View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repositorio Institucional Universidad de Málaga

Aspectos computacionales en la bisección de un n-simplex regular

G. Aparicio, [1] L.G. Casado, [2] I. García, E.M.T. Hendrix [3] y B. G.-Tóth [4]

Resumen— **En el ámbito de la optimización global basada en técnicas de ramificación y acotación, cuando el espacio de búsqueda es un n-simplex regular es habitual utilizar como regla de división la bisección por el lado mayor. Este modo de división evita que los subproblemas generados tengan una forma degenerada o poco redondeada y además da lugar a un muestreo más uniforme del espacio de búsqueda ya que la función objetivo es normalmente evaluada en los vértices de los sub-problemas o símplices. En este trabajo se muestra como la división por el lado mayor puede afectar a parámetros tales como el número total de sub-problemas generados, el número de formas similares que estos pueden tener o el grado de redondez de los sub-problemas. La dificultad de determinar estos parámetros se incrementa con el valor de n. En este trabajo se presentan los resultados de los estudios realizados para $n \leq 3$, es decir, hasta un espacio 4-dimensional.**

Debido al crecimiento exponencial del árbol binario de búsqueda generado, se hace necesario el uso de computación paralela cuando se usan criterios de terminación más precisos y/o n-símplices con $n \geq 3$. Aquí se presenta un modelo paralelo que hace uso de las posibilidades de paralelización de MATLAB.

Palabras clave— **Simplex, Bisección del Lado Mayor, Formas Similares, Paralelismo.**

I. Introducción

LOS algoritmos de ramificación y acotación (del inglés Branch-and-Bound, B&B) se aplican a la resolución de problemas de optimización global mediante la realización de una búsqueda exhaustiva del mínimo de una función objetivo en un dominio dado. En este trabajo estamos interesados en problemas donde el espacio de búsqueda está definido por un n-simplex regular, definido en un espacio $(n+1)$-dimensional [1]. Los algoritmos de B&B están caracterizados por las reglas de Acotación, Selección, División, Rechazo y Terminación. Se pretende estudiar las características del árbol de búsqueda generado cuando solo se tienen en cuenta las reglas de División y Terminación, es decir, ningún nodo del árbol es eliminado. Se usará como regla de división la bisección del lado mayor (BLM) [2], [3] y como regla de terminación la longitud del lado mayor de un subproblema o nodo del árbol.

En este trabajo investigamos el efecto de la BLM sobre el número de símplices generados, el número de formas similares de los subsímplices y su nivel de redondez para un determinado valor de la regla de terminación. El objetivo de estos estudios es la determinación del limite superior del número de símplices que se pueden generar a partir de uno dado, lo que permitirá estimar el trabajo pendiente a partir de un nodo del árbol de búsqueda. Esta información permitiría mejorar los métodos de balanceo dinámico de la carga en los correspondientes algoritmos paralelos de optimización global [4], [5].

En este articulo se presentan resultados para un 1-simplex, un 2-simplex y un 3-simplex. Debido al crecimiento exponencial del árbol binario, se hace necesario el uso de computación paralela cuando se usan criterios de terminación más precisos y/o n-símplices con $n \geq 3$. Por ejemplo, solo se han podido alcanzar 20 niveles del árbol de búsqueda para un 3-simplex usando un algoritmo secuencial en un computador de sobremesa.

En la Sección II se describen los Algoritmos de B&B utilizados para solucionar los problemas de Optimización Global que generan este tipo de árboles de búsqueda. En la Sección III se muestran las características específicas de los Algoritmos de B&B que realizan la búsqueda en un dominio definido por un simplex. En la Sección IV se presentan las métricas utilizadas para valorar las características del árbol que se genera, que son evaluadas en la Sección V. En la Sección VI se plantea el diseño de una versión paralela que haga uso de la ToolBox de computación paralela disponible en MATLAB. Finalmente en la Sección VII se muestran las conclusiones y trabajos futuros.

II. Optimización Global y los Algoritmos de B&B

La resolución de un problema de optimización global consiste en encontrar el valor mínimo o máximo de una función objetivo f, satisfaciendo ciertas restricciones en el espacio de búsqueda. Para problemas de minimización, cuando la solución está contenida en un simplex regular S, el problema de optimización global puede escribirse como:

$$f^* = \min_{x \in S} f(x), \text{ con } f(x^*) = f^*. \quad (1)$$

Cuando se requiere una búsqueda exhaustiva del mínimo global, se suelen utilizar algoritmos B&B. Un algoritmo B&B realiza una búsqueda exhaustiva basada en la descomposición sucesiva del problema inicial en sub-problemas de menor tamaño hasta alcanzar la(s) solución(es) final(es) o una aproximación a ella(s). Esta aproximación esta determinada por

[1]Grupo de Investigación TIC146: Supercomputación-Algoritmos, Universidad de Almería, España, e-mail: guillermoaparicio@ual.es
[2]Departamento de Informática, Universidad de Almería, España, e-mail: leo@ual.es
[3]Departamento de Arquitectura de Computadores, Universidad de Málaga, España, e-mail: igarciaf@uma.es Eligius.Hendrix@wur.nl
[4]Departamento de Ecuaciones Diferenciales. Universidad de Tecnología y Economía de Budapest, Hungría, e-mail: bog@math.bme.hu

Algoritmo 1 Ramificación y Acotación

1. $\Lambda := \{S_1 = S\}$ *Conjunto de trabajo*
2. $\Omega := \{\}$ *Conjunto final*
3. $ns := 1$ *Número de símplices*
4. **while** $\Lambda \neq \emptyset$
5. Selecciona un simplex S_i de Λ *Regla de selección*
6. Evalúa S_i *Regla de acotación: no usada aquí.*
7. **if** S_i no puede ser eliminado *Regla de rechazo: no usada aquí.*
8. **if** S_i satisface el criterio de terminación *Regla de terminación: $w(S_i) \leq \epsilon$*
9. Almacena S_i in Ω
10. **else**
11. $\{S_{2i}, S_{2i+1}\}$:=BLM(S_i) *Regla de división: Bisección del Lado Mayor.*
12. Almacena S_{2i}, S_{2i+1} en Λ
13. $ns := ns + 2$
14. **return** Ω y $f(x^*)$ *No usado aquí.*

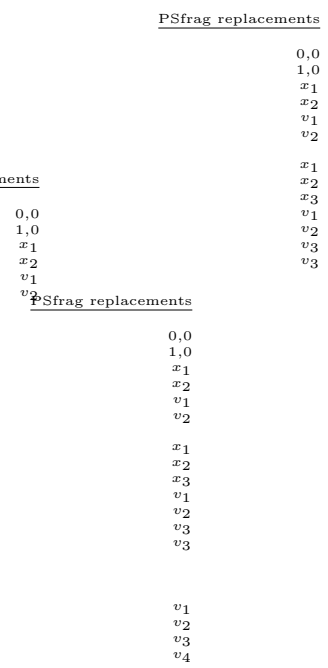

Fig. 1. Símplices de dimesiones 1, 2 y 3.

la precisión requerida para la solución. El algoritmo crea un árbol de búsqueda en el cual se puede podar una rama cuando se determina que un sub-problema no contiene una solución global. Para realizar la poda o rechazo de un sub-problema se suelen calcular cotas de los valores de la función objetivo para cada sub-problema. El Algoritmo 1 muestra un ejemplo básico. El comportamiento y por tanto la eficiencia de un algoritmo B&B se puede caracterizar por cinco reglas: Selección, Acotación, Rechazo, División y Terminación. En este estudio no se van a aplicar las reglas de Acotación, Rechazo y Selección ya que estamos interesados solo en la eficiencia de la regla de División. Al no existir poda del árbol de búsqueda, éste se generará completamente.

III. Algoritmos B&B para símplices

A. El simplex regular

En este estudio, el espacio inicial está determinado por un simplex regular. Un n-simplex está definido en un espacio de $n+1$ dimensiones. Un simplex es el politopo más sencillo posible en un espacio dado. Un 1-simplex es un segmento de una línea; un 2-simplex un triángulo; un 3-simplex es un tetraedro. La Figura 1 muestra gráficamente los ejemplos anteriores.

Estrictamente, un simplex se define como la envoltura convexa de un conjunto de $n+1$ puntos independientes afines en un espacio euclídeo de dimensión mayor o igual que n, es decir, el conjunto de puntos tal que ningún m-plano contiene más de $m+1$ de estos puntos con $m \leq n$.

El n-simplex unidad, o n-simplex estándar, es un subconjunto de $\mathbb{R}^{n+1}$ tal que:

$$T = \left\{ x \in \mathbb{R}^{n+1} \mid \sum_{j=1}^{n+1} x_j = 1;\ x_j \geq 0 \right\} \quad (2)$$

Por simplicidad y sin perdida de generalidad, en este estudio se va a hacer uso de un simplex regular con longitud de lado 1, que se define como:

$$S = \left\{ x \in \mathbb{R}^{n+1} \mid \sum_{j=1}^{n+1} x_j = \frac{\sqrt{2}}{2};\ x_j \geq 0 \right\}, \quad (3)$$

Las caracteristicas principales de un n-simplex son las siguientes

- Un n-simplex tiene $n+1$ vértices y $n(n+1)/2$ lados.
- A la envoltura convexa de un conjunto de $m+1$ puntos independientes afines de un n-simplex $(m < n)$ se denomina m-face.
- A la (n-1)-face se le denomina facet.
- Se define la anchura de un simplex, $\omega(S)$, como la longitud del lado mayor del simplex.

B. Bisección del Lado Mayor (BLM)

La BLM es uno de los métodos de refinamiento más populares ya que es muy simple y puede ser aplicado a problemas de cualquier dimensión [6].

La Figura 2 muestra los símplices generados después de tres bisecciones usando BLM sobre un 2-simplex regular.

Fig. 2. Primeras BLM en un 2-simplex regular.

El método de BLM puede implementarse como describe en el Algoritmo 2. La Figura 3 muestra los símplices en cada nivel del árbol binario que se ha

Algoritmo 2 BLM(S_i)

$S_i.\{v_1, \ldots, v_{n+1}\}$ *Conjunto de vértices de* S_i

1. $\{v_j, v_k\}$:= SeleccionaLadoMayor(S_i)
2. $v_{new} = \frac{v_j + v_k}{2}$
3. $S_{2i} := S_{2i+1} := S_i$ *Se heredan las características*
4. $S_{2i}.v_j = v_{new}$ *Se actualiza el nuevo vértice*
5. $S_{2i+1}.v_k = v_{new}$
6. **return** S_{2i}, S_{2i+1}

generado mediante el uso de BLM para un 2-simplex regular.

PSfrag replacements

Nivel 1

Nivel 2

Nivel 3

Nivel 4

Fig. 3. Niveles del árbol binario obtenido mediante BLM de un 2-simplex regular.

Si el simplex es irregular solo existe un lado mayor cuando $n \leq 2$. Para un 3-simplex, después de la primera BLM aparecen dos símplices irregulares con varias opciones para elegir el lado mayor. La Figura 4 muestra gráficamente este caso.

Fig. 4. Lados mayores tras la primera bisección de un 3-simplex regular.

IV. Caracterización del árbol de búsqueda.

Un método de división debería contribuir a:

- Realizar un muestreo uniforme del espacio de búsqueda.
- Garantizar la convergencia del algoritmo.
- Reducir el tamaño del árbol de búsqueda.

La BLM garantiza la convergencia del algoritmo ya que se ha establecido como regla de terminación la longitud del lado mayor. En cuanto a la realización de un muestreo uniforme, se pretende que la forma de los subproblemas sea lo más redondeada posible. Además, estamos interesados en que el número de formas similares generadas sea la menor posible para hacer más fácil la estimación del tamaño de un subárbol a partir de un nodo.

A continuación se presentan estás métricas, las cuales dependerán del lado mayor seleccionado, en los casos en los que existan varios y el simplex no sea regular.

A. Tamaño del árbol de búsqueda

El tamaño del árbol dependerá de como disminuye el tamaño de los sub-problemas conforme se aplica la regla de División. Kearfott presenta un límite superior del tamaño de los sub-problemas cuando se usa BLE en el siguiente teorema [7]:

Teorema 1: Sea S_0 un n-simplex, sea p un entero positivo cualquiera y sea S_p cualquier n-simplex que se ha producido tras p BLM divisiones de S_0. Entonces, el diámetro de S_p (o lado mayor, $\omega(S_p)$) nunca será mayor que $(\frac{\sqrt{3}}{2})^{\lfloor \frac{p}{n} \rfloor}$ veces el diámetro de S_0 ($\omega(S_0)$), donde $\lfloor \frac{p}{n} \rfloor$ es el mayor entero menor o igual que $\frac{p}{n}$.

Kearfott también mostró que, para un simplex S y $p > n$, los diámetros se reducían un factor 2 cada n iteraciones.

Stynes [8] mejoró la cota de Kearfott para un 2-simplex:

$$\omega(S_{2p}) \leq \frac{\sqrt{3}}{2} \left(\frac{1}{2}\right)^{\frac{p}{2}-1}. \tag{4}$$

Por ejemplo, para p=6, la cota de Kearfott es

$$\omega(S_6) \leq \frac{3\sqrt{3}}{2^3}, \tag{5}$$

mientras que la cota de Stynes es

$$\omega(S_6) \leq \frac{\sqrt{3}}{2^3}, \tag{6}$$

Nuestro propósito es hallar el valor exacto del diámetro de los símplices, después de $p > n$ iteraciones, en vez de una cota superior. Para ello nos basaremos en las siguientes definiciones.

Definición 1: Un nivel l del árbol binario se dice completo si el número de elementos en el nivel es 2^{l-1}.

Definición 2: El último nivel completo L_c de un árbol binario es máx$\{l,$ con 2^{l-1} elementos$\}$.

A continuación presentamos los resultados obtenidos para símplices con $n \leq 3$.

A.1 1-simplex

En el caso más simple, usando la BLM en un 1-simplex, el diámetro $\omega(X)$ de los símplices X en el nivel l es:

$$\omega(X) = \frac{\omega(S)}{2^l} \tag{7}$$

Para generar símplices con $\omega(X) \leq \epsilon$ se tiene que:

$$\epsilon \geq \frac{\omega(S)}{2^L}, \tag{8}$$

y el último nivel L del árbol es:

$$L = \lceil \log_2 \frac{\omega(S)}{\epsilon} \rceil \tag{9}$$

A.2 2-simplex

Para el caso de un 2-simplex se pueden obtener las siguientes conclusiones

Proposición 1: Sea $\epsilon > 0$ el criterio de terminación. Dado un 2-simplex regular inicial S con $\omega(S) = 1$ y $k \in \mathbb{N}^+$. El último nivel del árbol binario L es el último nivel completo L_c si ϵ está en el intervalo

$$\frac{1}{2^{k-1}} \leq \epsilon < \frac{\sqrt{3}}{2^k}, \tag{10}$$

con

$$L = L_c = 2k + 1, \tag{11}$$

Proposición 2: Bajo las mismas condiciones de la Proposición 1, el último nivel del árbol L no estará completo cuando: $\frac{\sqrt{3}}{2^k} \leq \epsilon < \frac{1}{2^{k+1}}$, siendo el número de nodos en el árbol igual a $2^{L-1} - 1 + \frac{2^{L-1}}{2}$. En este caso, el nivel alcanzado en el árbol binario será:

$$L = 2k + 2 \tag{12}$$

pero tan solo la mitad de los símplices de ese nivel habrán sido generados y por tanto el número total de nodos en el árbol será:

$$N_{simplex} = 2^{L-1} + 2^{L-2} - 1 \tag{13}$$

A.3 3-simplex

A partir de n=3 pueden existir varias opciones en la selección del lado mayor [9]. En este estudio se usarán los siguientes métodos:

- BLM_1*:* Es el método normalmente usado y se basa en elegir el primer lado mayor en términos de los índices de los vértices del simplex.
- BLM_α*:* Selecciona el lado mayor, cuyos ángulos con los otros lados son los menores. De esta manera se reducen los ángulos mayores.

Usando BLM_α se pueden deducir los intervalos en los que se encontraría ϵ según el valor de k, ya que es más fácil determinar el tamaño del lado mayor de los símplices:

$$\frac{\sqrt{3}}{2^k} \leq \epsilon_1 < \frac{\sqrt{2}}{2^k} \leq \epsilon_2 < \frac{1}{2^k} \leq \epsilon_3 < \frac{\sqrt{3}}{2^{k+1}} \tag{14}$$

La ecuación (15) muestra los posibles valores de k:

$$k = \frac{L - P}{3} \begin{cases} P = 4 \text{ si } \epsilon = \epsilon_1 \\ P = 5 \text{ si } \epsilon = \epsilon_2 \\ P = 6 \text{ si } \epsilon = \epsilon_3 \end{cases} \tag{15}$$

El valor de L viene dado por:

$$L = \lceil (P-3) + 3 \cdot log_2(\frac{Z}{\epsilon}) \rceil, \text{con} \begin{cases} P = 4 \ y \ Z = \sqrt{3} \text{ si } \epsilon = \epsilon_1 \\ P = 5 \ y \ Z = \sqrt{2} \text{ si } \epsilon = \epsilon_2 \\ P = 6 \ y \ Z = 1 \text{ si } \epsilon = \epsilon_3 \end{cases} \tag{16}$$

B. Número de clases de símplices

Definición 3: Dos símplices pertenecen a la misma clase si tienen formas similares, es decir, si aplicando transformaciones de escala, rotación y desplazamiento a un simplex, se puede obtener la identidad con el otro simplex.

La Figura 5 muestra el proceso para determinar si dos símplices pertenecen a la misma clase.

Fig. 5. Escalado, traslación y rotación de 2-símplices.

Los requerimientos computacionales del proceso de determinación del número de clases crece con el tamaño del árbol de búsqueda ya que hay que comprobar si un nuevo simplex pertenece a una clase existente o crea una nueva clase.

C. Índice de redondez de un simplex

En la literatura existen diversos índices que informan sobre la redondez de un simplex. En [10], [11] se presentan varias medidas de redondez o calidad para un simplex y se demuestra que la mayoría son equivalentes.

Aquí se muestra un nuevo índice de redondez de un simplex al que denotaremos por SQ (Simplex Quality) basado en la relación entre la media geométrica y la aritmética de los ángulos existentes entre los lados coincidentes en los vértices de un simplex [12]. Sea m el número total de ángulos que posee un n-simplex, se define SQ como:

$$SQ(S) = n \cdot \frac{\sqrt[n]{\alpha_1^2 \cdot \alpha_2^2 \cdot ... \cdot \alpha_m^2}}{\alpha_1^2 + \alpha_2^2 + ...\alpha_m^2} \tag{17}$$

Este índice genera valores elevados para símplices regulares y valores pequeños para símplices con forma de aguja.

Además, para obtener una versión normalizada de este índice, se define NSQ (Normalized Simplex Quality) como:

$$NSQ(S) = \frac{SQ(S)}{SQ(S_1)} \in (0, 1], \tag{18}$$

V. Resultados de la versión secuencial

El algoritmo secuencial se ha desarrollado en MATLAB y se ha ejecutado sobre un procesador Intel(R) Xeon(R) CPU E5-2620 con 8 cores a 2.00 GHz, 20Mb de caché L3 y 64GB de RAM.

La Tabla I muestra el número de símplices generados N, el nivel máximo del árbol de búsqueda L, el tiempo de ejecución cuando se desactiva el cálculo de clases en el algoritmo (T_{nc}) y el tiempo de ejecución cuando sí se realiza el cálculo de clases (T), para un 2-simplex y distintos valores de ϵ.

TABLA I
Resultados para un 2-simplex

ϵ	N	L	T_{nc}	T
0.5	11	4	0.24s	0.30s
0.25	47	6	0.26s	0.32s
0.125	191	8	0.36s	0.48s
0.0625	767	10	0.75s	1.14s
0.03125	3071	12	2.29s	3.74s
0.015625	12287	14	8.48s	14.34s

La Tabla II muestra la misma información para un 3-simplex usando BLM_α.

TABLA II
Resultados usando BLM_α para un 3-simplex

ϵ	N	L	T_{nc}	T
0.5	47	6	0.28s	1.46s
0.25	351	9	0.59s	10.93s
0.125	2751	12	3.01s	1.5m
0.0625	21887	15	22.37s	12.16m
0.03125	174847	18	2.95m	1.65h
0.015625	1398271	21	23.78m	13.20h

Puede observarse como el tamaño del árbol binario de búsqueda generado y el tiempo de ejecución aumenta conforme disminuye ϵ y aumenta n. Por este motivo se hace necesario el uso de computación paralela cuando se usan criterios de terminación más precisos y/o n-símplices con $n \geq 3$.

La Tabla III muestra el número máximo de clases NMC, los valores de NSQ y el número de símplices en cada clase, para un 2-Simplex y un 3-Simplex. Para el 3-simplex se usa BLM_α. Como puede observarse, BLM_α permite obtener un número de clases fijo para un 3-simplex.

La Tabla IV compara el número de símplices, el número total de clases, el valor medio de NSQ y el nivel alcanzado en el árbol binario obtenido usando BLM_1 y BLM_α sobre un 3-Simplex con $\epsilon \leq 0{,}03125$.

Para BLM_1 el número de clases crece conforme decrece ϵ [13]. BLM_α genera menos símplices, con formas más redondeadas en media y el número de clases es mucho menor y fijo.

TABLA III
NMC y NSQ para un 2-Simplex y un 3-Simplex

n	NMC	NSQ	Nº Simpl.
2	3	1	683
		0.6429	1706
		0.3333	682
3	8	1	1
		0.7504	12400
		0.6509	24800
		0.6177	25050
		0.4994	56176
		0.4414	6324
		0.4081	25048
		0.3729	25048

TABLA IV
Comparación de BLM_1 y BLM_α sobre un 3-Simplex. $\epsilon \leq 0{,}03125$.

	BLM_1	BLM_α
Nº símplices	259701	174847
Nº Clases	300	8
NSQ Medio	0.4117	0.5926
L	20	18

VI. Versión paralela

Como se ha mostrado, este tipo de experimentos tienen grandes requerimientos de procesamiento y memoria. Además, distintas ramas del árbol pueden generarse de forma independiente, lo que hace a este tipo de algoritmos buenos candidatos para su paralelización.

Para la ejecución de la versión paralela se utilizó un nodo de cómputo de un servidor Bullx, compuesto por dos procesadores como el usado en la versión secuencial. Se ha hecho uso de la Parallel Computing ToolBox que MATLAB posee para tareas específicas de computación paralela.

El modelo de paralelismo se basa en que cada proceso se encargue de generar una rama del árbol. El Algoritmo 3 muestra el pseudo-código.

Algoritmo 3 Paralelización del Código

1. NP *Número de procesos paralelos*
2. NS_h *Número de símplices hoja*
3. Se ejecuta el Algoritmo 1 hasta que $NS_h = NP$
4. Se crean NP procesos
5. Cada proceso ejecuta el Algoritmo 1 sobre uno de los simplices creados.
6. Se combina la información obtenida en los pasos 3 y 5
7. **return** Estadísticas globales de la ejecución.

La Figura 6 muestra los resultados del Algoritmo 3 para NP= 1, 2, 4, 8 y 12, utilizando tanto BLM_α como BLM_1 con distintos valores de ϵ.

Fig. 6. Speedup para distintos vales de ϵ utilizando BLM_α y BLM_1

Como puede observarse, para BLM_α el speed-up crece de manera lineal hasta llegar a 8 procesos, valor que coincide con el número de cores que posee cada procesador. Sin embargo, el speed-up para BLM_1 es menor. Esto es debido a que BLM_α genera solo 8 clases de símplices, mientras que cuando se usa BLM_1 el número de clases aumenta conforme disminuye ϵ. Cada proceso paralelo genera un número de clases parecido o igual al de la versión secuencual, por lo que el trabajo realizado para determinar si un simlex pertenece a una de las clases existentes o genera una nueva, tiene una carga computacional similar a la versión secuencial y no se realiza en paralelo. Además, el paso 6 del Algoritmo 3 se realiza de forma secuencial, siendo el número de combinaciones de información mayor en BLM_1 que en BLM_α, debido al número de clases generadas.

Hay que destacar que la versión paralela ha permitido la obtención de resultados para $\epsilon = 0{,}015625$ usando BLM_α. El tiempo de ejecución usando BLM_1 para ese valor de ϵ, es de más de un día.

Aunque la ToolBox de programación paralela de MATLAB permite definir hasta 12 ejecuciones en paralelo y disponer de dos procesadores por nodo de cómputo, los resultados obtenidos hacen pensar que el uso de los cores del segundo procesador reduce la tendencia lineal de la ganancia en velocidad obtenida en un solo procesador.

VII. Conclusiones y trabajo futuro

Los algoritmos de B&B aplicados a problemas optimización global, suelen exigir una alta carga computacional cuando se buscan soluciones precisas. Este trabajo presenta un estudio sobre el comportamiento de estos algoritmos cuando el espacio de búsqueda es un simplex. Se presentan estudios sobre el número de elementos del árbol, su nivel de calidad y el número de clases similares cuando se usa una bisección del lado mayor como regla de división. Se presenta un nuevo método de selección del lado mayor a dividir cuando existen varias alternativas. Estos algoritmos son buenos candidatos para su paralelización. Se muestran los resultados obtenidos con una versión paralela desarrollada en MATLAB obteniéndose un speed-up casi lineal hasta 8 cores cuando se usa el método de bisección propuesto BLM_α.

Como trabajos futuros se pretende extender los métodos desarrollados a problemas con $n > 3$. Para ello será necesario desarrollar algoritmos que permitan el uso eficiente de un número mucho mayor de cores.

Agradecimientos

Este trabajo ha sido subvencionado por el Ministerio de Ciencia e Innovación (TIN2008-01117, TIN2012-37483-C03-01 y TIN2012-37483-C03-03) y la Junta de Andalucía (P011-TIC7176) y financiado en parte por el Fondo Europeo de Desarrollo Regional (ERDF).

Referencias

[1] L.G. Casado, I. García, B.G. Tóth, and E.M.T. Hendrix, "On determining the cover of a simplex by spheres centered at its vertices," *Journal of Global Optimization*, vol. 50, no. 4, pp. 645–655, 2011, doi:10.1007/s10898-010-9524-x.

[2] Andrew Adler, "On the Bisection Method for Triangles," *Mathematics of Computation*, vol. 40, no. 162, pp. 571–574, 1983, doi:10.1090/S0025-5718-1983-0689473-5.

[3] R. Horst, "On generalized bisection of n-simplices," *Mathematics of Computation*, vol. 66, no. 218, pp. 691–698, 1997.

[4] José L. Berenguel, L.G. Casado, I. García, and E.M.T. Hendrix, "On estimating workload in interval branch-and-bound global optimization algorithms," *Journal of Global Optimization*, In Press, doi:10.1007/s10898-011-9771-5.

[5] J.F. Sanjuan-Estrada, L.G. Casado, and I. García, "Adaptive parallel interval branch and bound algorithms based on their performance for multicore architectures," *Journal of Supercomputing*, vol. 58, pp. 376–384, 2011, doi: 10.1007/s11227-011-0594-4.

[6] A. Hannukainen, S. Korotov, and M. Krízek, "On numerical regularity of the longest-edge bisection algorithm," *BcaMath*, 2013.

[7] Baker Kearfott, "A proof of convergence and an error bound for the method of bisection in $\mathbf{R}^n$," *Journal of Mathematical Computing*, vol. 32, no. 144, pp. 1147–1153, 1978.

[8] M. Stynes, "On the faster convergence of the bisection method for all triangles," *Math. Comp.*, vol. 35, pp. 1195–1201, 1980.

[9] Anwei Liu and Barry Joe, "On the shape of tetrahedra from bisection," *Math. Comput.*, vol. 63, no. 207, pp. 141–154, July 1994, doi: 10.2307/2153566.

[10] V.N. Parthasarathy, C.M. Graichen, and A.F. Hathaway, "A comparison of tetrahedron quality measures," *Finite Elements in Analysis and Design*, vol. 15, no. 3, pp. 255 – 261, 1994.

[11] A. Plaza y G. F. Carey, "Explicación geométrica de una medida de forma de símplices n-dimensional.," *Métodos numéricos para cálculo y diseño en ingeniería*, vol. 18, no. 4, pp. 475–480, 2011.

[12] Ivo G. Rosenberg and Frank Stenger, "A lower bound on the angles of triangles constructed by bisecting the longest side," *Mathematics of Computation*, vol. 29, no. 130, pp. 390–395, 1975, doi:10.1090/S0025-5718-1975-0375068-5.

[13] Joseph M. Maubach, "The efficient location of neighbors for locally refined n-simplicial grids," in *In 5th Int. Meshing Roundable*, 1996, pp. 137–153.