

Optimización de los Parámetros de Control del Algoritmo Genético Simple usando Regresión de Vectores Soporte

Mauricio Orozco Alzate
Grupo de Control y Procesamiento
Digital de Señales
Universidad Nacional de Colombia
Manizales, Caldas
Email: mauricio_orozco@unalmzl.edu.co

Resumen— Se propone un meta-algoritmo genético modificado con una etapa de regresión de vectores soporte (SVR) para predecir el desempeño de un algoritmo genético simple (SGA). La SVR modela el desempeño como una función de los parámetros de control, de este modo se obtiene no sólo el ajuste automático de los parámetros óptimos sino también una explicación funcional de la compleja interacción entre ellos.

I. INTRODUCCIÓN

Aunque el SGA es un método robusto para resolver un amplio rango de problemas, la solución de un problema específico requiere un conjunto de parámetros específico que garantice un desempeño satisfactorio. La optimización de estos parámetros es una tarea que consume gran cantidad de tiempo y que demanda a menudo la intervención subjetiva de un experto. Existen valores sugeridos para sintonizar estos parámetros, sin embargo, estos valores pueden no ser los óptimos para todo tipo de aplicaciones.

También es común sintonizar cada parámetro por separado, sin tener en cuenta la influencia de cada uno sobre los demás. Un método de auto-sintonización denominado meta-algoritmo ha demostrado ser una forma acertada de resolver este problema; no obstante, la ejecución del meta-algoritmo es computacionalmente costosa y no entrega un modelo de la influencia de los parámetros de control sobre el desempeño del algoritmo genético.

En este artículo presentamos un método de meta-algoritmo al cual se le ha agregado una etapa de regresión de vectores soporte. La superficie de regresión obtenida con la SVR explica la compleja interacción de los parámetros y reduce la demanda computacional de la ejecución del meta-algoritmo.

Este artículo está organizado como sigue: en la próxima sección se hace un breve repaso de los conceptos y la notación sobre el Algoritmo Genético Simple. En la sección III se introduce al problema de optimización de parámetros para algoritmos genéticos y se discuten trabajos previos en esta área. En la sección IV se propone una modificación de la estrategia de meta-algoritmo para optimización de parámetros, mediante la incorporación de una Regresión de Vectores Soporte (SVR) que predice y modela el desempeño del SGA. En la sección V

se presenta una serie de experimentos propuestos para medir el desempeño del SGA en la optimización de funciones.

II. EL ALGORITMO GENÉTICO SIMPLE

El Algoritmo Genético Simple (SGA) es un caso especial de búsqueda heurística aleatoria [1], para el cual el espacio de búsqueda Ω corresponde a:

$$\Omega = \underbrace{\mathcal{Z}_2 \times \cdots \times \mathcal{Z}_2}_{\ell \text{ veces}} \quad (1)$$

donde \mathcal{Z}_2 es el conjunto de los enteros módulo 2 [2]. La expresión (1) corresponde exactamente a la representación binaria de los enteros en el intervalo $[0, 2^\ell)$; por tanto, Ω está compuesto por $n = 2^\ell$ cadenas binarias de longitud ℓ . La longitud ℓ se conoce como *longitud del cromosoma*.

La exploración en Ω se realiza mediante la aplicación de mecanismos denominados *operadores*, que actúan sobre los elementos de un subconjunto P_i del espacio de búsqueda. Cada colección P_i recibe el nombre de *i-ésima población* o *i-ésima generación*, e.g.

$$P_i = \underbrace{\{ \overbrace{10 \cdots 01}^{\ell \text{ bits}}, 11 \cdots 00, \dots, 11 \cdots 10 \}}_{r \text{ individuos}} \quad (2)$$

El parámetro r en (2) representa el *tamaño de la población*, i.e. la cardinalidad de cada generación P_i . A medida que r se incrementa se posee una mayor diversidad genética para la exploración sobre Ω .

A. Operadores Genéticos

Los operadores genéticos son los mecanismos evolutivos que generan la secuencia de poblaciones

$$P_0 \xrightarrow{\tau} P_1 \xrightarrow{\tau} P_2 \xrightarrow{\tau} \dots \quad (3)$$

donde τ es una *regla de transición* que se implementa mediante la combinación de tres operadores básicos: selección, mutación y cruzamiento.

1) *Selección*: Es el proceso por el cual se eligen miembros de una población de acuerdo con una *función de adaptabilidad* $f : \Omega \rightarrow \mathbb{R}$ que se desea maximizar. Los individuos elegidos serán los *padres* de la próxima generación.

El uso de la función f determina un esquema de selección. Los principales esquemas son: selección proporcional, selección jerárquica y selección por torneo [1], [3].

2) *Mutación*: Consiste en alterar aleatoriamente cada gen (bit) con una probabilidad típicamente pequeña [4]. Este mecanismo asegura que la búsqueda heurística pueda realizarse en puntos de Ω que no pueden ser explorados mediante la reproducción de la población inicial.

3) *Cruzamiento*: Los cromosomas de los individuos padres intercambian bits en aquellas posiciones donde una máscara de cruzamiento es 1, produciendo dos nuevos cromosomas denominados *hijos*.

El cruzamiento no se aplica a todas las parejas seleccionadas, su ejecución está sujeta a una distribución de probabilidad. Los tipos clásicos de cruzamiento son: cruzamiento en un punto y cruzamiento uniforme.

B. Parámetros de Control

Los parámetros básicos del SGA que deben ser ajustados son: el tamaño de la población (r); la probabilidad o tasa de mutación, denotada por $\mu \in [0, 1]$; y la probabilidad o tasa de cruzamiento, denotada por $\chi \in [0, 1]$. Estos parámetros no son independientes [5] y tienen gran influencia sobre el desempeño del algoritmo genético.

III. OPTIMIZACIÓN DE LOS PARÁMETROS DE CONTROL

El problema de encontrar los parámetros de control óptimos ha sido estudiado ampliamente durante las últimas tres décadas. De Jong [6] determinó experimentalmente que se obtenía un desempeño aceptable en la optimización de funciones, ajustando los parámetros en los siguientes valores: $r = 50$, $\mu = 0.001$, $\chi = 0.6$. Grefenstette [7] usó un meta-algoritmo parametrizado con los valores propuestos por De Jong y obtuvo mejores resultados en línea con los parámetros de control: $r = 30$, $\mu = 0.01$ y $\chi = 0.95$. Schaffer *et al.* [8] hicieron esfuerzos adicionales en la determinación experimental de parámetros óptimos; sus resultados recomiendan la elección de los parámetros dentro de los rangos: $r \in [20, 30]$, $\mu \in [0.005, 0.01]$ y $\chi \in [0.75, 0.95]$.

Bramlette [9] y Wu y Chow [10] usaron también el método de meta-algoritmo. El primero introdujo modificaciones a los métodos de selección de poblaciones iniciales y a los operadores de mutación para mejorar el desempeño en la optimización de funciones. Los otros aplicaron el meta-algoritmo para optimizar los parámetros de algoritmos genéticos para problemas de optimización mixtos no lineales.

También se ha llevado a cabo una considerable cantidad de estudios teóricos y experimentales sobre el efecto de cada parámetro de control por separado, por ejemplo acerca de la influencia del tamaño de la población: [11]–[14], la tasa de mutación: [15]–[19] y la tasa de cruzamiento: [20]–[24].

Eiben, Hinterding y Michalewicz [5] sugieren que los parámetros de control deben ser dinámicos, es decir, cambiar durante la ejecución del algoritmo de acuerdo con la etapa en la que se encuentre el proceso evolutivo. No obstante, aún existe la necesidad de buenos métodos de sintonización de parámetros ya que en un gran número de aplicaciones evolutivas se continúan usando parámetros estáticos. En [25] se propone una técnica de auto-sintonización denominada “sin parámetros”, que consiste en eliminar del diseño del GA las tasas de mutación y cruzamiento y el parámetro del tamaño de la población.

Cicirello y Smith [26] tomaron el modelo de meta-algoritmo para optimizar los parámetros de control de un algoritmo genético aplicado a la solución del problema de la mayor subgráfica común. En dicho trabajo se introdujo una red neuronal para la evaluación de la función de adaptabilidad, con el fin de simplificar la ejecución del meta-algoritmo, entrenando la red para predecir el desempeño del algoritmo genético.

En [27] se realizó un estudio estadístico usando análisis de varianza (ANOVA) para encontrar los parámetros que tienen una mayor influencia estadística sobre el comportamiento y el desempeño del GA, teniendo en cuenta las interacciones que se presentan entre ellos.

En este artículo presentamos una modificación del método de optimización de parámetros de control usando meta-algoritmo. Hemos incluido una etapa de Regresión de Vectores Soporte para modelar las interacciones de los parámetros de control y predecir el desempeño del GA ante diferentes combinaciones $\{r, \mu, \chi\}$. En contraste con el trabajo [26], además de lograr una simplificación del meta-algoritmo, obtendremos un modelo de la interacción de los parámetros.

A. Funciones de Evaluación Comparativa

Para evaluar el desempeño y comportamiento de un algoritmo genético existe un conjunto de funciones de prueba estándar, denominadas funciones de evaluación comparativa. En [28] se presenta una revisión de las principales funciones de evaluación que deberían ser consideradas en un estudio de sintonización de parámetros de algoritmos genéticos. En la tabla I se muestran 8 funciones de optimización estándar para realizar las pruebas.

IV. SVR PARA PREDECIR Y MODELAR LA INFLUENCIA DE LOS PARÁMETROS DE CONTROL

Trabajar con un meta-algoritmo consiste en correr un GA de alto nivel que busca un conjunto de parámetros óptimos para un GA de bajo nivel. El GA de alto nivel, o meta-GA, corre sobre una población cuyos individuos son codificaciones de los parámetros de control, mientras que el GA de bajo nivel es un GA regular que usa los parámetros encontrados por el GA de alto nivel. Sin embargo, un inconveniente con este método es su costo computacional [26]. Hemos incluido una SVR en el SGA de alto nivel. La superficie de regresión será un modelo de las interacciones de los parámetros.

TABLA I
FUNCIONES DE PRUEBA

Nombre	Función	Límites
F1	$f_1 = \sum_{i=1}^2 x_i^2$	$-5.12 \leq x_i \leq 5.12$
F2	$f_2 = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$	$-2.048 \leq x_i \leq 2.048$
F3	$f_3 = \sum_{i=1}^5 \text{int.}(x_i)$	$-5.12 \leq x_i \leq 5.12$
F4	$f_4 = \sum_{i=1}^{30} (ix_i^4 + \text{Gauss}(0, 1))$	$-1.28 \leq x_i \leq 1.28$
F5	$f_5(x_i) = 0.002 + \sum_{j=1}^{25} \left(\frac{1}{j} + \sum_{i=1} 2(x_i - a_{ij})^6 \right)$	$-65536 \leq x_i \leq 65536$
F6	$f_6 = 10V + \sum_{i=1}^{10} (-x_i \sin \sqrt{ x_i })$, $V = 4189.829101$	$-500 \leq x_i \leq 500$
F7	$f_7 = 20A + \sum_{i=1}^{20} (x_i^2 - 10 \cos(2\pi x_i))$, $A = 10$	$-5.12 \leq x_i \leq 5.12$
F8	$f_8 = 1 + \sum_{i=1}^{10} \left(\frac{x_i^2}{4000} - \prod_{i=1}^{10} \left(\cos \left(\frac{x_i}{\sqrt{i}} \right) \right) \right)$	$-600 \leq x_i \leq 600$

La función de adaptabilidad del meta-algoritmo usa la SVR para predecir el desempeño del SGA, dado un conjunto de parámetros $\Pi = \{r, \mu, \chi\}$. Es computacionalmente menos costoso correr el SGA unas cuantas veces con diferentes combinaciones de $\{r, \mu, \chi\}$ y crear un modelo de la interacción e influencia de los parámetros. Se necesita crear un conjunto de entrenamiento de la forma:

$$\mathcal{T} = [A|y] \quad (4)$$

donde,

$$A = \begin{bmatrix} r_0 & \mu_0 & \chi_0 \\ r_1 & \mu_1 & \chi_1 \\ \vdots & \vdots & \vdots \\ r_n & \mu_n & \chi_n \end{bmatrix}, \quad y = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} \quad (5)$$

Se desea aproximar $y \in \mathbb{R}^m$ mediante una función de A de la forma:

$$y \approx Aw + be \quad (6)$$

El resultado de la SVR es una superficie de regresión [29] que determina los valores aproximados para y :

$$\hat{y} = x'w + b \quad (7)$$

A. Creación del Conjunto de Entrenamiento

Para cada una de las funciones de prueba, se debe ejecutar el SGA n veces, cada una de ellas para una combinación específica de Π . Para cada una de las ejecuciones se medirá el tiempo de convergencia del GA. Los valores y_i que debe aproximar la SVR serán:

$$y = 1 - \frac{t_{\text{convergencia}}}{t_{\text{max}}} \quad (8)$$

donde t_{max} es el máximo tiempo de convergencia observado, después de n ejecuciones del algoritmo. Los valores de entrenamiento son generados aleatoriamente, distribuidos en los siguientes rangos: tamaño de la población de 10 a 160; valores de tasa de mutación entre 1 y 2^{-15} y tasa de cruzamiento entre 0.25 y 1.

B. Meta-algoritmo

El tamaño de la población se codificará usando 4 bits, las 16 cadenas representarán tamaños de población de 10 a 160 en incrementos de 10. La tasa de mutación con los próximos 4 bits representando los valores de la forma 2^{-i} para $i = 0, \dots, 15$ y tasa de cruzamiento entre 0.25 y 1 en incrementos de 0.05. El individuo completo se crea al adjuntar las cadenas binarias correspondientes a los tres parámetros. El meta-algoritmo evalúa los parámetros sobre la superficie de regresión de la SVR.

V. EXPERIMENTOS PROPUESTOS

Para validar este método, se propone encontrar los parámetros óptimos para el SGA en la solución de las funciones mostradas en la tabla I. Los resultados obtenidos con este método se compararán con el tiempo de convergencia resultante al utilizar los parámetros de control propuestos en [6], [7] y [8].

REFERENCIAS

- [1] M. D. Vose, *The Simple Genetic Algorithm: Foundations and Theory*. Cambridge, MA: The MIT Press, 1999.
- [2] T. Kaczynski, K. Mischaikow, and M. Mrozek, *Algebraic Topology: A Computational Approach*. Disponible en: <http://books.pdx.net/>, 2000, ch. 1.
- [3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.
- [4] D. Beasley, D. R. Bull, and R. R. Martin, "An overview of genetic algorithms: Part 1, fundamentals," *University Computing*, vol. 15, no. 2, pp. 58–69, 1993, disponible en: citeseer.ist.psu.edu/beasley93overview.html.
- [5] Ágoston E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 124–141, 1999.
- [6] K. D. Jong, "The Analysis of the Behavior of a Class of Genetic Adaptive Systems," Ph.D. dissertation, Department of Computer Science, University of Michigan, Ann Arbor, Michigan, 1975.
- [7] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Trans. Syst., Man, Cybern.*, vol. 16, no. 1, pp. 122–128, 1986.
- [8] J. D. Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das, "A study of control parameters affecting online performance of genetic algorithms for function optimization," in *Proceedings of the 3rd International Conference on Genetic Algorithms*, J. D. Schaffer, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 51–60.

- [9] M. F. Bramlette, "Initialization, mutation and selection methods in genetic algorithms for function optimization," in *Proceedings of the Fourth International Conference on Genetic Algorithms*, R. K. Belew and L. B. Booker, Eds. San Mateo, CA: Morgan Kaufmann, 1991, pp. 100–107.
- [10] S. J. Wu and P. T. Chow, "Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization," *Engineering Optimization*, vol. 24, no. 2, pp. 137–159, 1995.
- [11] D. E. Goldberg, K. Deb, and J. H. Clark, "Genetic algorithms, noise and the sizing of populations," *Complex Systems*, vol. 6, pp. 333–362, 1992.
- [12] J. Arabas, Z. Michalewicz, and J. Mulawka, "Gavaps – a genetic algorithm with varying population size," in *Proceedings of the 1st IEEE Conference on Evolutionary Computation*. IEEE Press, 1994, pp. 73–78.
- [13] G. R. Harik, E. Cantú-Paz, D. E. Goldberg, and B. L. Miller, "The glambert's ruin problem, genetic algorithms and the sizing of populations," in *Proceedings of 1997 IEEE Interational Conference on Evolutionary Computation*, T. Bäck, Ed. New York: IEEE Press, 1997, pp. 7–12.
- [14] S. Gotshall and B. Rylander, "Optimal population size and the genetic algorithm," *WSEAS Transactions on Computers*, 2002.
- [15] T. Fogarty, "Varying the probability of mutation in the genetic algorithm," in *Proceedings of the 3rd International Conference on Genetic Algorithms*, J. D. Schaffer, Ed. Morgan Kaufmann, 1989, pp. 104–109.
- [16] J. Hesser and R. Männer, "Towards an optimal mutation probability for genetic algorithms," in *Proceedings of the 1st Conference on Parallel Problem Solving from Nature*, H.-P. Schwefel and R. Männer, Eds. Springer-Verlag, 1991, pp. 23–32, number 496 in Lecture Notes in Computer Science.
- [17] H. Mühlenbein, "How genetic algorithms really work: I. mutation and hillclimbing," in *Proceedings of the 2nd Conference on Parallel Problem Solving from Nature*, R. Männer and B. Manderick, Eds. Amsterdam, The Netherlands: Elsevier Science, 1992, pp. 15–25.
- [18] T. Bäck, "Optimal mutation rates in genetic search," in *Proceedings of the Fifth International Conference on Genetic Algorithms*, S. Forrest, Ed. San Mateo, CA: Morgan Kaufmann, 1993, pp. 2–8.
- [19] J. Smith and T. Fogarty, "Self-adaptation of mutation rates in steady-state genetic algorithm," in *Proceedings of the 3rd IEEE Conference on Evolutionary Computation*, 1996, pp. 318–323.
- [20] J. D. Schaffer and A. Morishima, "An adaptive crossover distribution mechanism for genetic algorithms," in *Proceedings of the 2nd International Conference on Genetic Algorithms and Their Applications*, J. J. Grefenstette, Ed. Lawrence Erlbaum Associates, 1987, pp. 36–40.
- [21] T. White and F. Oppacher, "Adaptive crossover using automata," in *Proceedings of the 3rd Conference on Parallel Problem Solving from Nature*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. Springer-Verlag, 1994, pp. 229–238, number 866 in Lecture Notes in Computer Science.
- [22] W. Spears, "Adapting crossover in evolutionary algorithms," in *Proceedings of the 4th Annual Conference on Evolutionary Programming*, J. R. McDonnell, R. G. Reynolds, and D. B. Fogel, Eds. MIT Press, 1995, pp. 367–384.
- [23] J. Lis and M. Lis, "Self-adapting parallel genetic algorithm with the dynamic mutation probability, crossover rate and population size," in *Proceedings of the 1st Polish National Conference on Evolutionary Computation*, J. Arabas, Ed., Oficyna Wydawnica Politechniki Warszawskiej, 1996, pp. 324–329.
- [24] A. E. Eiben, I. G. Sprinkhuizen-Kuyper, and B. A. Thijssen, "competing crossvers in adaptive ga framework," in *proceedings of the 5th IEEE Conference on Evolutionary Computation*. IEEE Press, 1998, pp. 787–792.
- [25] F. G. Lobo, "The parameter-less genetic algorithm: rational and automated parameter selection for simplified genetic algorithm operation," Ph.D. dissertation, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, 2000. [Online]. Available: <http://w3.ualg.pt/~flobo/phd/thesis.ps.gz>
- [26] V. Cicirello and S. Smith, "Modeling GA performance for control parameter optimization," in *GECCO-2000: Proceedings of the Genetic and Evolutionary Computation Conference*, July 2000.
- [27] I. Rojas, J. González, H. Pomares, J. J. Merelo, P. A. Castillo, and G. Romero, "Statistical analysis of the main parameters involved in the design of a genetic algorithm," *IEEE Trans. Syst., Man, Cybern. C*, vol. 32, no. 1, pp. 31–37, 2002.
- [28] J. Digalakis and K. Margaritis, "An experimental study of benchmarking functions for evolutionary algorithms," *International Journal of Computer Mathematics*, vol. 79, no. 4, pp. 403–416, April 2002. [Online]. Available: citeseer.ist.psu.edu/digalakis02experimental.html
- [29] D. R. Musicant and A. Feinberg, "Active set support vector regression," *IEEE Trans. Neural Networks*, vol. 15, no. 2, pp. 268–275, 2004.