Escuela Técnica Superior de Ingenieros Informáticos

Universidad Politécnica de Madrid

# Sistema evolutivo híbrido para la construcción de Redes de Neuronas

Trabajo Fin de Máster

Máster Universitario en Inteligencia Artificial

AUTOR: Carlos Vázquez Losada
TUTOR: Daniel Manrique Gamo

20 de julio de 2019

# Agradecimientos

Deseo expresar mi agradecimiento, en primer lugar, a Daniel, un investigador incansable y un profesor excepcional. Gracias por dedicarme tu tiempo y adaptarte a mis horarios, y gracias por despertar en mi el interés en la Computación Evolutiva con tus clases y con tu implicación.

A mis padres, Isabel Losada y Fco. Javier Vázquez, por su amor, comprensión y por su apoyo incondicional en todo aquello que me he propuesto.

A Cristina, espectadora de mis éxios y mi acompañante. Siempre sabes que decir y cómo complementarme. Te quiero.

# Resumen

Este Trabajo de Fin de Máster consiste en estudiar la idoneidad de la Computación Evolutiva para la generación de Redes de Neuronas Artificiales. En concreto, en un estudio de la Programación Genética y si el entrenamiento parcial de las redes candidatas es o no suficiente para resolver estos problemas frente al entrenamiento total de las mismas.

La Programación Genética es una técnica evolutiva que se utiliza en problemas de optimización cuyas soluciones son programas informáticos. La Programación Genética Guiada por Gramáticas extiende las posibilidades de la Programación Genética tradicional con la introducción de las gramáticas, que permiten crear individuos sintácticamente válidos.

Estas gramáticas permiten la confección de arquitecturas de Redes de Neuronas Artificiales que son válidas, dado cualquier número de neuronas en la capa de entrada y en la capa de salida. El resultado de una ejecución nos devolverá la arquitectura de red que mejor se amolde al problema dado, teniendo en cuenta las propias peculiaridades de la gramática utilizada. El objetivo es determinar si el entrenamiento parcial de estas redes da resultados lo más similares posibles al del entrenamiento total, ya que es bastante ineficiente y tardío.

# Summary

This Master Thesis Dissertation consists in a research about how appropiate is to use Evolutionary Computation for creating Artificial Neural Networks. A technique of Evolutionary Computation is used for this purpose; Grammar-Guided Genetic Programming. It will be used for looking the best Artificial Neural Network for a concrete dataset, by checking both partially and fully trained networks.

Genetic Programming is an evolutionary technique (it is inspired by biology as Evolutionary Computation does) and it is used for solving optimization problems whose solution is a computer program. The Grammar-Guided Genetic Programming extends Genetic Programming by adding grammars, that allow to create valid individuals.

These grammars allow creating Neural Network architectures that are all valid, given any number of neurons in both input and output layers. The result of an execution would be the architecture that best fixes (in terms of accuracy) the given data.

# 1. Introducción

La optimización matemática estudia un tipo concreto de problemas donde se desea elegir el mejor entre un conjunto de elementos. El problema clásico (Boyd and Vandenberghe, 2004) consiste en maximizar o minimizar una función objetivo que representa o mide la calidad de las decisiones. Además, esta función está sujeta a un conjunto de restricciones que acotan el espacio de soluciones.

Para resolver estos problemas, existen algoritmos de optimización, métodos iterativos y heurísticas. Uno de los primeros algoritmos de optimización es el algoritmo de Simplex (Dantzig, 1990). Los métodos iterativos buscan la convergencia hacia una solución determinada. Un ejemplo es el Método de Newton (Nocedal et al., 1999). Las heurísticas (Polya, 1945), en cambio, aproximan la solución en el caso de que previamente no haya habido convergencia.

La Computación Evolutiva es una familia de heurísticas inspiradas en la propia evolución biológica de los seres vivos para la resolución de problemas de optimización. En concreto, la Programación Genética (Koza, 1992) surge por la necesidad de extender la optimización para involucrar programas informáticos.

Las Redes de Neuronas Artificiales son técnicas de Aprendizaje Automático (Samuel, 1959) inspiradas en las redes de neuronas propias de los seres vivos. El objetivo de este modelo es la resolución de problemas tal y como lo hace el cerebro humano. Son especialmente utilizadas en los ámbitos de Visión Artificial (Roberts, 1965) y en el Reconocimiento de Voz (Waibel et al., 1989), aunque también son ampliamente usadas para problemas de Aprendizaje Supervisado básicos como los conformados por información proveniente de ficheros, y es en este tipo de clasificación en la que se centra este trabajo.

La Computación Evolutiva y las Redes de Neuronas Artificiales, ambas inspiradas en la biología, no son opuestas sino que se complementan. La Computación Evolutiva permite resolver problemas de optimización y búsqueda, mientras que las Redes de Neuronas Artificiales son indicadas para el Aprendizaje Automático. La Programación Genética supuso una alternativa a la búsqueda de la mejor Red de Neuronas Artificiales, ya que, antiguamente, la selección de la arquitectura se basaba en la experiencia previa, y en la preba y el error.

El objetivo principal de este trabajo es estudiar la idoneidad de la Computación Evolutiva para la generación de Redes de Neuronas Artificiales. Para ello, utilizaremos una de las técnicas de esta familia de heurísticas, la Programación Genética Guiada por Gramáticas, para tal fin. En concreto, se establecen varios objetivos:

1. Determinar si el entrenamiento parcial de los individuos (redes) de un progra-

ma genético para buscar la mejor arquitectura de Red de Neuronas Artificiales es comparable con el entrenamiento total y se obtienen resultados idénticos, para ahorrar así grandes tiempos de ejecución.

2. Implementar un Programa Genético que lidie con Redes de Neuronas y que permita el estudio propuesto.

3. Probar la gramática con ejemplos diversos para realizar un estudio más exhaustivo sobre ella.

Este documento se estructura en siete secciones después de esta introducción. Las dos primeras se centran en la *Computación Evolutiva* y en las *Redes de Neuronas Artificiales*, respectivamente. La tercera sección trata sobre la *construcción de Redes de Neuronas*, seguida del *planteamiento del problema*, en la que se describe con profundidad el propósito de este trabajo y la necesidad real de él. A esta sección le sigue la *solución propuesta*, que contiene la metodología y el procedimiento seguido, exponiendo los *resultados* obtenidos en la séptima sección. Finalmente, el trabajo finaliza con las *conclusiones y líneas futuras* propuestas.

# 2. Computación Evolutiva

La Computación Evolutiva comprende un conjunto de técnicas para la resolución de problemas de búsqueda y optimización inspiradas en la evolución biológica de los seres vivos. En este capítulo se ofrecen, en primer lugar, un resumen de la *historia* de la Computación Evolutiva. Seguidamente, una *definición* sobre esta rama de estudio y que finaliza hablando sobre la *Programación Genética* y la *Programación Genética Guiada por Gramáticas*.

## 2.1. Historia

La Computación Evolutiva surge con los trabajos de Box (Box, 1957), Friedberg (Friedberg, 1958, 1959) y Bremermann (Bremermann, 1962). Sin embargo, no se consiguieron grandes avances debido a la pobre metodología todavía sin desarrollar y a las limitaciones computacionales de la época.

Algunos años después surgen los primeros desarrollos metodológicos en una década de destacable logro científico. El trabajo de Fogel (Fogel et al., 1966) sienta las bases de la Programación Evolutiva (*evolutionary programming*) y el de Holland (Holland, 1967) las de los Algoritmos Genéticos (*genetic algorithms*). También, en esa misma época, las Estrategias Evolutivas (*evolution strategies*) fueron introducidas por Rechenberg (Rechenberg, 1965) y Schwefel (Schwefel, 1965).

Una década después de los primeros descubrimientos, alrededor de los años 80, los avances computacionales permitieron aplicar las técnicas evolutivas descubiertas para resolver problemas de optimización del mundo real. En esa misma década, los estudios de Cramer (Cramer, 1985) y Koza (Koza, 1988) desembocaron en la aparición de una nueva técnica perteneciente a la Computación Evolutiva: la Programación Genética (*genetic programming*). Pocos años después, la Programación Genética contaba con más de 10.000 artículos publicados (Hu et al., 2014). Pocos años después ya habría cerca de 10.000 publicaciones sobre esta peculiar heurística. También se sucedieron una gran cantidad de conferencias internacionales y talleres centrados en aspectos teóricos de los Algoritmos Genéticos (Grefenstette, 1985, 1987; Schaffer, 1989), entre muchos otros. Estos años destacaron también por la aparición de otras técnicas de Computación Evolutiva como la Vida Artificial (*artificial life*) abreviada muy comúnmente como *A-Life* (Langton, 1986), los Sistemas Inmunitarios Artificiales (*artificial immune systems*) (Farmer et al., 1986), la Inteligencia de Enjambre (*swarm intelligence*) (Beni and Wang, 1989) y los Algoritmos Meméticos (Moscato, 1989).

Hacia 1990, era innegable admitir que parte de la comunidad científica del mun-

do ponía sus ojos en la Computación Evolutiva. A las conferencias anteriormente expuestas le siguieron las cuatro conferencias de IEEE sobre Computación Evolutiva, que asentaron esta rama de estudio como una de las articulaciones de la Inteligencia Artificial y herramienta indispensable para la resolución de problemas de optimización y búsqueda en el ámbito académico y empresarial. La conferencia de Programación Genética (Koza et al., 1996) tuvo gran éxito y aceptación, a la cual le siguieron otras conferencias sobre el mismo ámbito, como la EuroGP (Banzhaf, 1998). Destacaron también conferencias sobre desarrollos metodológicos sobre los Algoritmos Genéticos (Rawlins, 1991; Whitley, 1993; Vose, 1995), que se consolidó como la técnica de Computación Evolutiva más destacable. Esta década se cierra con la llegada de nuevas técnicas de Computación Evolutiva: los Algoritmos Culturales (*cultural algorithms*) (Reynolds, 1994), la Evolución Diferencial (*differential evolution*) (Storn, 1996; Storn and Price, 1997) y la Evolución Gramatical (*grammatical evolution*) (Ryan et al., 1998), además de la aparición de una variante de la Programación Genética, la Programación Genética Guiada por Gramática (Whigham

A estas casi cuatro décadas de gran logro científico le siguieron dos más de un impecable desarrollo metodológico. En estos años destacaron los estudios sobre los Algoritmos Genéticos (Deb et al., 2002; Hassan et al., 2005; Pezzella et al., 2008), entre muchos otros, además de la aparición de nuevas heurísticas como la Búsqueda Armónica (*harmony search*) (Geem et al., 2001) o los Algoritmos Genéticos Basados en Humanos (*human based genetic algorithms*) (Kosorukoff, 2001). Otras técnicas evolutivas también tuvieron su amplio desarrollo metodológico y crecimiento teórico (Couchet and Manrique, 2006; De Araujo and Tavares, 2014; Beni, 2004).

## 2.2. Funcionamiento general

La Computación Evolutiva se fundamenta en la evolución biológica y cuenta con un conjunto de heurísticas para la resolución de problemas de búsqueda y optimización. Cada una de ellas sirve para un tipo concreto de problemas de este ámbito, no asegurando su eficacia si se utiliza en otro dominio.

En la evolución biológica (Darwin, 1959), los individuos son los protagonistas. Estos individuos viven en poblaciones, de tal forma que se produce una evolución conjunta de la población mediante la aplicación de operadores evolutivos tales como la selección, el reemplazo, el cruce y la mutación.

Para la Computación Evolutiva, los individuos son las soluciones candidatas al problema de búsqueda u optimización dado. La población se compone del conjunto de individuos base y de la que parte la ejecución del algoritmo. Sobre esta población se aplican los operadores evolutivos (referidos en este ámbito como operadores genéticos) anteriormente mencionados, con el objetivo de que se produzca una mejora y un acercamiento gradual hacia la solución óptima en cada generación. Teóricamente, se espera que tras un número finito de generaciones, la población haya convergido

hacia la solución óptima al problema dado. En la práctica, puede que sea conveniente limitar este número de generaciones por las capacidades computacionales limitadas que hoy en día experimentamos, y nuestra población llegue hasta un cierto grado de optimalidad.

Este proceso se lleva a cabo conformando el Ciclo Evolutivo (Koza, 1996) que se repite hasta que finalmente se cumple la condición de parada.

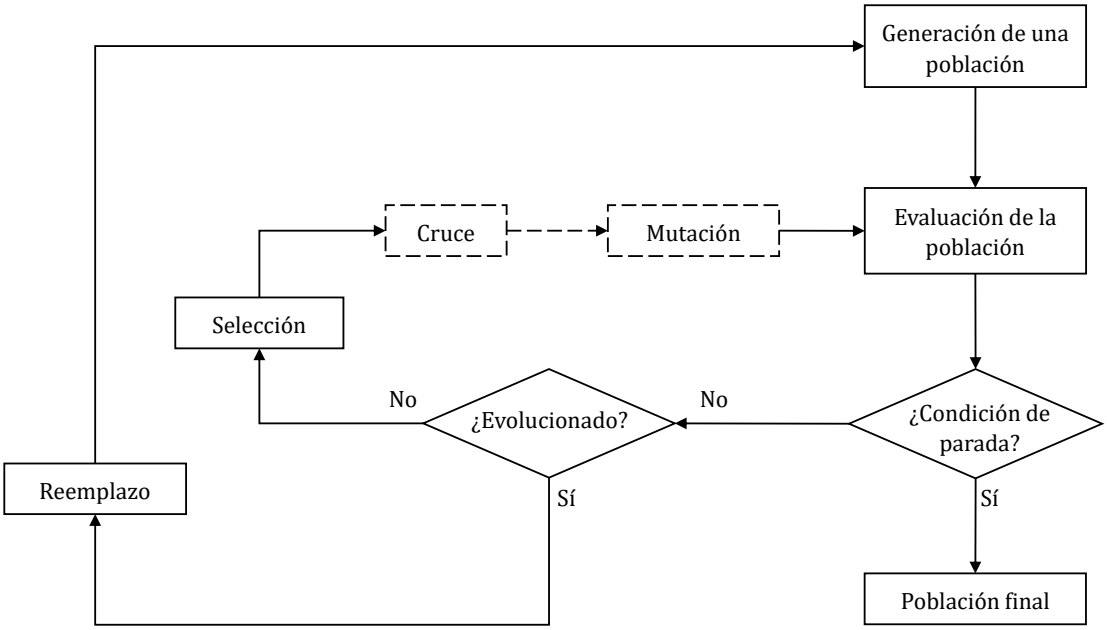


Fig. 1: Ciclo Evolutivo que rige el desarrollo de los algoritmos en la Computación Evolutiva.

En Fig. 1 se puede apreciar este Ciclo Evolutivo. Todo comienza con una población inicial, que se corresponderá con un conjunto de individuos que serán seleccionados

## 2.3. Programación Genética

## 2.3. Programación Genética Guiada por Gramáticas

# 3. Redes de Neuronas Artificiales

# 4. Construcción de Redes de Neuronas

# 5. Planteamiento del problema

# 6. Solución propuesta

# 7. Resultados

# 8. Conclusiones y líneas futuras

# Bibliografía


Banzhaf, W., Poli, R., Schoenauer, M. and Fogarty, T.C. (1998). *Genetic Programming.*

Beni, G. and Wang, J. (1989). Swarm intelligence in cellular robotic systems. *Proceedings5 of the NATO Advance Workshop on Robots and Biological Systems.* 102:703-712.

Beni, G. (2004). From Swarm Intelligence to Swarm Robotics. *International Workshop on Swarm Robotics.* 3342:1-9.

Boyd, S. and Vandenberghe, L. (2004). Convex Optimization. *Cambridge University Press 2004*: 129.

Box, G.E.P. (1957). Evolutionary operation: A method for increasing industrial productivity. *Journal of the Royal Statistical Society. Series C.* 6(2):81-101.

Box, G.E.P. and Draper, N.P. (1969). *Evolutionary Operation. A method for Increasing Industrial Productivity.*

Bremermann, H.J. (1962). Optimization through evolution and recombination. *Self-Organizing Systems 1962*: 93-106.

Couchet, J. and Manrique, D. (2006). Crossover and mutation operations for grammar-guided genetic programming. *Soft Computing.* 11(10):943-955.

Cramer, N.L. (1985). A representation for the Adaptive Generation of Simple Sequential Programs. *Proceedings of the First International Conference on Genetic Algorithms and the Applications*: 183-187.

Dantzig, G.B. (1990). Origins of the simplex method. *A history of scientific computing*, pages 141-151.

Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection, or Preservation of Favoured Races in the Struggle for Life.*

De Araujo, A.F. and Tavares, J.M.R.S. (2014). An Artificial Life Model for Image Enhancement. *15th International Conference on Experimental Mechanics.* 41(13):5892-5906.

Deb, K., Agarwal, S., Pratap, A. and Meyarivan, T. (2002). A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation.* 6(2):128-197.

Farmer, J.D., Packard, N.H. and Perelson, A.S. (1986). The Immune System, Adaptation, and Machine Learning. *Physica D: Nonlinear Phenomena*. 22(1): 187-204.

Fogel, L.J., Owens, A.J. and Walsh, M.J. (1966). *Artificial Intelligence through Simulated Evolution*.

Fraser, A.S. (1957). Simulation of Genetic Systems by Automatic Digital Computers. *Australian journal of biological sciences*. 10:484-499.

Friedberg, R.M. (1958). A learning machine: part I. *IBM Journal of Research and Development*. 2(1):2-13.

Friedberg, R.M., Dunham, B. and North, J.H. (1959). A learning machine: part II. *IBM Journal of Research and Development*. 3(3):282-287.

Geem, Z.W., Kim, J.H. and Loganathan, G.V. (2001) A New Heuristic Optimization Algorithm: Harmony Search. *Simulation*, 76(2):60-68.

Grefenstette, J.J. (1985). *Proceedings of the First International Conference on Genetic Algorithms and the Applications* (Hillsdale, NJ: Lawrence Erlbaum).

Grefenstette, J.J. (1987). *Proceedings of the Second International Conference on Genetic Algorithms and the Applications* (Hillsdale, NJ: Lawrence Erlbaum).

Hassan, R., Cohanim, B., de Weck, O. and Venter, G. (2005). A comparison of particle swarm optimization and the genetic algorithm. *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*.

Holland, J.H. (1962). Nonlinear environments permitting efficient adaptation. *Computer and Information Sciences II*: 147-164.

Hu, T., Banzhaf, W. and Moore, J.H. (2014). The effects of recombination of phenotypic exploration and robustness in evolution. *Artificial Life*. 20(4):457-470.

Koza, J.R. (1988). Non-Linear Genetic Algorithms for Solving Problems. *University States Patent 4935877*.

Koza, J.R. (1992). *Genetic Programming 1992* (Cambridge, MA: MIT Press).

Koza, J.R.(1996). *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (Cambridge, MA: MIT Press).

Koza, J.R., Goldberg, D.E., Fogel, D.B. and Riolo, R.L. (1996). *Genetic Programming 1996* (Cambridge, MA: MIT Press).

Kosorukoff, A. (2001). Human based genetic algorithm. *2001 IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace*: 3464-3469.

Langton, C.G. (1986). Studying artificial life with cellular automata. *Physica D: Nonlinear Phenomena*. 22(1):120-149.

Moscato, P. (1989). On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts - Towards Memetic Algorithms. *Caltech Concurrent Computation Program*: 158-179.

Nocedal, J. and Wright, S.J. (1999). Numerical optimization. *Springer-Verlag*.

Pezzella, F., Morganti, G. and Ciaschetti, G. (2008). A genetic algorithm for the Flexible Job-shop Scheduling Problem. *Computers & Operations Research*. 35(10):3202-3212.

Polya, G. (1945). How to Solve It. *Princeton University Press 1945*.

Rawlins, G.J.E. (1991). *Foundations of Genetic Algorithms* (San Mateo, CA: Morgan Kaufmann).

Rechenberg, I. (1965). Cybernetic Solution Path of an Experimental Problem. *Royal Aircraft Establishment Library Translation 1122*.

Reynolds, R.G. (1994). An Introduction to Cultural Algorithms. *Proceedings of the 3rd Annual Conference on Evolutionary Programming*: 131-139.

Roberts, L. (1965). Machine perception of 3D solids. *Optical and Electro-Optical Information Processing*. 9:159-197.

Ryan, C., Collins, J.J. and O'Neil, M. (1998). Grammatical Evolution: Evolving Programs for an Arbitrary Language. *Proceedings of the First European Workshop on Genetic Programming*: 83-96.

Samuel, A.L. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*. 3(3):210-229.

Schaffer, J.D. (1989). *Proceedings of the Third International Conference on Genetic Algoritms and the Applications* (San Mateo, CA: Morgan Kaufmann).

Schwefel, H-P. (1965). Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik.

Storn, R. (1996). On the usage of differential evolution for function optimization. *Biennial Conference of the North American Fuzzy Information Processing Society*: 519-523.

Storn, R. and Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*. 11(4):341-359.

Vose, M.D. and Whitley, L.D. (1995). *Foundations of Genetic Algorithms 3* (San Francisco, CA: Morgan Kaufmann).

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K. and Lang, K.J. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 37(3):328-339.

Whigham, P.A. (1995). Grammatically-based genetic programming. *Proceedings of the workshop on genetic programming: from theory to real-world applications*: 33-41.

Whitley, L.D. (1993). *Foundations of Genetic Algorithms 2* (San Mateo, CA: Morgan Kaufmann).