Algoritmos Evolutivos - Estudio de Escalabilidad sobre el Problema 3-SAT

Hugo Alfredo Alfonso
Facultad de Ingeniería
Universidad Nacional de La Pampa
Calle 9 esq. 110, General Pico
La Pampa, Argentina
email: alfonsoh@ing.unlpam.edu.ar

Resumen

En este trabajo se investiga la influencia del tamaño del problema en la performance de los Algoritmos Evolutivos (AEs), utilizados para resolver un conocido problema NP-completo: el 3-SAT. Para ello se ha realizado una amplia recopilación de diferentes variantes de AEs, desarrollados para mejorar la calidad de los resultados. También, se describen los AEs usados para analizar el comportamiento de los mismos frente a problemas de tamaño creciente, tomados de benchmarks internacionales. Este estudio también incluye el análisis de la incorporación de distribución y paralelismo en la resolución del problema, como caminos alternativos para resolverlo en menor tiempo.

1. Introducción

El *Problema de Satisfacción de cláusulas lógicas (SAT)* es un paradigmático problema NP-completo [Coo71], que también cuenta con un muy importante interés práctico, pues se aplica en problemas de razonamiento, diagnóstico y planificación [KS92], interpretación de imágenes [RM89], entre otros. El Problema SAT está basado sobre un conjunto de variables lógicas $x_1, x_2, \ldots, x_n$ y una fórmula lógica $f : \mathcal{B}^n \rightarrow \mathcal{B}, \mathcal{B} = \{0, 1\}$. Siendo la cuestión a resolver la existencia de una asignación de valores posibles $x = (x_1, x_2, \ldots, x_n) \in \mathcal{B}^n$ que satisfagan dichas fórmula, es decir que $f(x) = 1$. Una instancia SAT se denomina *satisfecha* si existe tal valor x, e *insatisfecha* en caso contrario. La fórmula f está en *Forma Normal Conjuntiva - (Conjunctive Normal Norm-CNF)* si $f(x) = c_1(x) \wedge \ldots \wedge c_m(x)$, donde cada una de las cláusulas c_i es una disjunción de literales, y un literal es una variable o su negación. Se puede asumir que las instancias SAT están expresadas en CNF sin pérdida de generalidad [S.68], y las clases l-SAT contiene las fórmulas cuyas cláusulas contienen exactamente l literales distintos. La clase de problemas 2-SAT es resoluble en tiempo polinomial, pero las clases l-SAT es NP-completa para $l \geq 3$ [GJ79].

Debido a la importancia teórica y práctica del problema SAT, y particularmente el 3-SAT, este ha sido ampliamente estudiado y se han desarrollado muchos algoritmos exactos y heurísti-

cos. De acuerdo a la compilación que realizase Jun Gu et al. [GPFW96] varios de los algoritmos dan una respuesta definitiva a la pregunta inicialmente planteada, si cada instancia del problema es satisfecha o no, pero tiene una complejidad exponencial al peor caso, a menos que $P = NP$. Los algoritmos heurísticos pueden encontrar soluciones para instancias satisfechas rápidamente, pero ellos no pueden garantizar el dar una respuesta para todas las instancias del problema.

Los *Algoritmos Evolutivos (AEs)* son algoritmos heurísticos que se han aplicado al problema SAT, como así también a otros tantos problemas NP-completos. Algunos resultados negativos ponen en duda la capacidad de los AEs para resolver el problema SAT. De Jong y Spears [DS89] propusieron un *Algoritmo Genético (AG)* clásico para el SAT y observaron que un AG no podía obtener mejores resultados que los algoritmos específicos perfectamente adecuados al problema. Este resultado fue confirmado por las experiencias de Fleurent y Ferland [FF93], quienes informaron un escaso rendimiento del AG puro cuando es comparado con búsqueda local. Posteriormente, Rana y Whitley [RW98] demostraron que los AGs clásicos no eran lo suficientemente apropiados para la función de fitness del problema de maximización *MAXSAT*, el cual cuenta el número de cláusulas satisfechas, por que el dominio correspondiente al problema presenta engañosos esquemas de información de baja importancia y el espacio de búsqueda tiende a ser homogéneo. Sin embargo, estudios más recientes muestran que los AEs pueden obtener buenos resultados para el problema SAT si se le incorporan técnicas adicionales con el objetivo de mejorar los AGs clásicos. Estas técnicas incluyen el uso de funciones de fitness adaptativas, operadores de variación específicos al problema, optimización local, paralelización, etc. [DS89] [BEV98] [EvdH97] [FPS98] [MR99].

2. Detalle de Experimentos

Los experimentos se han realizado sobre un conjunto de doce instancias que poseen soluciones conocidas, ello permite evaluar a los algoritmos analizando el porcentaje de experiencias que han alcanzado la solución óptima. Este conjunto de instancias se ha generado usando el generador `mkcnf.c` desarrollado por Allen van Gelder y que se basa en el algoritmo `mwff.c` de Bart Selman [SLM92].

El Cuadro 1 nos muestra las doce instancias con el correspondiente tamaño del problema - dado por la cantidad de variables a considerar en la fórmula n - y el número de cláusulas que deben ser satisfechas m.

Instancias	**Tamaño del Problema** n	**Número de Clausulas** m
1,2,3	30	129
4,5,6	40	172
7,8,9	50	215
10,11,12	100	430

Cuadro 1: Instancias de Testeo del 3-SAT

El generador `mkcnf.c` puede forzar la generación de fórmulas duras, que pertenecen a la fase de transición donde $m = 4{,}3 * n$ y para las cuales existe solución factible (ver Figura 1).

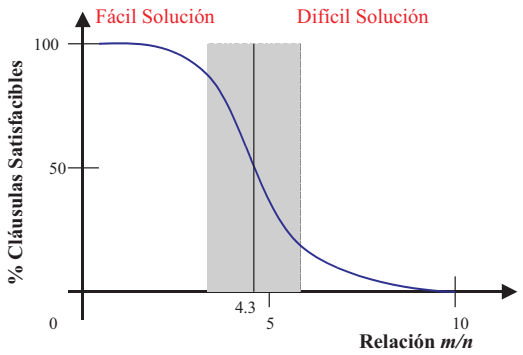

Figura 1: Punto de Cruce y Región Crítica

Las variables que permiten medir la performance de los algoritmos son: SR (Success Rate) y AES (Average number of Evaluations to Solution) las cuales han sido reportadas en diversos trabajos del área [JK] [MR99] [GMR02]:

SR indica el porcentaje del total de experimentos en los cuales la solución óptima ha sido alcanzada.

AES indica el número de evaluaciones necesarias para terminar aquellos experimentos que alcanzan la solución óptima.

Es claro que, si $SR = 0$ entonces AES no está definido.

El Algoritmo Evolutivo utilizado para realizar las pruebas pertenece al paquete MALLBA que fue desarrollado como parte del Proyecto Mallba [AAB+01] integrado por investigadores de tres Universidades Españolas: **MÁ**laga, **L**a **L**aguna y **BA**rcelona.

En el proyecto MALLBA se ataca la resolución de diversos problemas de optimización combinatoria usando esqueletos algorítmicos desarrollados en C++, para correr sobre una cluster de PCs conectadas en una red geográficamente distribuida. Además, MALLBA ofrece tres familias de métodos de resolución genérica de problemas: exactos, heurísticos e híbridos. Y, para cada uno de estos métodos ofrece tres variantes de implementación: secuencial, LAN y WAN.

Para la experimentación se han considerado las 12 instancias descritas y para cada una de ellas se han realizado series de 50 ejecuciones independientes. Cada instancia ha sido tratada con tres algoritmos. Ellos son:

- AE - Algoritmo Evolutivo
- AEd - Algoritmo Evolutivo Distribuido
- AEp - Algoritmo Evolutivo Paralelo

Estos algoritmos usan la función de fitness SAW -*stepwise adaptation fitness*-, propuesta por Back [EvdH97], que consiste en utilizar una función de fitness adecuada por medio de pesos. Dichos pesos permiten identificar soluciones distintas que satisfacen la misma cantidad de cláusulas, las cuales eran imposibles de distinguir con la función de fitness original.

Para el AE Distribuido (AEd) se usa un modelo de 4 procesos asíncronos trabajando sobre el mismo computador. El modelo isla realiza una migración cada 25 generaciones, migrando 5

individuos por vez según una topología de anillo. El grupo de individuos migrados que cada isla recibe es evaluado para ser insertados en la población actual si algunos de los individuos recibidos es mejor que los individuos actuales de la isla.

Para el AE Paralelo (AEp) se usa el mismo modelo descrito para AEd, pero trabajando en 4 computadoras diferentes.

El tamaño de la población de cada isla se fija en un cuarto de la población total para mantener fijo el tamaño total de la misma.

3. Resultados Preliminares

A la luz de los resultados ya evaluados se puede indicar que la calidad de los mismo se mejora al utilizar una función de fitness que permita distinguir adecuadamente las soluciones más allá de la cantidad de cláusulas que satisfagan. Por otra parte, al hacer la evolución en forma distribuida, en uno o varios procesadores se mejora significativamente la calidad de los resultados, contabilizando la cantidad de casos resueltos, manteniéndose un similar número de evaluaciones totales. Pero al distribuir el procesamiento en varios procesadores, el costo expresado en tiempo se ve claramente disminuido.

Si bien el análisis sobre la complejidad de cada una de las instancias, determinado por el número de variables a considerar en el problema, permite vincularla con el costo -expresado por la cantidad de evaluaciones-, dicho análisis se ve distorsionado por la complejidad particular de cada instancia. Para sortear dicho obstáculo se están evaluando caminos alternativos que permitan evitar tal distorsión, como así también, evaluar los procedimientos sobre variantes del problema SAT, tal como el COUNTSAT donde las instancias son generadas aleatoriamente.

Otro trabajo futuro consiste en realizar un estudio de eficacia/eficiencia sobre las características de las distintas variantes de los métodos heurísticos para elaborar las conclusiones finales.

Referencias

[AAB+01] E. Alba, F. Almeida, M. Blesa, C. Cotta, M. Díaz, I. Dorta, J. Gabarró, J. González, C. León, L. Moreno, J. Petit, J. Roda, A. Rojas, and F. Xhafa. Mallba: Towards a combinatorial optimization library for geographically distributed systems. In *Proceedings of the XII Jornadas de Paralelismo*, pages 105–110. Editorial U.P.V., 2001.

[BEV98] Back, Eiben, and Vink. A superior evolutionary algorithm for 3-SAT. In *EP: International Conference on Evolutionary Programming, in cooperation with IEEE Neural Networks Council*. LNCS, 1998.

[Coo71] S.A. Cook. The complexity of theorem-proving procedures. *In Proc. of the 3rd. Anual ACM Symposium on the Theory of Computing*, pages 151–158, 1971.

[DS89] Kenneth A. De Jong and William M. Spears. Using genetic algorithm to solve NP-complete problems. In James D. Schaffer, editor, *Proc. of the Third Int. Conf. on Genetic Algorithms*, pages 124–132, San Mateo, CA, 1989. Morgan Kaufmann.

[EvdH97] A.E. Eiben and J.K. van der Hauw. Solving 3-sat with adaptive genetic algorithms. In *Proceedings of the 4th IEEE Conference on Evolutionary Computation*, pages 81–86. IEEE Press, 1997.

[FF93] Charles Fleurent and Jacques Ferland. Genetic hybrids for the quadratic assignment problem, 1993.

[FPS98] G. Folino, C. Pizzuti, and G. Spezzano. Combining cellular genetic algorithms and local search for solving satisfiability problems, 1998.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: a Guide to the Theory of NP-completeness*. Freeman, San Francisco, California, 1979.

[GMR02] Jens Gottlieb, Elena Marchiori, and Claudio Rossi. Evolutionary algorithms for the satisfiability problems. *Evolutionary Computation*, 10(2):35–50, Spring 2002.

[GPFW96] J. Gu, P. Purdom, J. Franco, and B. Wah. Algorithms for the satisfiability (sat) problem: a survey, 1996.

[JK] Michiel B. De Jong and Walter A. Kosters. Solving 3-sat using adaptive sampling.

[KS92] Henry A. Kautz and Bart Selman. Planning as satisfiability. In *Proceedings of the Tenth European Conference on Artificial Intelligence (ECAI'92)*, pages 359–363, 1992.

[MR99] Elena Marchiori and Claudio Rossi. A flipping genetic algorithm for hard 3-SAT problems. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 393–400, Orlando, Florida, USA, 13-17 1999. Morgan Kaufmann.

[RM89] R. Reiter and A. Mackworth. A logical framework for depiction and image interpretation. *Artifitial Intelligence*, 41(3):123–155, 1989.

[RW98] Soraya Rana and Darrell Whitley. Genetic algorithm behavior in the MAXSAT domain. *Lecture Notes in Computer Science*, 1498:785–790, 1998.

[S.68] Tseitin G. S. On the complexity of derivation in propositional calculus. *Studies in Constructive Mathematics and Mathematical Logic*, 2:115–125, 1968. New York-London.

[SLM92] Bart Selman, Hector J. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In Paul Rosenbloom and Peter Szolovits, editors, *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 440–446, Menlo Park, California, 1992. AAAI Press.