Uso de algoritmos genéticos para la obtención de modelos estadísticos de rendimiento

D. R. Martínez[1], V. Blanco[2], J. C. Cabaleiro[1], T. F. Pena[1] y F. F. Rivera[1]

Resumen— **El entorno de análisis TIA proporciona modelos precisos del rendimiento de aplicaciones paralelas mediante el uso de técnicas estadísticas basadas en selección de modelos. Este método selecciona el mejor modelo entre un conjunto de candidatos proporcionados por el usuario, y ofrece información adicional para que el usuario haga una valoración subjetiva del modelo propuesto por el entorno. Uno de los aspectos que condiciona la aplicabilidad del modelado es el elevado tiempo necesario para encontrar la solución óptima debido a la alta dimensionalidad del espacio de búsqueda. En este artículo se presenta la aplicación de un algoritmo genético para reducir el tiempo de búsqueda cuando el conjunto de modelos candidato es muy elevado. Los resultados demuestran que con unos tiempos muy reducidos se consigue alcanzar en muchos casos el modelo óptimo y en el resto un modelo con una calidad próxima a la óptima.**

Palabras clave— **Modelos de rendimiento, selección de modelos, algoritmos genéticos, AIC.**

I. Introducción

La gran demanda de recursos computacionales ha motivado un cambio de paradigma en el desarrollo de los sistemas de computación. El aumento de la complejidad de los sistemas paralelos actuales ha dotado al usuario final de una mayor capacidad de computación. El coste de este beneficio es una mayor complejidad en el análisis del rendimiento y la evaluación, tanto de las aplicaciones paralelas como de los propios sistemas. El desarrollo de herramientas de análisis capaces de lidiar con esta complejidad computacional es un aspecto fundamental para el aprovechamiento de los recursos computacionales.

TIA (*Tools for Instrumentacion and Analysis*) [1] es un entorno para la generación de modelos de rendimiento de aplicaciones paralelas que está compuesto de dos módulos independientes, uno de instrumentación y otro de análisis. En el módulo de análisis se realiza el procesamiento y selección de los datos obtenidos por la monitorización. Los datos resultantes del paso anterior se ajustan a cada una de las fórmulas proporcionadas por el usuario, dando como resultado final el modelo con el mejor índice de calidad en cuanto al ajuste con los datos experimentales.

El proceso de generación de modelos analíticos se basa en la eliminación progresiva de los términos de la función inicial que presentan un mayor error relativo en el ajuste de los datos monitorizados. Este proceso se realiza hasta que todos los términos de la función presenten un error relativo menor que el umbral establecido. El proceso se ha generalizado para obtener el mejor modelo a partir de un conjunto de funciones. Ese conjunto de funciones se define como todas las posibles combinaciones de términos que pueden influir en el modelado. Dicho conjunto puede ser de gran dimensionalidad, y constituye el principal cuello de botella en su aplicación práctica.

Es bien conocido que el problema de búsquedas en un espacio de gran dimensionalidad puede ser abordado de manera eficiente utilizando técnicas heurísticas como algoritmos genéticos (GA) [2]. En este trabajo se muestra la aplicación de esta técnica para mejorar el tiempo de ejecución de la etapa de análisis de datos de TIA.

La estructura de este artículo se describe a continuación. En la sección II se introduce el proceso de obtención de modelos estadísticos de rendimiento. El entorno de análisis en el que se ha implementado este proceso se muestra en la sección III. En la sección IV se describe el algoritmo genético implementado en TIA para reducir el tiempo de ejecución de la obtención del modelo. En la sección V se lleva a cabo un caso de estudio donde se muestra la mejora obtenida utilizando algoritmos genéticos en lugar de la búsqueda exhaustiva. Finalmente, se presentan las principales conclusiones de este trabajo.

II. Selección de modelos

Conceptualmente, los datos observados contienen información que se puede expresar de una manera compacta a través de un modelo analítico. El objetivo ideal de la selección de modelos es conseguir una traslación perfecta, uno a uno, de manera que no se pierda información durante el proceso de generación del modelo. Este objetivo es imposible debido a que el conjunto de datos siempre está constituido por un número finito de elementos, que contienen una cantidad limitada de información. Teniendo en cuenta esta dificultad, el objetivo real es obtener el modelo que mejor se ajuste a los datos, esto es, el modelo que pierda la menor cantidad de información posible.

El criterio de información de Akaike (*An Information Criterion*, AIC) [3] proporciona un método simple y objetivo que selecciona el modelo más adecuado para caracterizar los datos experimentales. Este criterio, que se enmarca en el campo de la teoría de la información, se define como:

$$\mathrm{AIC} = -2\log(\mathcal{L}(\widehat{\theta})) + 2K \qquad (1)$$

donde $\log(\mathcal{L}(\widehat{\theta}))$ es el logaritmo de la máxima verosimilitud, que permite determinar los valores de los parámetros libres de un modelo estadístico [4], y K es

[1]Centro de Investigación en Tecnoloxías da Información – CITIUS, Dpto. de Electrónica y Computación, Univ. Santiago de Compostela, e-mail: {diego.rodriguez, jc.cabaleiro, tf.pena, ff.rivera}@usc.es

[2]Dpto. de Estadística e Investigación Operativa, Univ. La Laguna, e-mail: Vicente.Blanco@ull.es

el número de parámetros libres del modelo. Esta expresión proporciona una estimación de la distancia entre el modelo y el mecanismo que realmente genera los datos observados, que es desconocido y en algunos casos imposible de caracterizar. Como la estimación se hace en función de los datos experimentales, esta distancia es siempre relativa y dependiente del conjunto de datos experimentales. Por tanto, un valor individual de AIC no es interpretable por sí solo, y los valores AIC sólo tienen sentido cuando se realizan comparaciones utilizando los mismos datos experimentales.

Aunque no justifica las bases teóricas en las que se fundamenta, es posible hacer una interpretación heurística de la expresión (1). Esta interpretación considera el primer término de esta ecuación como una medida de la calidad con la que el modelo se ajusta a los datos experimentales, mientras que el segundo sería una *penalización* que se incrementa con la complejidad del modelo. Por ejemplo, supongamos que, a partir de un conjunto de datos experimentales, se calculan los valores AIC obtenidos para dos modelos diferentes. El menor valor de AIC indica que o bien el modelo se ajusta mejor a los datos experimentales o que es menos complejo, y, en realidad, una combinación de ambos factores. Por lo tanto, este criterio ofrece un valor objetivo que, de manera relativa, cuantifica simultáneamente la precisión y sencillez del modelo.

Cuando el número de parámetros (K) es muy elevado en relación con el tamaño de la muestra (n) los resultados que proporciona AIC pueden no ser satisfactorios. En estos casos se utiliza una aproximación de segundo orden:

$$\mathrm{AIC}_c = \mathrm{AIC} + \frac{2K(K+1)}{n-K-1} \qquad (2)$$

Cuando el cociente n/K es suficientemente grande, ambos valores (AIC y AIC_c) son muy similares.

III. El entorno de análisis TIA

El entorno de análisis TIA [1] consta de dos fases. En la primera, denominada fase de instrumentación, el usuario instrumenta el código fuente de una aplicación y la información acerca del rendimiento, obtenida durante la ejecución de la aplicación, es almacenada en ficheros XML. La segunda etapa, denominada fase de análisis, utiliza la información procedente de múltiples ejecuciones instrumentadas para generar un modelo analítico del rendimiento de la aplicación bajo estudio mediante análisis estadísticos exhaustivos [5]. La primera fase (instrumentación) está basada en la herramienta CALL [6], una herramienta que permite al usuario interaccionar con el código y obtener métricas de rendimiento de aplicaciones paralelas. CALL es una herramienta modular que, a través de *drivers*, coordina el manejo de herramientas externas o de módulos desarrollados en la propia herramienta. La instrumentación del código se implementa mediante el uso de *pragmas*, que permiten definir las zonas de interés del código, así como las métricas que se emplearán en la caracterización del rendimiento de la aplicación. La información de múltiples ejecuciones del código instrumentado se utiliza en la segunda fase (análisis) para obtener, mediante técnicas estadísticas, un modelo analítico del tiempo de ejecución de la aplicación en función de las métricas seleccionadas [5]. Esta segunda fase está basada en R [7], un conocido y potente lenguaje y entorno para el análisis estadístico.

El proceso de obtención de modelos estadísticos de aplicaciones paralelas implementado en la fase de análisis del entorno TIA está basado en la selección de modelos mediante AIC [8]. Este proceso realiza un análisis AIC sobre un conjunto finito de modelos candidato, generados a partir de información suministrada por el usuario. El resultado final es el modelo que presente el AIC más bajo, así como información acerca del análisis que permita al usuario hacer una valoración del resultado obtenido.

El conjunto de modelos candidato (`MC`) es el elemento fundamental para garantizar la calidad del proceso de selección AIC. La responsabilidad de esta tarea recae inicialmente sobre el usuario, que debe proporcionar un conjunto de modelos adecuado. En particular, se ha diseñado un mecanismo de descripción para representar un conjunto de modelos mediante una estructura de dos niveles. Esta descripción define un modelo global y considera como elementos del conjunto de modelos todos los posibles modelos anidados derivados del modelo global. Los elementos constitutivos del nivel más interno del anidamiento son las diferentes variables (métricas o parámetros) que a priori tienen influencia en el comportamiento del experimento. El nivel más externo del anidamiento está formado por conjuntos disjuntos de estas variables.

Este agrupamiento de las variables permite definir, de manera implícita, una nueva lista de variables derivadas formada por todos los posibles productos (denominados términos) de variables pertenecientes a diferentes conjuntos. En esta lista también se considera que cada grupo contiene implícitamente el *elemento identidad*. El modelo global que define el conjunto de modelos está formado por la suma de todos los términos, con sus correspondientes coeficientes. Para una descripción que genere N términos, el modelo global tendrá la siguiente estructura, que no representa más que la combinación lineal de todos los términos considerados:

$$M_{\mathrm{global}} = \sum_i C_i T_i \,, \qquad i = 1, 2, \ldots, N \qquad (3)$$

donde T_i representa el término i-ésimo y C_i su correspondiente coeficiente de ponderación. En cualquier caso, una vez definidos los diferentes conjuntos de variables primarias, la construcción del modelo global, incluyendo la generación de términos, es automática. En este proceso, el usuario únicamente necesita determinar los conjuntos de variables primarias. En un entorno R, esta estructura de conjuntos se puede implementar mediante una lista de listas, que denominamos Lista Inicial (`LI`), en la que cada

lista interior contiene las variables asociadas a cada conjunto. Para complementar este mecanismo, se ha considerado una extensión que permite al usuario contemplar otros *términos* aparte de los generados automáticamente a partir de los grupos de variables primarias.

Una vez determinado el conjunto de modelos candidatos, se realiza la selección de modelos con AIC. En primer lugar, se ajustan todos los modelos a los datos experimentales y se obtiene, para cada modelo, una aproximación de segundo orden de AIC [9]. El modelo que presente un menor valor de AIC se propondrá como *mejor* modelo. A partir de los valores AIC de todos los modelos candidatos, se calculan los pesos de Akaike (ω^{AIC}), pero también se proporciona la importancia relativa de cada uno de los términos considerados ($\omega_{+}^{\mathrm{AIC}}$). La importancia relativa de una variable predictora j será la suma de los pesos de Akaike de todos los modelos que contengan j como variable predictora. Con esta información, el usuario tendrá la capacidad de valorar si el modelo propuesto es adecuado para caracterizar el comportamiento de la aplicación bajo estudio.

IV. Algoritmo genético para la selección de modelos

El método de selección de modelos implementado en TIA realiza una búsqueda exhaustiva sobre el conjunto de modelos candidato. Aunque este mecanismo garantiza la obtención del mejor modelo candidato (el que presente el menor valor AIC), el procedimiento no es escalable porque el número de potenciales modelos crece rápidamente con el número de métricas y parámetros de rendimiento considerados en la generación del conjunto de modelos candidato. En particular, el tiempo de ejecución del proceso de búsqueda exhaustiva crece de forma exponencial con el tamaño de la lista de términos (`TL`) generada por TIA. Por lo tanto, es necesario el uso de metaheurísticas para que la búsqueda bajo estas condiciones pueda realizarse en un tiempo razonable.

En este trabajo se propone el uso de algoritmos genéticos [2] para reducir el tiempo de ejecución del proceso de selección de modelos del entorno TIA. La figura 1 muestra el pseudocódigo del AG implementado en la fase de análisis de TIA.

El primer paso del AG es la generación de una población inicial aleatoria de μ individuos. En este caso, cada individuo representará a un modelo candidato. Los modelos candidatos pueden ser caracterizados por la presencia/ausencia de los diferentes términos, por lo que se ha utilizado una codificación binaria de los individuos: cada bit de la representación está asociado a un elemento de la lista `TL`, por lo que la longitud de la cadena de bits coincide con el número de elementos de esta lista. El siguiente paso es la obtención de los valores AIC de los modelos representados en esta población inicial.

A continuación comienza el ciclo de selección/reproducción. En primer lugar, se realiza una selección mediante torneo binario para obtener los padres de la próxima generación de individuos. El operador de cruce de un único punto ha sido usado para generar la descendencia de los individuos seleccionados anteriormente. Se ha utilizado un modelo generacional, por lo que se descartan los padres para formar parte de la siguiente generación. El operador de mutación modifica los individuos de esta nueva generación. Este operador niega el valor binario de, como mucho, ν bits de la codificación de cada individuo. Para preservar la pérdida del mejor individuo de cada generación, se utiliza un esquema elitista en el que el mejor individuo de cada generación forma parte de la nueva generación sin verse afectado por las mutaciones. En este punto, la nueva generación, formada por μ individuos, se convierte en la generación actual del AG. Finalmente, se evalúa el valor AIC de los individuos de la generación actual, finalizando el ciclo de selección/reproducción del AG. Este ciclo se repite hasta que la población actual alcanza la condición de terminación. En ese momento, el individuo que presente el menor valor AIC es seleccionado como el mejor modelo candidato.

La implementación particular de este AG en el entorno TIA fue desarrollada para permitir al usuario ajustar los parámetros del algoritmo. En particular, el usuario debe seleccionar el tamaño de la población (μ), el número máximo de mutaciones permitidas en cada ciclo (ν) y la condición de terminación. En esta implementación, la condición de terminación se activa cuando el mejor individuo (el modelo que presenta un menor AIC) ha permanecido en la población durante las últimas ϕ generaciones, es decir, cuando el ciclo de selección/reproducción es ejecutado ϕ veces sin que se haya encontrado un modelo con un menor AIC.

V. Caso de estudio: HPL

El benchmark HPL (*High Performance Linpack*) es una extensión del benchmark LINPACK para resolver sistemas lineales densos de ecuaciones de orden N en entornos paralelos distribuidos [10]. La distribución de la matriz de coeficientes entre los procesadores garantiza la escalabilidad del programa, así como un buen balanceo de la carga. La matriz, de tamaño $N \times (N+1)$, se distribuye en bloques de tamaño $NB \times NB$ de forma cíclica, en ambas dimensiones, en una malla bidimensional de $P \times Q$ procesadores.

En anteriores trabajos [11], se ha obtenido el modelo de rendimiento del benchmark dentro del entorno de análisis TIA y se ha utilizado la siguiente lista para iniciar el proceso de análisis:

$$\mathtt{LI} = \left\{\mathtt{N}^3, \mathtt{N}^2\right\}, \left\{\frac{1}{\mathtt{NB}}\right\}, \left\{\frac{1}{\mathtt{Q}}\right\}, \left\{\frac{1}{\mathtt{P}}\right\} \tag{4}$$

Esta lista inicial genera 16.777.216 modelos candidatos ($2^{24} - 1$). Para cada uno de estos modelos se ha obtenido el valor AIC correspondiente y finalmente, de un modo absoluto, se ha seleccionado el modelo

```
BEGIN
  Inicialización: generación población inicial aleatoria
  Evaluación: obtención de AIC de la población inicial
  REPEAT UNTIL ( condición de terminación ) DO
    Selección: selección de los padres de la siguiente generación
    Recombinación: generación de la descendencia
    Mutación: mutaciones en la descendencia
    Evaluación: obtención de AIC de la población actual
  END DO
END
```

Fig. 1
Pseudocódigo del AG

con un menor valor AIC. El modelo obtenido es:

$$\begin{aligned} T_{\mathrm{AIC}}(s) = \; & 1{,}812 - 2{,}749 \cdot 10^{-12} N^3 \\ & +1{,}974 \cdot 10^{-7} N^2 - 29{,}65 \tfrac{1}{NB} \\ & -3{,}239 \tfrac{1}{Q} + 48{,}49 \tfrac{1}{NBQ} + 7{,}099 \tfrac{1}{QP} \\ & -7{,}903 \cdot 10^{-7} \tfrac{N^2}{NB} + 8{,}228 \cdot 10^{-8} \tfrac{N^2}{Q} \\ & +4{,}059 \cdot 10^{-12} \tfrac{N^3}{P} - 2{,}706 \cdot 10^{-7} \tfrac{N^2}{P} \\ & +7{,}940 \cdot 10^{-10} \tfrac{N^3}{NBP} \\ & -5{,}817 \cdot 10^{-7} \tfrac{N^2}{NBP} + 1{,}613 \cdot 10^{-10} \tfrac{N^3}{QP} \\ & -1{,}666 \cdot 10^{-10} \tfrac{N^3}{NBQP} \end{aligned} \tag{5}$$

El tiempo de ejecución de este proceso exhaustivo de búsqueda es típicamente superior a las 48 horas en un PC de sobremesa actual.

El AG descrito en la sección anterior se ha utilizado para obtener un modelo de rendimiento de HPL. Dado que el AG no garantiza, en ningún caso, obtener la solución óptima, se han realizado diferentes ejecuciones bajo diferentes configuraciones (μ, ν, ϕ) y se han analizado conjuntamente los resultados obtenidos. En primer lugar, se han considerado diferentes condiciones de terminación (ϕ=10, 20 y 30). Para cada condición de terminación se ha ejecutado el AG para una población μ=16, 32, 64 y 128 individuos, y para cada tamaño de población se han considerado ν=1, 2 y 4 mutaciones.

Las figuras 2, 3 y 4 representan la evolución del mejor individuo (el modelo con un menor valor AIC) durante la ejecución del AG para las diferentes condiciones de terminación, donde $\mathrm{AIC}^{\mathrm{Abs}}$ representa el valor óptimo AIC para este conjunto de modelos candidato.

La tabla I muestra, para cada condición de terminación (ϕ), el tiempo de ejecución total (T_{tot}) y el número de iteraciones (ciclos selección/reproducción) de las 12 configuraciones (μ, ν) del AG. Los valores temporales han sido medidos en el mismo sistema en el que se ha realizado la búsqueda exhaustiva. El diferente coste temporal está directamente asociado al número de iteraciones que es necesario ejecutar para alcanzar la condición de terminación. En la última columna se muestra el valor medio AIC ($\overline{\mathrm{AIC}}$) obtenido con cada condición de terminación.

TABLA I
Características de la ejecución del AG para diferentes condiciones de terminación

ϕ	T_{tot} (min)	# iteraciones	$\overline{\mathrm{AIC}}$
10	8,9	314	14245 ± 2
20	12,7	485	$14243,7 \pm 0{,}7$
30	20,6	751	$14243,3 \pm 0{,}4$

A partir de estos resultados, se observa que, en este caso particular, un valor ϕ=30 es suficiente para que el algoritmo alcance, en la mayoría de los casos, un comportamiento asintótico hacia el resultado óptimo. Esta condición de terminación es la más costosa computacionalmente, ya que es necesario un mayor número de iteraciones para alcanzar la condición de terminación, pero, en cualquiera de los casos, la ejecución de las diferentes configuraciones (μ, ν) consume menos del 1 % del tiempo empleado por la búsqueda exhaustiva.

La figura 5 muestra, considerando únicamente la condición de terminación ϕ=30, el valor AIC obtenido para las diferentes configuraciones (μ, ν). En esta gráfica se puede observar que el AG alcanza en la mitad de las ejecuciones el resultado óptimo ($\mathrm{AIC}^{\mathrm{Abs}}$=14,242,97).

La figura 6 muestra un histograma de las apariciones de cada uno de los 24 términos generados por la lista `LI` en los modelos obtenidos por estas ejecuciones del AG. Como se puede observar en esta gráfica, hay 8 términos que siempre están presentes en las soluciones propuestas por el AG. En cualquier caso, los 16 términos con mayor número de apariciones se corresponden con los términos del modelo obtenido con una búsqueda exhaustiva (estos términos están marcados con el símbolo '*'). Esta gráfica proporciona al usuario información acerca de los términos con mayor relevancia en el comportamiento de la aplicación. De este modo, el AG también proporciona un mecanismo para eliminar aquellos términos cuya influencia en comportamiento es despreciable, y reducir el tiempo de una búsqueda exhaustiva del mejor modelo candidato.

VI. Conclusiones

El uso de algoritmos genéticos es una buena aproximación para reducir el tiempo de obtención de mo-

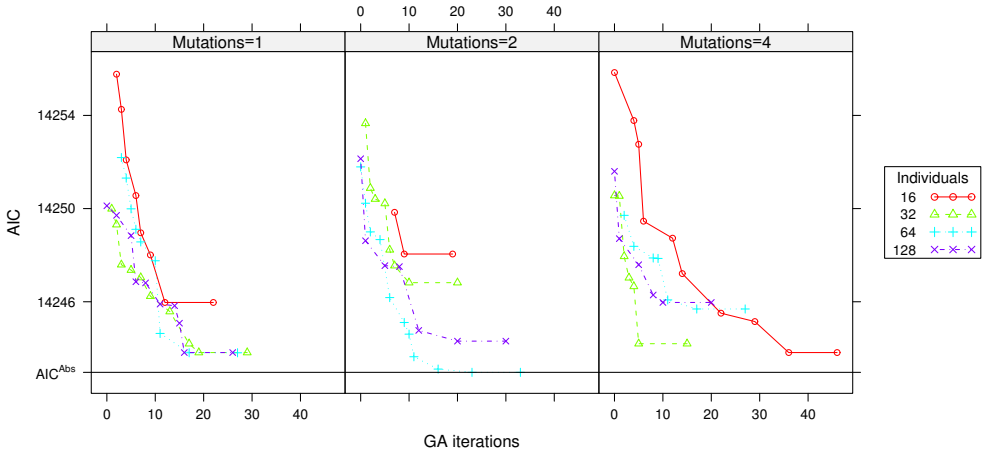

Fig. 2

Evolución del mejor valor AIC durante las ejecuciones del AG con ϕ=10

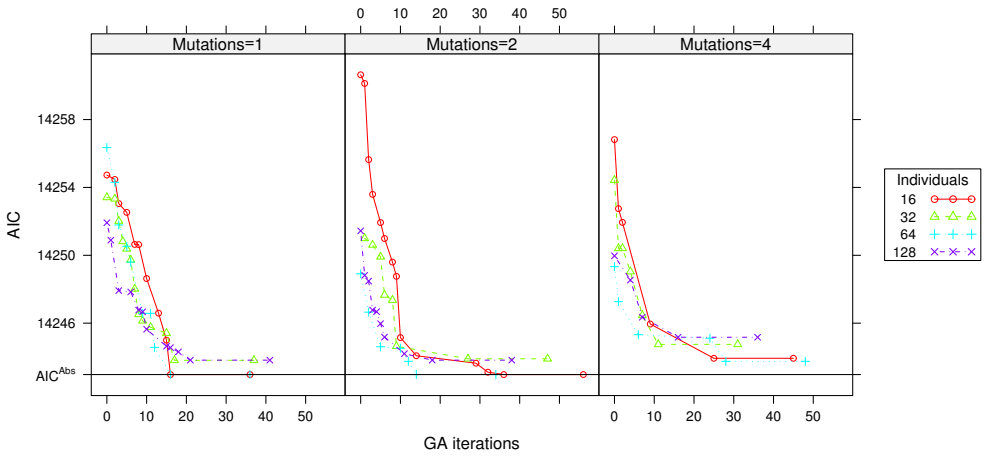

Fig. 3

Evolución del mejor valor AIC durante las ejecuciones del AG con ϕ=20

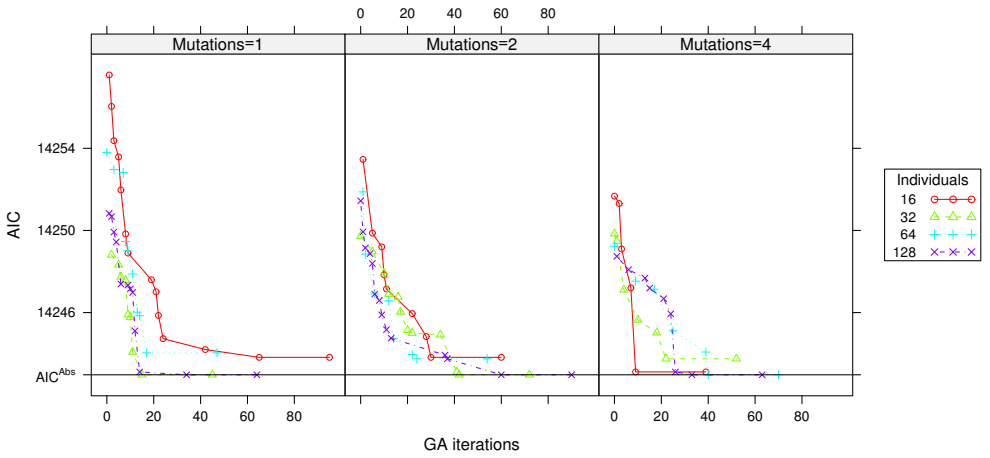

Fig. 4

Evolución del mejor valor AIC durante las ejecuciones del AG con ϕ=30

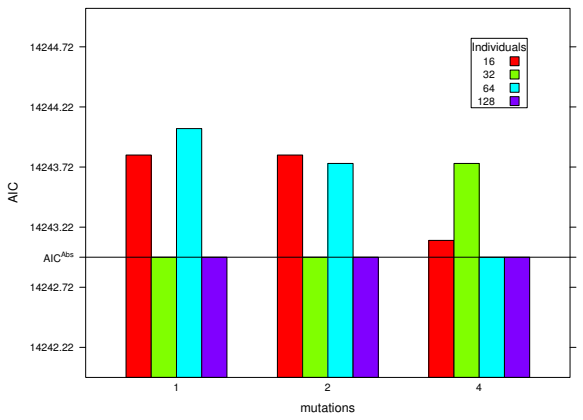

Fig. 5

Valor AIC obtenido en las diferentes configuraciones experimentales del AG

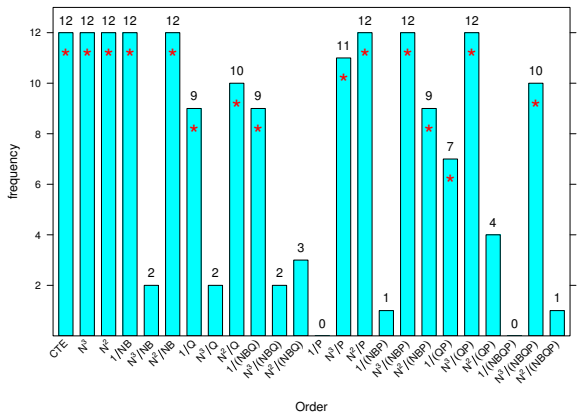

Fig. 6

Número de apariciones en las diferentes ejecuciones del AG de los términos considerados en el análisis

delos de rendimiento dentro del proceso de modelado del entorno de análisis TIA. En este trabajo se ha presentado una implementación de algoritmo genético, que ha sido evaluado usando el conocido benckmark HPL. Los experimentos realizados muestran que, aunque no se garantiza la obtención del resultado óptimo, este mecanismo de búsqueda es una alternativa muy útil desde el punto de vista de la selección de modelos y requiere un tiempo de ejecución mucho menor (del orden del 1 %) que una búsqueda exhaustiva en el conjunto de modelos candidato.

Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Educación y Ciencia de España, con los proyectos TIN 2010-17541 y TIN2011-24598, por el Gobierno de Canarias, con el proyecto ProID20100222, y por las redes HiPEAC-2 y CAPAP-H.

Referencias

[1] Diego R. Martínez, Vicente Blanco, Marcos Boullón, José Carlos Cabaleiro, Casiano Rodríguez, and Francisco F. Rivera, "Software tools for performance modeling of parallel programs," in *IEEE International Parallel and Distributed Processing Symposium*, 2007, IPDPS.

[2] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, Springer, 2003.

[3] H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, pp. 716–723, 1974.

[4] John Aldrich, "R. A. Fisher and the making of maximum likelihood 1912–1922," *Statistical Science*, vol. 12, no. 3, pp. 162–176, 1997.

[5] Diego R. Martínez, Vicente Blanco, Marcos Boullón, José Carlos Cabaleiro, and Tomás F. Pena, "Analytical performance models of parallel programs in clusters," in *Parallel Computing: Architectures, Algorithms and Applications*, 2007, ParCo.

[6] V. Blanco, J. A. González, C. León, C. Rodríguez, G. Rodríguez, and M. Printista, "Predicting the performance of parallel programs," *Parallel Computing*, vol. 30, pp. 337–356, 2004.

[7] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2009.

[8] Diego R. Martínez, Julio L. Albín, José Carlos Cabaleiro, Tomás F. Pena, Francisco F. Rivera, and Vicente Blanco, "El criterio de información de Akaike en la obtención de modelos estadísticos de rendimiento," in *XX Jornadas de Paralelismo*, 2009.

[9] Kenneth P. Burnham and David R. Anderson, *Model Selection and Multimodel Inference. A Practical Information-Theoretic Approach*, Spring Science + Bussiness Media, LLC, 2002.

[10] HPL—A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers, ," http://www.netlib.org/benchmark/hpl/, 2010.

[11] Diego R. Martínez, Julio L. Albín, Tomás F. Pena, José Carlos Cabaleiro, Francisco F. Rivera, and Vicente Blanco, "Using accurate AIC-based performance models to improve the scheduling of parallel applications," *Journal of Supercomputing*, vol. 58, no. 3, pp. 332–340, 2011.