los métodos multi-arranque reside en un balance adecuado entre las fases de generación y búsqueda. Esto no es más que poner en otros términos el conocido principio de que el éxito de un procedimiento meta-heurístico reside en la correcta interacción entre la intensificación, o exploración en profundidad de una cierta área, y la diversificación, o muestreo inteligente del espacio de soluciones.

En lo que sigue se enumeran algunas de las alternativas propuestas para los diferentes elementos que definen un Método Multi-arranque. Para ello, se realiza una revisión de los trabajos más relevante publicados hasta la fecha. Se distingue entre resultados y alternativas para problemas de optimización global y para problemas combinatorios.

2. Generación de soluciones

2.1. Optimización global

El algoritmo multi-arranque más básico que podemos considerar es aquél en el que las soluciones iniciales se generan al azar en la región factible del problema, y la fase de búsqueda se realiza mediante algún procedimiento de búsqueda local B desde cada uno de los puntos generados. Este método converge al óptimo global del problema con probabilidad uno cuando el número de puntos generados tiende a infinito. Sin embargo, el procedimiento es muy ineficiente puesto que se pueden generar muchos puntos cercanos entre sí, de modo que al aplicarles el procedimiento de búsqueda B se obtenga repetidamente el mismo óptimo local.

Por ello, se han propuesto diversas alternativas a este esquema. Algunas de ellas afectan a la forma en que se generan las soluciones inciales y otras, una vez que se ha generado una solución inicial, determinan si se aplica o no el método de mejora a ésta.

Solis y Wets [30] estudian la convergencia de los métodos multi-arranque en los que, la probabilidad de selección de la siguiente solución inicial a la que aplicarle el método de búsqueda, depende de la situación actual del procedimiento. Algunas extensiones de estos métodos tratan de reducir el número de búsquedas locales y aumentar la probabilidad de considerar puntos cercanos al óptimo global del problema (Mayne y Meewella [23]).

El método conocido como *Multi-Level Single Linkage* (MLSL) debido a Rinnooy Kan y Timmer [28] soluciona este problema manteniendo la convergencia del método. Se genera una muestra aleatoria de n puntos en el espacio de soluciones. Se evalúan los puntos y se ordenan de acuerdo a su valor, seleccionando una proporción qn del total ($0 \leq q \leq 1$). El procedimiento de búsqueda B se aplica, siguiendo el orden, a los puntos seleccionados. Se descartan, y por lo tanto no se les aplica el método B, los puntos que:

1. Estén muy cerca de otro punto al que previamente se le aplicó el método B.
2. Estén muy cerca de la frontera del espacio de soluciones.
3. Estén muy cerca de un óptimo local obtenido anteriormente.

Una vez aplicado el método B a los puntos seleccionados que cumplen estas condiciones, se generan nuevamente n puntos o soluciones al azar y se vuelve a aplicar el mismo procedimiento al conjunto resultante de unir éstos con los generados en iteraciones anteriores. La distancia crítica para considerar que un punto es muy cercano a otro se actualiza en cada iteración, disminuyendo su valor. Los autores prueban que, si el muestreo continúa indefinidamente, se obtendrán todos los óptimos locales de la función, pero el número de búsquedas locales será finito con probabilidad 1. Además, se propone un criterio de parada en términos de probabilidades. Recientemente, Fylstra et al. han propuesto una versión del método MLSL para el caso de problemas restringidos, aunque no generaliza las propiedades de convergencia y ha de considerarse un método heurístico.

Uno de los inconvenientes del método MLSL es que al ir disminuyendo la distancia crítica según aumentan las iteraciones, se han de guardar todos los puntos generados, ya que los rechazados en una iteración, podrían ser aceptados en iteraciones posteriores. Es decir, cada vez que se reduce la distancia crítica, se han de revisar todos los puntos considerados anteriormente. Locatelli y Schoen [19] introducen el método *Random Linkage* (RL) que mantiene las propiedades de convergencia del MLSL y no requiere de la revisión reiterada de las soluciones. A diferencia del método anterior, en éste los puntos generados al azar se consideran uno a uno. Cada vez que se genera un punto o solución, se aplica el optimizador B según una función de probabilidad no-decreciente $p(d)$ donde d es la distancia al punto más cercano generado anteriormente.

Byrd et al. [9] proponen una versión paralela del MLSL. Se trata de un algoritmo concurrente en el que tres de las fases del MLSL (generación de soluciones, selección de aquellas soluciones de la fase anterior a las que aplicar el método de mejora y aplicación del método de mejora) se implementan de forma paralela.

Hickernell y Yuan [16] presentan un algoritmo multi-arranque basado en muestras *quasi-aleatorias*. Estas muestras están formadas por conjuntos de soluciones deterministas, al contrario que las aleatorias, distribuidas uniformemente sobre el conjunto de soluciones del problema. El método consiste en aplicar un algoritmo rápido de búsqueda local (*descenso greedy*) sobre un conjunto de soluciones quasi-aleatorias para concentrar la búsqueda. A continuación, se reduce la muestra reemplazando la peor solución con nuevas soluciones. Cuando una solución permanece en el conjunto durante un cierto número de iteraciones, se le aplica el algoritmo de búsqueda local en una versión extendida. El método termina cuando, tras un determinado número de iteraciones, no se encuentra ningún mínimo local nuevo. Los autores comparan computacionalmente el método con otros algoritmos de búsqueda local, mostrando su efectividad.

Hu et al. [18] estudian la combinación de la búsqueda lineal dada por el gradiente con el método multi-arranque en el que las soluciones iniciales se generan al azar. Los autores consideran tanto modelos continuos como discretos y abordan diferentes aspectos como el procesamiento en paralelo, la distribución de probabilidad asociada al multi-arranque y el número de soluciones iniciales generadas. Entre las conclusiones podemos destacar que la distribución uniforme parece la más adecuada para este procedimiento.

Ugray et al. [32] proponen un esquema multi-arranque que filtra las soluciones, x_i, que han sido generadas al azar, para realizar una búsqueda selectiva. Contemplan dos tipos de filtrado

> *Filtro de diversidad.* Este filtro garantiza que las soluciones a las que se aplica el método de mejora sean diversas, en el sentido de que no estén cerca de ningún óptimo local previamente encontrado. Su objetivo es no aplicar más de una vez el método de mejora desde soluciones que pertenecen al mismo *cono de atracción (región de atracción)* de un óptimo local. [1]
>
> Todas las soluciones (óptimos locales) $x_i^{'}$ que se obtienen al aplicar el método de mejora, se almacenan en una lista. Sea $maxdist$ el máximo de las distancias euclídeas entre todas las soluciones de la lista. Cada vez que se genera una solución aleatoria x_i, se calcula su distancia a todos los óptimos locales almacenados. Si la distancia entre x_i y alguna de las $x_i^{'}$ es menor que $distfactor \cdot maxdist$, entonces se descarta la solución x_i y no se le aplica el algoritmo de mejora.
>
> Este filtro asume que el cono de atracción de un óptimo local es esférico y de radio al menos $maxdist$. El parámetro $distfactor$ controla la intensidad del filtro. Si tiene un valor cercano a 0, el filtro apenas actúa. Por el contrario, si es mayor que 1 la aplicación del método de mejora se realizará en muy pocas ocasiones. Tras realizar unas pruebas fijó $distfactor = 0{,}75$.
>
> *Filtro de calidad.* Este filtro garantiza el que sólo se aplique el algoritmo de mejora a soluciones relativamente buenas, rechazando a priori soluciones con un valor mayor que un umbral UB. Este umbral se inicializa al valor objetivo de la primera solución generada, x_1.
>
> Si durante $maxr$ iteraciones consecutivas, las soluciones han sido rechazadas por este filtro, entonces se aumenta el umbral mediante la expresión:
>
> $$UB = UB + \delta(1+ \mid UB \mid)$$
>
> Por otro lado, cuando una solución pasa el filtro (su valor es inferior a UB), entonces se actualiza UB haciéndolo igual al valor objetivo de dicha solución. De esta forma el valor de UB se adapta a los valores encontrados durante la búsqueda, endureciendo o relajando el filtro en función del número de rechazos. Tras realizar unas pruebas el parámetro δ se fijó en 0,2.

[1] El *cono o región de atracción* de un óptimo local, $x_i^{'}$, está formado por aquellas soluciones desde las que una búsqueda local acabaría en $x_i^{'}$.

2.2. Optimización combinatoria

A partir de la experiencia en la resolución de problemas no lineales, los métodos multi-arranque se han aplicado también a la resolución de problemas de optimización combinatoria. En la mayor parte de los casos, estos métodos no presentan propiedades de convergencia al óptimo global del problema, y su aplicación se limita al campo de los algoritmos heurísticos o aproximados. Dada su sencillez de implementación y su facilidad para combinarse con otras técnicas de resolución, podemos encontrar estos métodos, por sí solos o combinados, en multitud de problemas. En esta sección vamos a comentar algunos de ellos.

Boese et al. [8] analizan la relación entre los óptimos locales de un problema al tratar de determinar el mejor de todos ellos. Basado en los resultados de ese estudio, proponen un método multi-arranque llamado *Adaptive Multi-Start* (AMS), en el que los puntos iniciales se generan a partir de los mejores óptimos locales encontrados. En un primer paso, AMS genera r soluciones al azar y les aplica a todas ellas un procedimiento de búsqueda local *greedy* para determinar el conjunto inicial de óptimos locales. En el segundo paso (*adaptive*) se construyen las soluciones iniciales a partir del conjunto de óptimos locales. A estas soluciones iniciales se les aplica varias veces el método de mejora. Los autores prueban el método en la resolución del problema del viajante de comercio, y muestran que mejora significativamente a implementaciones previas de métodos multi-arranque.

Hagen y Kang [13] proponen un método multi-arranque de tipo AMS para el problema de partición VLSI donde el objetivo es minimizar el número de señales que circulan entre componentes. El método tiene dos fases, en la primera se genera un conjunto soluciones aleatorias y se les aplica a todas ellas un algoritmo de búsqueda local, obteniendo un conjunto de óptimos locales. En la segunda parte se construyen soluciones iniciales como los puntos centrales de los mejores óptimos locales conocidos. Con el objetivo de reducir el tamaño del problema a resolver, se añade una fase de preproceso basada en técnicas cluster. Un estudio empírico permite establecer la superioridad del método propuesto frente a algoritmos previos para este problema.

Ulder et al. [31] combinan los algoritmos genéticos con las estrategias de búsqueda local obteniendo un procedimiento que mejora a los algoritmos genéticos puros en la resolución del problema del viajante. Cada individuo o solución es mejorado antes y después del proceso de combinación para formar una nueva solución (*offspring*). En esencia, los autores proponen un procedimiento *memético* que incorpora nuevos elementos al esquema de los algoritmos genéticos clásicos.

El uso de la técnica multistart, consistente en aplicar repetidamente un método heurístico relativamente simple, es una forma sencilla y eficaz de obtener un método que mejora al heurístico considerado. Aunque en ocasiones esta mejora sea sólo marginal y el resultado sea un método que proporciona soluciones de calidad media, es un procedimiento utilizado en muchos trabajos como referencia para medir la bondad de un método heurístico o meta-heurístico nuevo. Baluja [2] compara diferentes algoritmos genéticos en la resolución de seis problemas diferentes: el problema del viajante de comercio, el problema de secuenciación job-shop, el problema de la mochila, el problema del empaquetado, el entrenamiento de redes neuronales y la optimización numérica. En este trabajo se utiliza un método multi-arranque denominado *Multiple Restart Stochastic Hill-climbing* (MRSH) como un primer método básico, a partir del cual medir la contribución relativa de los algoritmos genéticos considerados. Otras comparativas entre métodos multi-arranque MRSH y algoritmos genéticos pueden encontrarse en Ackley [1], Wattenberg y Juels [33] y Houck et al. [17]).

Uno de los métodos multi-arranque más aplicados actualmente es el denominado GRASP, debido a Feo y Resende [11] [12]. Como el resto de métodos multi-arranque, GRASP consta de dos fases. La construcción o generación y la mejora o búsqueda. En cada iteración de la fase constructiva, GRASP mantiene un conjunto de elementos candidatos que pueden ser añadidos a la solución parcial que se está construyendo. Todos los elementos candidatos se evalúan usando una función que mide su atractivo. En lugar de seleccionar el mejor de todos los elementos, se construye la lista restringida de candidatos RCL (*restricted candidate list*) con los mejores según una cantidad establecida (esta es la parte *greedy* del método). El elemento que finalmente se añade a la solución parcial actual, se escoge al azar del conjunto RCL (esta es la parte probabilística). Entonces se recalcula la lista de elementos candidatos y se realiza una nueva iteración (esta es la parte *que se adapta* en el método). Estos pasos se reiteran hasta que se obtiene una solución del problema. A ésta se le aplica el método de mejora. A continuación, se repiten la fase constructiva y de mejora hasta que se cumpla el criterio de parada.

Existen diferentes variantes de este esquema entre las que podemos destacar el método heurístico conocido como semi-greedy debido a Hart y Shogan [14]. Este método sigue el esquema multi-arranque basado en aleatorizar una evaluación greedy en la construcción, pero no tiene una fase de búsqueda local o mejora. Actualmente se están implementando versiones de este método denominadas *Reactive GRASP* en donde el ajuste de los parámetros necesarios (básicamente los que determinan la RCL) se realizan de forma dinámica según el estado de la búsqueda [27]

Melián et al. [25] usan información sobre las soluciones de inicio y sobre los óptimos locales encontrados para dirigir la búsqueda. La técnica se diseña para problemas combinatorios en los que se desea encontrar la mejor selección de un número dado de items desde un universo de items. Sea $RefSet$ el conjunto de óptimos locales encontrados hasta el momento. Para cada $x_i^{'}$ del conjunto $RefSet$, se conoce el porcentaje de búsquedas locales que, comenzando en soluciones a distancia menor o igual que k ($k = 1, \ldots$) de $x_i^{'}$, acaban en $x_i^{'}$. Sea $k(x_i^{'})$ el valor de k cuyo porcentaje asociado es mayor que α (parámetro fijado por el usuario), y considérese

$$k^* = \max_{x_i^{'} \in RefSet} k(x_i^{'})$$

Sea, además, $w(x_i^{'})$ el porcentaje de búsquedas locales que acaban en $x_i^{'}$, y, para cada item u,

$$w(u) = \sum_{u \in x_i^{'} \in RefSet} w(x_i^{'}),$$

la suma de los porcentajes de aquellos óptimos locales que incluyen a u como parte de la solución.

Los anteriores valores se emplean para diversificar e intensificar la búsqueda. Para diversificar la misma, se desarrollan búsquedas locales desde soluciones a distancia mayor que k^* de un óptimo local seleccionado aleatoriamente de $RefSet$. Para intensificar la búsqueda, se construye aleatoriamente una solución en la que la probabilidad de incluir un elemento u en la misma es proporcional a los valores $w(u)$.

3. Reglas de parada

Como ocurre con cualquier método heurístico para resolver un problema, uno de los elementos más difíciles de fijar en un Método Multi-arranque es el criterio de parada. Los principales criterios de parada propuestos analizan tres variables aleatorias: valores objetivos de los mínimos locales, número de mínimos locales distintos de la función objetivo y número de iteraciones necesarias para alcanzar el mínimo global.

Los y Lardinois [20], en uno de los primeros trabajos dedicados a los Métodos Multi-arranque, obtienen reglas de parada a partir de la función de distribución asintótica del mínimo global. El Teorema de Fisher y Tipper establece que esta distribución debe ser una de las tres distribuciones de Gumbel: tipo I, tipo II o tipo III. En particular, se trata de la distribución asintótica tipo III o de Weibull. De esta forma, se pueden obtener estimaciones puntuales e intervalos de confianza para el óptimo global, y reglas de parada para el método. El trabajo analiza varias metodologías para obtener las anteriores estimaciones e intervalos de confianza y compara experimentalmente las diferentes reglas de parada que, consecuentemente, se obtienen.

Usando un esquema Bayesiano, Betró y Schoen [3] asumen una distribución a priori sobre los valores objetivos de los óptimos locales encontrados, y utilizan ésta para obtener reglas de parada.

Si el número de mínimos locales, κ, de la función objetivo fuese conocido, un criterio de parada obvio sería desarrollar búsquedas locales hasta encontrarlos todos. Sin embargo, este valor es desconocido. No obstante, el número de veces que aparece cada uno de los mínimos encontrados al aplicar las búsquedas locales suministra información sobre κ y sobre el tamaño de las correspondientes regiones de atracción. Boender y Rinnooy Kan [7] realizan un estudio detallado de estos parámetros siguiendo la metodología bayesiana. En ésta se supone que los parámetros son variables aleatorias para las que se asume una distribución a priori conocida que luego se modifica por las evidencias muestrales. Así, obtienen, inicialmente, dos reglas de parada: parar cuando se haya encontrado un número de mínimos locales distintos mayor o igual que el estimador bayesiano entero óptimo de κ; parar cuando el estimador del tamaño relativo de las regiones de atracción muestreadas sea suficientemente grande.

Las reglas obtenidas por Boender y Rinnooy Kan no tienen en cuenta el coste que supone desarrollar nuevas búsquedas locales. Una metodología alternativa para obtener reglas de parada consiste en suponer que cada vez que se finaliza un método multi-arranque se incurre en dos pérdidas: una *pérdida de finalización*, que depende del coste que supone finalizar la búsqueda antes de encontrar el mínimo global, y una *pérdida de ejecución*, que depende del coste de realizar nuevas búsquedas locales. Boender y Zielinski [5] y Boender y Rinnooy Kan [6][7] consideran tres estructuras de pérdida según el esquema anterior y, para cada una de ellas, obtienen reglas de parada.

Moreno et al. [24] proponen una regla de parada para el método multi-arranque basada en el estudio estadístico del número de iteraciones necesarias para encontrar el óptimo global. Los autores introducen dos variables aleatorias cuya combinación proporciona el número de iteraciones necesarias hasta encontrar el óptimo global (variable ψ). Estas variables son: el número de soluciones iniciales generadas hasta que la correspondiente búsqueda local alcanza el óptimo global y el número de evaluaciones de la función objetivo hasta que la correspondiente búsqueda local alcanza un óptimo local. Si bien la distribución exacta de la variable que suministra el número de iteraciones necesarias para encontrar el óptimo global es difícil de obtener, si puede aproximarse apropiadamente usando la distribución normal. El criterio de parada propuesto consiste en finalizar la búsqueda después de e iteraciones siempre que la probabilidad de que ψ sea menor que e sea suficientemente grande. Por tanto, la regla de parada propuesta es, tras cada aplicación de una búsqueda local, verificar si la anterior condición es cierta o no. Si la respuesta es afirmativa, el algoritmo finaliza; en caso contrario, se selecciona una nueva solución desde la que aplicar una búsqueda local.

Hart [15] describe diversas reglas de parada secuenciales para la Búsqueda Aleatoria Pura que se basan en la estimación del óptimo global de una función. A continuación, las modifica y generaliza para otros algoritmos secuenciales, y describe como pueden usarse en un multi-arranque. De la experiencia computacional desarrollada concluye que estas reglas de parada se comportan de forma similar a las reglas de parada bayesianas propuestas por [3][7]. Además, son, a juicio del autor, más sencillas y fáciles de usar.

4. Multi-arranque para el problema del ancho de banda

Para ilustrar el método multi-arranque, en lo que sigue se describen los diferentes elementos de este método para el problema del ancho de banda (*Bandwidth Reduction Problem, BRP*). El BRP es un problema combinatorio $NP - hard$ con numerosas aplicaciones. Entre otras, la del preproceso de matrices para la resolución de sistemas lineales y la de la ubicacin en memoria de los documentos de una página web.

El problema ha sido estudiado por numerosos autores, desde la propuesta en 1969 del algoritmo de Cuthill-McKee [10] hasta la reciente aplicación de diversos metaheurísticos (Martí et al. [22] o Piñana et al. [26]). En esta sección se propone un método multi-arranque con filtros para obtener buenas soluciones aproximadas al BRP. El objetivo no es competir con los algoritmos existentes para este problema, sino únicamente ilustrar algunas de las técnicas descritas en las secciones anteriores.

El BRP puede enunciarse tanto en términos de matrices como de grafos. Se considerará la segunda formulación por facilitar la descripción algorítmica. Dado un grafo $G = (V, E)$ con n vértices ($|V| = n$) y m aristas ($|E| = m$), una solución del problema consiste en una numeración (etiquetado) del conjunto de vértices. Así, una solución f, asigna los enteros $\{1, 2, \ldots, n\}$ a los vértices del grafo. El ancho de cada vértice v según el etiquetado f se calcula como:

$$B_f(v) = \max\{\mid f(v) - f(u) \mid : u \in N(v)\}$$

donde $N(v)$ es el conjunto de vértices adyacentes al vértice v. A partir del ancho de cada vértice se obtiene el ancho del grafo como:

$$B_f(G) = \max\{B_f(v) : v \in V\}$$

El BRP consiste en encontrar el etiquetado f del conjunto de vértices que minimiza el valor de $B_f(G)$. Notar que un etiquetado es una renumeracin y, por lo tanto, el conjunto de soluciones de este problema es el conjunto de permutaciones de n elementos.

Una forma simple y rápida de construir soluciones para el BRP consiste en generar permutaciones al azar. Sin embargo, la calidad de dichas soluciones será, sin duda, baja. Se ha de incorporar cierto conocimiento sobre las características del problema al método de construcción para obtener soluciones de calidad aceptable. Un proceso constructivo natural para este problema consiste en asignar, en cada iteración, una etiqueta a un vértice (también podríamos explorar la posibilidad *simétrica* de asignar un vértice a una etiqueta dada). Si llamamos U al conjunto de vértices no etiquetados, el algoritmo selecciona en cada paso un vértice de U, le asigna una etiqueta, y lo elimina de este conjunto. De esta forma, en n pasos se construye una solución. Piñana et al. [26] proponen diferentes métodos constructivos basados en el concepto de estructuras de nivel y la metodología GRASP. Aquí se considerará un método más sencillo que los propuestos por estos autores, aunque haciendo uso del concepto denominado etiqueta ideal que ellos introducen.

En la primera iteración, el algoritmo constructivo que se propone para el BRP escoge probabilísticamente un vértice usando una distribución de probabilidad que asigna mayor probabilidad a los vértices con mayor grado. Al vértice seleccionado se le asigna la etiqueta $\lfloor n/2 \rfloor$ y se elimina de U. En sucesivas iteraciones se selecciona probabilísticamente un vértice de U de acuerdo a una distribución que asigna mayor probabilidad a los vértices con mayor número de vértices adyacentes previamente etiquetados. De esta forma, se escoge para etiquetar, vértices que son adyacentes a un número grande de vértices ya etiquetados. Al vértice escogido se le asigna la etiqueta disponible más cercana a su etiqueta ideal, $mid(v, U)$. La etiqueta ideal es la mejor posible para ese vértice, considerando el valor de las etiquetas de sus adyacentes ya etiquetados. Las siguientes expresiones muestran el cálculo de la etiqueta ideal:

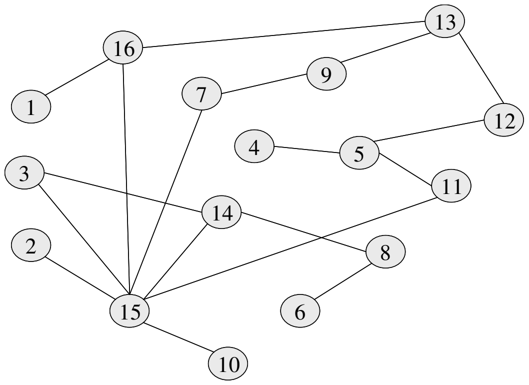

Figura 2: Ejemplo del BRP

$$\begin{aligned} \max(v, U) &= \max\{f(u) : \\ &\qquad u \in N(v) \cap (V - U)\}, \\ \min(v, U) &= \min\{f(u) : \\ &\qquad u \in N(v) \cap (V - U)\}, \\ mid(v, U) &= \left\lfloor \frac{\max(v, U) + \min(v, U)}{2} \right\rfloor \end{aligned}$$

Aunque el mecanismo de construcción propuesto es relativamente simple, proporciona soluciones de mejor calidad que la construcción aleatoria. Considérese el ejemplo de la figura 2 con 16 vértices y 18 aristas en el que el valor de la solución óptima es de 4. Se ha generado un conjunto de 30 soluciones al azar y otro con el generador propuesto. En el primer caso el promedio del ancho de la banda de las soluciones es de 13,56 mientras que en el segundo el promedio es de 9,36.

El cuadro 1 muestra las 10 primeras soluciones generadas con el método propuesto para el BRP. La columna BD muestra el valor del ancho de banda de cada solución. Si se aplica el filtro de calidad (descrito en la sección 2.1), y sólo se consideran las soluciones con valor por encima del umbral UB para aplicarles la fase de mejora, se tendría que, inicialmente, UB es

Sol	1	2	3	4	5	6	7	8	9	10
f(1)	4	16	9	1	12	13	16	1	11	11
f(2)	11	4	5	6	9	6	12	10	4	6
f(3)	12	9	4	3	5	4	7	9	9	7
f(4)	1	5	12	12	16	12	12	16	15	1
f(5)	2	7	11	10	15	12	13	14	14	2
f(6)	16	15	1	16	2	1	3	15	1	16
f(7)	8	2	16	9	7	10	2	2	7	4
f(8)	15	14	2	15	1	2	4	12	2	15
f(9)	7	1	15	12	11	11	1	3	10	13
f(10)	10	3	6	2	6	9	8	8	8	9
f(11)	13	6	10	8	10	7	10	13	16	3
f(12)	3	11	12	11	14	16	15	5	13	14
f(13)	6	13	13	14	13	15	11	4	12	12
f(14)	14	10	3	7	3	3	5	11	3	8
f(15)	9	8	8	5	4	5	6	7	5	5
f(16)	5	12	7	4	8	8	9	6	6	10
BD	11	12	8	10	6	7	10	9	11	12

Cuadro 1: Soluciones iniciales

Sol	1	2	3	5	14	18	20	26
f(1)	4	16	9	12	1	1	15	14
f(2)	11	4	5	9	14	11	7	9
f(3)	12	9	4	5	13	13	5	5
f(4)	1	5	12	16	2	9	14	16
f(5)	2	7	11	15	5	6	13	10
f(6)	16	15	1	2	16	16	1	1
f(7)	8	2	16	7	9	2	9	8
f(8)	15	14	2	1	15	14	4	3
f(9)	7	1	15	11	8	4	12	11
f(10)	10	3	6	6	12	15	2	2
f(11)	13	6	10	10	4	12	11	7
f(12)	3	11	12	14	6	5	16	15
f(13)	6	13	13	13	7	4	10	12
f(14)	14	10	3	3	11	10	3	4
f(15)	9	8	8	4	10	8	6	6
f(16)	5	12	7	8	3	7	8	13
BD	11	12	8	6	7	7	7	7

Cuadro 2: Soluciones tras el filtro de calidad

igual a 11, el valor de la primera solución. Así, en las siguientes iteraciones se aceptarán la segunda y tercera solución y se actualizará UB con el valor de 8. De esta forma, posteriormente se rechazará la solución 4 con valor 10. En la quinta iteración se actualizará el valor del filtro a 6, por lo que en las próximas iteraciones se rechazarán todas las soluciones con valor superior a 6. Tras rechazar $maxr = 5$ soluciones consecutivas (de la 6 a la 10), se incrementa el valor del umbral UB para no colapsar el método. Utilizando la expresión $UB = UB + 0{,}2(1 + \mid UB \mid)$ se obtiene $UB = 7{,}4$. Así, en la próxima iteración se rechazarán las soluciones con valor igual o mayor que 8. El cuadro 2 muestra las 8 soluciones, de las 30 generadas, que han pasado el test de calidad.

Como se ha comentado, al aplicar un método de búsqueda local a dos soluciones similares, probablemente se obtendrá el mismo óptimo local. Para evitar esto se aplica el filtro de diversidad, y se descartan (y, por tanto, no se les aplica la fase de mejora) las soluciones que disten menos de una cantidad β de las soluciones previamente generadas a las que sí se les aplicó la fase de mejora. Dadas dos soluciones (permutaciones) $p = (p_1, p_2, ..., p_n)$ y $q = (q_1, q_2, ..., q_n)$, se considera como distancia entre ambas la siguiente:

$$d(p, q) = \sum_{i=1}^{n} \mid p_i - q_i \mid$$

Cada vez que se genera aleatoriamente una solución se calcula su distancia a los óptimos locales almacenados. Si la distancia a alguno de ellos es menor que el umbral considerado ($distfactor * maxdist$), entonces se descarta la solución y no se le aplica la fase de mejora. Se considera aquí la fase de mejora propuesta en Piñana et al. [26], basada en el intercambio de etiquetas.

Considérese en el cuadro 2 la primera solución; se le aplica la fase de mejora y se obtiene la solución

$$(3, 11, 12, 7, 8, 16, 6, 15, 2, 13, 10, 4, 1, 14, 9, 5)$$

con valor 5. La distancia de la solución original a la mejorada es de 32 ($maxdist = 32$), por lo que, considerando el parámetro $distfactor = 0{,}75$, se tiene que el umbral es de 24. En la segunda iteración, se considera la distancia de la segunda solución al óptimo local encontrado en la primera iteración. Dicha distancia es 78, superior al umbral considerado, por lo que no se descarta la solución y se le aplica la fase de mejora. Se obtiene un óptimo local con valor 6 y con una distancia a la soluciń inicial de 20. Por tanto, no se actualiza el valor $maxdist$ y se mantiene en 32. En la siguiente iteración se calcula la distancia de la tercera solución a los dos óptimos locales obtenidos. Estas distancias son de 122 y 108 respectivamente, por lo que no se descarta la solución y se le aplica la fase de mejora, obteniendo una solución con valor igual a 5. Siguiendo de esta forma, se comprueba la cercanía de las soluciones generadas a los óptimos locales obtenidos, y se descartan las correspondientes soluciones. En la cuarta iteración, el óptimo local obtenido es una solución óptima global del problema.

Agradecimientos

Los autores desean agradecer a Estefanía Piñana por su ayuda en la realización de los experimentos del problema del ancho de banda

Referencias

[1] **D.H. Ackley** An Empirical Study of Bit Vector Function Optimization. En *Genetic Algorithms and Simulated Annealing*, Ed. Davis, Morgan Kaufmann Publishers (1987)

[2] **S. Baluja** *An Empirical Comparison of 7 Iterative Efunction Optimization Heuristics* School of Computer Science, Carnegie Mellon University (1995)

[3] **B. Betrò, F. Schoen** Sequential Stopping Rules for the Multistart Algorithm in Global Optimization *Mathematical Programming* vol. 22 pp. 125-140 (1982)

[4] **C.G.E. Boender, A.H.G. Rinnooy Kan, L. Stougie, G.T. Timmer.** A Stochastic Method for Global Optimization. *Mathematical Programming* vol. 22 pp. 125-140 (1982)

[5] **C.G.E. Boender, R. Zielinski** A Sequential Bayesian Approach to Esimating the Dimension of Multinomial Distribution. En R. Zielinski, editor, *Sequential Methods in Statistics*. Banach Center Publications vol. 16, PWN-Polish Scientif Publishers, Warsaw (1982)

[6] **C.G.E. Boender, A.H.G. Rinnooy Kan** A Bayesian Analysis of the Number of Cells of a Multinomial Distribution *The Statistician* vol. 32 pp. 240-248 (1983)

[7] **C.G.E. Boender, A.H.G. Rinnooy Kan** Bayesian Stopping Rules for Multistart Global Optimization Methods *Mathematical Programming* vol. 37 pp. 59-80 (1987)

[8] **K.D. Boese, A.B. Kahng, S. Muddu** A New Adaptive Multistart Technique for Combinatorial Global Optimisation *Operations Research Letters* vol. 16 pp. 103-113 (1994)

[9] **R. H. Byrd, C.L. Dert, A.H.G. Rinnooy Kan, R. B. Schnabel** Concurrent Stochastic Methods for Global Optimization *Mathematical Programming* vol. 46 pp. 1-29 (1990)

[10] **E. Cuthill, J. McKee** Reducing the Bandwidth of Sparse Symmetric Matrices. En *Proc. ACM National Conference, Association for Computing Machinery*, New York, pp. 157-172 (1969)

[11] **T.A., M.G.C. Resende** A Probabilistic Heuristic for A Computationally Difficult Set Covering Problem. *Operations Research Letters* vol. 8 pp. 67-71 (1989)

[12] **T.A. Feo, M.G.C. Resende** Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization* vol. 6 pp. 109-133 (1995)

[13] **L.W. Hagen, A.B. Kahng A.B.** Combining Problem Reduction and Adaptive Multi-start: A New Technique for Superior Iterative Partitioning *IEEE Transactions on CAD* vol. 16 pp. 709-717 (1997)

[14] **J.P. Hart, A.W. Shogan** Semi-greedy Heuristics: An Empirical Study, *Operations Research Letters* vol. 6 pp. 107-114 (1987)

[15] **W.W. Hart** Sequential Stopping Rules for Random Optimization Methods with Applications to Multistart Local Search *Siam Journal on Optimization* vol. 9 pp. 270-290 (1998)

[16] **F.J. Hickernell, Y. Yuan** A Simple Multistart Algorithm for

Global Optimization, *OR Transactions* vol. 1 (1997)

[17] **C.R. Houck, J.A. Joines, M.G. Kays** Comparison of Genetic Algorithms, Random Restart and Two-Opt Switching for Solving Large Location-Allocation Problems, *Computers Operations Resarch*, vol. 23 pp. 587-596 (1996)

[18] **X. Hu, R. Shonkwiler, M.C. Spruill** *Random Restarts in Global Optimization* Georgia Institute of technology, Atlanta. (1994)

[19] **M. Locatelli, F. Schoen** Random Linkage: A Family of Acceptance-Rejection Algorithms for Global Optimization *Mathematical Programming* vol. 85 pp. 379-396 (1999)

[20] **Marc Los, Christian Lardinois** Combinatorial Programming, Statistical Optimization and the Optimal Transportation Network Problem. *Transportation Research* vol. 2 pp. 89-124 (1982)

[21] **R. Martí** Multi-Start Methods, En *State of the Art Handbook on MetaHeuristics*, Eds. F. Glover y G . Kochenberger, Kluwer (2000)

[22] **R. Martí, M. Laguna, F. Glover, V. Campos** Reducing the Bandwidth of a Sparse Matrix with Tabu Search, *European Journal of Operational Research* vol. 135 pp. 211-220 (2001)

[23] **D.Q. Mayne, C.C. Meewella** A Non-Clustering Multistart Algorithm for Global Optimization. En *Analysis and Optimization of Systems*, Ed. Bensoussan and Lions, Lecture Notes in Control and Information Sciences, vol. 111, Springer Verlag (1988)

[24] **J.A. Moreno, N. Mladenovic, J. M. Moreno-Vega** *An Statistical Analysis of Strategies for Multistart Heuristic Searches for p-Facility Location-Allocation Problems* Eighth Meeting of the EWG on Locational Analysis, Lambrecht, Germany (1995)

[25] **M.B. Melián-Batista, J. A. Moreno, J.M. Moreno-Vega** A Multistart Clustering Tecnhique for Combinatorial Optimization, en *Proceedings of the International Conference on Modelling and Simulation*, Eds. R. Berriel, V. Hernández, R. Montenegro, J. Rocha, Universidad de Las Palmas de Gran Canaria, (2000)

[26] **E. Piãna, I. Plana, V. Campos, R. Martí** GRASP and Path Relinking for the Matrix Bandwidth Minimization, *European Journal of Operational Research*, por aparecer (2003)

[27] **M. Prais, C.C. Ribeiro** Reactive GRASP: An Application to a Matrix Decomposition Problem inf TDMA Traffic Assignment, *Informs Journal on Computing* vol. 12 pp. 164-176 (2000)

[28] **A.H.G. Rinnooy Kan, G.T. Timmer** Stochastic Global Optimization Methods. Part II: Multi Level Methods *Mathematical Programming* vol. 39 pp. 57-78 (1987)

[29] **F. Schoen** Two Phase Methods for Global Optimization. En *Handbook of Global Optimization 2: Heuristic Approaches*, Eds. P. Pardalos y E. Romeijn, Kluwer Academic Publishers (2002)

[30] **F. Solis, R. Wets.** Minimization by Random Search Techniques. *Math. of Operations Research* vol. 6 pp. 19-30. (1981)

[31] **N.L.J. Ulder, E.H.L. Aarts, H.J. Bandelt, P.J.M. Van Laarhoven, E. Pesch** Genetic Local Search Algorithms for the Traveling Salesman Problem, *Parallel problem solving from nature*, Eds. Schwefel y Mnner, Springer Verlag, pp. 109-116 (1990)

[32] **Z. Ugray, L. Lasdon, J. Plummer, F. Glover, J. Kelly,**

R. Martí A Multistart Scatter Search Heuristic for Smooth NLP and MINLP Problems, enviado a *Informs Journal on Computing* (2001)

[33] **M. Wattenberg, A. Juels** *Stochastic Hillclimbing as a Baseline Method for Evaluationg Genetic Algorithms* University of California - Berkeley, CSD94-834 (1994)