Introducción a los algoritmos genéticos y sus aplicaciones

Autora: Piedad Tolmos Rodríguez-Piñero
Dirección: Paseo de los Artilleros s/n Madrid 28032
Teléfono: 91-3019901
E-Mail: tolmos@poseidon.fcjs.urjc.es

Resumen: Los algoritmos genéticos son un logro más de la Inteligencia Artificial en su intento de replicar comportamientos biológicos, con los avances científicos que ello implica, mediante la computación. Se trata de algoritmos de búsqueda basados en la mecánica de la selección natural y de la genética. Utilizan la información histórica para encontrar nuevos puntos de búsqueda de una solución óptima del problema planteado, con esperanzas de mejorar los resultados. En el presente artículo se realizará una introducción a los Algoritmos Genéticos: qué son, de dónde proceden, y en qué difieren de otros métodos de búsqueda, comentándose, asimismo, sus aplicaciones principales.

Palabras Clave: Optimización, problemas de búsqueda, gen cromosoma, cadena de bits, operador, población, capacidad,algoritmo genético.

1. Introducción

La capacidad del ser humano para predecir el comportamiento de su entorno, se ha ido incrementando con el paso del tiempo. De igual modo, ha comprendido que, si bien era capaz de controlar muchos aspectos de su vida, y su interacción con lo que le rodeaba, no lo era para otros tantos.

La inteligencia artificial es responsable de muchos de esos logros. Los pioneros de esta ciencia estaban tan interesados en la electrónica, como en la biología, y por eso sus aplicaciones iban desde calcular trayectorias de misiles, a tratar de modelizar el cerebro, de imitar el proceso de aprendizaje humano, y de simular la evolución biológica.

Los años ochenta (en el siglo pasado) marcan el florecimiento del interés de la comunidad científica por estos temas computacionales inspirados en la biología, que han visto como su desarrollo les llevaba a cotas inimaginables, primero en el campo de las Redes Neuronales, luego en el del Aprendizaje, y por último en lo que ahora se conoce como "computación evolutiva", de la que los algoritmos genéticos constituyen su máximo exponente.

2. Breve recorrido histórico por la computación evolutiva

El origen de lo que se conoce como *computación evolutiva* hay que buscarlo en su razón de ser: los conocimientos sobre evolución se pueden aplicar en la resolución de problemas de optimización. Fue en las décadas de 1950 y 1960 cuando varios científicos, de modo independiente, comenzaron a estudiar los sistemas evolutivos, guiados por la intuición de que se podrían emplear como herramienta en problemas de optimización en ingeniería. La idea era "evolucionar" una población de candidatos a ser solución de un

problema conocido, utilizando operadores inspirados en la selección natural y la variación genética natural.

Fue Rechenberg quien, en la década de 1960 (1965, 1973) introdujo las "estrategias evolutivas", método que empleó para optimizar parámetros reales para ciertos dispositivos. La misma idea fue desarrollada posteriormente por Schwefel (1975, 1977). El campo de las estrategias evolutivas ha permanecido como un área de investigación activa, cuyo desarrollo se produce, en su mayor parte, de modo independiente al de los algoritmos genéticos (aunque recientemente se ha visto como las dos comunidades han comenzado ha colaborar). Fogel, Owens y Walsh (1966), fueron los creadores de la "programación evolutiva", una técnica en la cual las candidatas a soluciones a tareas determinadas, eran representadas por máquinas de estados finitos, cuyos diagramas de estados de transición se evolucionaban mediante mutación aleatoria, seleccionándose el que mejor aproximara. Una formulación más amplia de la programación evolutiva, es un campo de investigación que también continúa en activo (ver, por ejemplo, a Fogel y Altman 1993). Estas tres áreas, estrategias evolutivas, algoritmos genéticos, y programación evolutiva, son las que forman la columna vertebral de la Computación Evolutiva, y de ellas parten los caminos hacia todos los campos de investigación inspirados en nuestros conocimientos sobre Evolución.

Pero muchos otros investigadores desarrollaron su trabajo en los algoritmos para la optimización y el aprendizaje inspirados en la evolución. Cabe resaltar nombres como los de Box (1957), Friedman (1959), Bledsoe (1961), Bremermann (1962), y Reed, Toombs y Baricelli (1967). Sin embargo, su trabajo no ha tenido, ni con mucho, la atención que han recibido las estrategias evolutivas, programación evolutiva, y los algoritmos genéticos. Hay que recordar además a los biólogos evolucionistas que han utilizado el ordenador para simular la evolución para realizar experimentos controlados (Baricelli 1957, 1962; Fraser 1957 a,b; Martin y Coreman 1960). Pero habría que esperar hasta que la computación electrónica se desarrollara, para poder apreciar la consolidación definitiva de la computación evolutiva.

Centrémonos en los Algoritmos Genéticos. La primera mención del término, y la primera publicación sobre una aplicación del mismo, se deben a Bagley (1967), que diseñó algoritmos genéticos para buscar conjuntos de parámetros en funciones de evaluación de juegos, y los comparó con los algoritmos de correlación, procedimientos de aprendizaje modelizados después de los algoritmos de pesos variantes de ese periodo. Pero es otro científico el considerado creador de los Algoritmos Genéticos: John Holland, que los desarrolló, junto a sus alumnos y colegas, durante las décadas de 1960 y 1970. En contraste con las estrategias evolutivas y la programación evolutiva, el propósito original de Holland no era diseñar algoritmos para resolver problemas concretos, sino estudiar, de un modo formal, el fenómeno de la adaptación tal y como ocurre en la naturaleza, y desarrollar vías de extrapolar esos mecanismos de adaptación natural a los sistemas computacionales. El libro que Holland escribió en 1975 *Adaptación en Sistemas Naturales y Artificiales* presentaba el algoritmo genético como una abstracción de la evolución biológica, y proporcionaba el entramado teórico para la adaptación bajo el algoritmo genético. El Algoritmo Genético de Holland era un método para desplazarse, de una población de cromosomas (bits) a una nueva población, utilizando un sistema similar a la "selección natural" junto con los operadores de *cruces*, *mutaciones* e *inversión* inspirados en la genética. En este primitivo algoritmo, cada cromosoma consta de genes (bits), y cada uno

de ellos es una muestra de un alelo particular (0 o 1). El operador de selección escoge, entre los cromosomas de la población, aquellos con capacidad de reproducción, y entre éstos, los que sean más "compatibles", producirán más descendencia que el resto. El de cruce extrae partes de dos cromosomas, imitando la combinación[1] biológica de dos cromosomas aislados (gametos). La mutación se encarga de cambiar, de modo aleatorio, los valores del alelo en algunas localizaciones del cromosoma; y, por último, la inversión, invierte el orden de una sección contigua del cromosoma , recolocando por tanto el orden en el que se almacenan los genes.

La mayor innovación de Holland fue la de introducir un algoritmo basado en poblaciones con cruces, mutaciones e inversiones[2]. Es más, Holland fue el primero en intentar colocar la computación evolutiva sobre una base teórica firme (Holland, 1975). Hasta hace poco, esta base teórica, fundamentada en la noción de "esquemas", fue la estructura sobre la que se edificaron la mayoría de los trabajos teóricos sobre algoritmos genéticos en las décadas siguientes.

En estos últimos años se ha generado una amplia interacción entre los investigadores de varios métodos de computación evolutiva, rompiéndose las fronteras entre algoritmos genéticos, estrategias evolutivas y programación evolutiva. Como consecuencia, en la actualidad, el término "algoritmo genético" se utiliza para designar un concepto mucho más amplio del que concibió Holland.

3. ¿Qué son los algoritmos genéticos?

3.1 Concepto

Los objetivos que perseguían John Holland y sus colegas de la Universidad de Michigan cuando concibieron los algoritmos genéticos, eran dos: (1) abstraer y explicar rigurosamente el proceso adaptativo de los sistemas naturales, y (2) diseñar sistemas artificiales que retuvieran los mecanismos más importantes de los sistemas naturales. En este sentido, podemos decir que los algoritmos genéticos son

Algoritmos de búsqueda basados en los mecanismos de selección natural y genética natural. Combinan la supervivencia de los más compatibles entre las estructuras de cadenas, con una estructura de información ya aleatorizada, intercambiada para construir un algoritmo de búsqueda con algunas de las capacidades de innovación de la búsqueda humana[3].

Básicamente, el Algoritmo Genético funciona como sigue: en cada generación, se crea un conjunto nuevo de "criaturas artificiales" (cadenas) utilizando bits y partes más adecuadas del progenitor. Esto involucra un proceso aleatorio que no es, en absoluto, simple. La novedad que introducen los Algoritmos Genéticos es que explotan eficientemente la información histórica para especular sobre nuevos puntos de búsqueda, esperando un funcionamiento mejorado.

---

[1] Aquí, como en la mayoría de la literatura sobre algoritmos genéticos, *combinación* equivale a *cruce*.

[2] Las estrategias evolutivas de Rechenberg comenzaban con una población de dos individuos, un padre y un descendiente, siendo éste una versión mutada del padre; hasta más tarde no se incorporaron poblaciones de más individuos, ni cruces entre ellos. En cuanto a los programas evolutivos de Fogel, Owens y Walsh, sólo utilizaban mutaciones para producir variaciones.

[3] Goldberg, D. (1989) Genetics Algorithms in Search, Optimization and Machine Learning. Addison Wesley.

El tema central en las investigaciones sobre algoritmos genéticos,, ha sido la *robustez*, el equilibrio necesario entre la eficiencia y la eficacia suficiente para la supervivencia en entornos diferentes. Las implicaciones que tiene la robustez en los sistemas artificiales son variadas. Si se puede conseguir que un sistema artificial sea más robusto, se podrán reducir, e incluso eliminar, los costes por rediseños. Y si se es capaz de lograr niveles altos de adaptación, los sistemas podrán desarrollar sus funciones mejor y durante más tiempo. Sin embargo, ante la robustez, eficiencia y flexibilidad de los sistemas biológicos, sólo podemos sentarnos a contemplar, y maravillarnos; mentiríamos si dijéramos que somos capaces de igualarlos.

Pero, ¿por qué basarse en nuestros conocimientos sobre la evolución biológica? La respuesta la encontramos si observamos una constante que se repite en muchos problemas: la búsqueda de soluciones entre una cantidad ingente de candidatos. Tómese, por ejemplo, el cálculo de un conjunto de reglas (ecuaciones) capaz de regir las subidas y bajadas de un mercado financiero. El modo de llegar a la mejor solución en estas situaciones, pasa por ser capaz de obtener rendimiento de un uso eficaz del *paralelismo*, que permita explorar diferentes posibilidades de modo simultáneo. Para ello, se precisa, tanto paralelismo computacional (contar con varios procesadores computando al mismo tiempo), como una *estrategia* adecuada de búsqueda.

Por otro lado, muchos problemas computacionales, precisan de un programa *adaptativo*, capaz de comportarse bien ante cambios en el entorno. Además, la mayoría de estos problemas tienen soluciones complejas, muy difíciles de programar a mano. Entre las técnicas que han surgido al tratar de resolver estas cuestiones, encontramos el "conexionismo" (el estudio de programas computacionales inspirados en sistemas neuronales, de los que ya se ha hablado en otras Jornadas) , y la "computación evolutiva". Si en la primera las reglas pasaban por *umbrales* neuronales, propagación de la *activación*, y refuerzo o no de las *conexiones*., en la segunda son la *selección natural*, con variaciones debidas a *cruces* y/o *mutaciones*, y su objetivo es el diseño de soluciones de alta calidad para problemas de elevado grado de complejidad, y la habilidad de adaptar esas soluciones de cara a cambios en el entorno.

La evolución, tal y como la conocemos, es básicamente un método de búsqueda entre un número enorme de posibles "soluciones". En biología las posibilidades están formadas por un conjunto de secuencias genéticas posibles, y las soluciones deseadas, por organismos capaces de sobrevivir y reproducirse en sus entornos. La evolución puede verse, asimismo, como un modo de "diseñar" soluciones a problemas complejos, con la capacidad de *innovar*. Estos son los motivos de que los mecanismos evolutivos sean una fuente de inspiración para los algoritmos de búsqueda. Por supuesto, el buen funcionamiento de un organismo biológico depende de muchos criterios, que además varían a medida que el organismo evoluciona, de modo que la evolución está "buscando" continuamente entre un conjunto cambiante de posibilidades. Por ello, podemos considerarla como un método de búsqueda masivamente paralelo, ya que evalúa y cambia millones de especies en paralelo. Para terminar, las reglas de la evolución, aunque de alto nivel, son simples: las especies evolucionan mediante variaciones aleatorias (vía mutaciones, recombinaciones, etc.) seguidas por la selección natural, donde el mejor tiende a sobrevivir y reproducirse, propagando así su material genético a posteriores generaciones.

3.2 Elementos de un algoritmo genético y su traducción biológica.

Todos los organismos que conocemos están compuestos por una o más células, cada una de las cuales contiene a su vez uno o más *cromosomas* (esto es, cadenas de ADN), que tienen la función de ser una especie de "anteproyecto" del organismo del que forman parte. Un cromosoma se puede dividir, conceptualmente, en *genes*, bloques funcionales de ADN que codifican una determinada proteína. Solemos pensar en los genes, aunque en una visión muy superficial, como los responsables de determinar los rasgos del individuo, tales como el color de los ojos, o del cabello. Las diferentes posibilidades de escoger un rasgo (ojos azules, marrones o verdes, por ejemplo) reciben el nombre de *alelos*. Cada gene está localizado en una determinada posición (*lugar*) dentro del cromosoma que integra.

Muchos organismos tienen varios cromosomas en cada célula. El *genoma* del organismo es la colección completa del material genético. Lo que se conoce como *genotipo*, es el conjunto de genes contenido en un genoma. El genotipo dará lugar, tras el desarrollo fetal, al *fenotipo* del organismo, esto es, a sus características físicas y mentales, tales como el color de ojos, la estatura, o la inteligencia.

La mayoría de las especies reproductoras sexualmente que habitan nuestro planeta, almacenan sus cromosomas por parejas (son diploides; se llaman *haploides* en caso contrario). En el caso del ser humano, cada célula somática (no germen) de su cuerpo contiene 23 pares de cromosomas. Durante la reproducción sexual se produce una *recombinación* o *cruce*: en cada padre, se intercambian los genes entre cada par de cromosomas, para formar un *gameto* (un cromosoma único), y entonces, los gametos de los dos padres se emparejan para constituir un conjunto completo de cromosomas diploides. En el caso de la reproducción haploide, los genes se intercambian entre dos padres con una sola rama de cromosomas. La descendencia está sujeta a *mutaciones*, donde se produce un cambio en algún nucleótido (bits elementales de ADN) de padre a hijo; esas modificaciones son resultado habitualmente de "errores de copia". La *capacidad* del organismo se define como la probabilidad de que el organismo viva para reproducirse (*viabilidad*) o como una función del número de descendencia que tenga ese organismo (*fertilidad*).

Traslademos estos conceptos a la "vida artificial": en los algoritmos genéticos, el término *cromosoma* se refiere a un candidato a solución del problema, que a menudo se codifica como una cadena de bits. Los *genes* son tanto un bit como bloques cortos de bits adyacentes que codifican un elemento particular del candidato a solución (por ejemplo, en el caso de la optimización de una función multiparamétrica, los bits que codifican un parámetro particular, se considera un gene). Un *alelo* en una cadena de bits será un 0 o un 1 (para alfabetos largos cada *lugar* puede tener más alelos). El *cruce* consiste, normalmente, como en su espejo biológico, en un intercambio de material genético entre dos cromosomas de dos padres haploides. La *mutación* es una permutación en un bit en un lugar aleatorio (o, en el caso de alfabetos largos, remplazar el símbolo de un lugar escogido aleatoriamente con un símbolo nuevo escogido también aleatoriamente).

En la mayoría de las aplicaciones de los algoritmos genéticos nos encontraremos con individuos haploides, concretamente, de cromosomas únicos. El *genotipo* de un individuo en un algoritmo genético que emplea cadenas de bits es, simplemente, la configuración de bits del cromosoma de ese individuo. La noción de *fenotipo* no aparece en el contexto de los algoritmos genéticos, aunque avances recientes en la materia trabajan con

algoritmos que poseen un nivel "genotípico" y uno "fenotípico" (por ejemplo, la cadena de bits que codifica una red neuronal, y la red en sí misma).

Según hemos visto, podemos considerar que los algoritmos genéticos tienen, al menos, estos elementos en común: poblaciones de cromosomas, selección en base a su capacidad, cruces para producir descendencia nueva, y mutación aleatoria de la nueva descendencia. La inversión - el cuarto elemento de los algoritmos genéticos tal y como los concibió Holland - se usa raramente en las implementaciones actuales, y sus ventajas, si las tiene, no están del todo establecidas.

Se puede pensar en cada cromosoma de un algoritmo genético como en un punto en el espacio de búsqueda de candidatos a soluciones. El algoritmo genético procesa poblaciones de cromosomas, remplazando sucesivamente cada población por otra. El algoritmo suele requerir una *función de capacidad o potencial* que asigna una puntuación (la capacidad) a cada cromosoma de la población actual. La capacidad o el potencial de un cromosoma depende de cómo resuelva ese cromosoma el problema a tratar[4]. Un concepto relacionado es el del "paisaje del potencial". Definido originalmente por Sewell Wright (1931) en el contexto de la genética de poblaciones, el paisaje de un potencial es la representación del espacio de todos los posibles genotipos junto con sus capacidades. De este modo, los operadores de cruce y mutación pueden verse como modos de mover una población en el paisaje definido por su función de capacidad, y un algoritmo genético como un método de búsqueda de ese paisaje para cadenas altamente cualificadas[5].

La forma más simple de algoritmo genético utiliza tres tipos de **operadores**: selección, cruce y mutación.

- **Selección o reproducción:** Este operador escoge cromosomas entre la población para efectuar la reproducción. Cuanto más capaz sea el cromosoma, más veces será seleccionado para reproducirse.
- **Cruce:** Se trata de un operador cuya labor es elegir un lugar, y cambiar las secuencias antes y después de esa posición entre dos cromosomas, para crear nueva descendencia (por ejemplo, las cadenas 10010011 y 11111010 pueden cruzarse después del tercer lugar para producir la descendencia 10011010 y 11110011). Imita la recombinación biológica entre dos organismos haploides.
- **Mutación:** Este operador produce variaciones de modo aleatorio en un cromosoma (por ejemplo, la cadena 00011100 puede mutar su segunda posición para dar lugar a la cadena 01011100). La mutación puede darse en cada posición de un bit en una cadena, con una probabilidad, normalmente muy pequeña (por ejemplo 0.001).

Como se ve, los Algoritmos Genéticos difieren de los métodos tradicionales de búsqueda y optimización, en cuatro cuestiones esenciales:

---

[4] Como ejemplo, si se trata de maximizar la función $f(y) = y + |sen(32y)|, 0 \le y < \pi$ (Riolo, 1992), los candidatos a soluciones son valores de *y,* que se codifican como cadenas de bits que representan números reales. El cálculo de la capacidad traslada una cadena de bits dadas, *x,* en un n º real, *y,* y se evalúa entonces la función en ese valor. La capacidad de una cadena es, pues, el valor de la función en ese punto.

[5] Esta idea de evolución como movimientos de poblaciones a lo largo de paisajes invariables, no es realista biológicamente. Por ejemplo, a un organismo no se le puede asignar un valor potencial independientemente de otro organismo de su entorno; a medida que la población cambia, la capacidad de un genotipo particular cambia también.

1. Trabajan con un código del conjunto de parámetros, no con el conjunto mismo (necesitan que el conjunto de parámetros del problema de optimización esté codificado en cadenas finitas sobre un determinado alfabeto). Por trabajar a nivel de código, y no con las funciones y sus variables de control, como los otros métodos, son más difíciles de "engañar".
2. Buscan una población de puntos, no un único punto. Manteniendo una población de puntos muestrales bien adaptados, se reduce la probabilidad de caer en una cima falsa.
3. Emplean la función objetivo, no necesitan derivadas ni otra información complementaria, tan difícil a veces de conseguir. De este modo ganan en eficiencia y en generalidad.
4. Se valen de reglas de transición estocásticas, no deterministas. Los Algoritmos Genéticos se valen de operadores aleatorios para guiar la búsqueda de los mejores puntos; puede parecer extraño, pero la Naturaleza está llena de precedentes al respecto.

4. Un ejemplo simple de algoritmo genético

Sea *X* el problema a resolver. Dada una representación de candidatas a soluciones en una cadena de bits, un algoritmo genético simple, tal y como se describe en Mitchell M. (1998), trabajaría del siguiente modo:

1. Comenzar con una población *P* generada aleatoriamente de *n* cromosomas de *l* bit.
2. Calcular la capacidad *f(x)* para cada cromosoma *x* de *P*.
3. Repetir los siguientes pasos hasta que se hayan creado *n* descendientes:
   a. Seleccionar un par de cromosomas padre de *P* , siendo la probabilidad de selección una función creciente de la capacidad. La selección se realiza "con remplazamiento", es decir, que el mismo cromosoma puede ser seleccionado en más de una ocasión para ser padre.
   b. Con probabilidad p_c (*probabilidad de cruce*, o tasa de cruce), cruzar el par en un punto elegido aleatoriamente (con probabilidad uniforme) para formar dos descendientes. Si no tiene lugar ningún cruce, formar dos descendientes que sean copias exactas de sus respectivos padres. (Obsérvese que aquí la probabilidad de cruce se define como la probabilidad de que dos padres se crucen sobre un único punto. Hay otras versiones de algoritmos genéticos que son de "cruces en múltiples puntos", en los que la tasa de cruce para una pareja de padres es el n º de puntos en los que tiene lugar un cruce).
   c. Mutar los dos descendientes en cada lugar con probabilidad p_m (*probabilidad de mutación*, o tasa de mutación), y colocar los cromosomas resultantes en la nueva población *P'*.
      Si *n* es impar, se puede rechazar aleatoriamente a un miembro de la nueva población.
4. Remplazar la población actual *P* con la nueva *P'*.
5. Volver al paso 2.

Cada iteración del proceso recibe el nombre de *generación*. Lo usual es iterar el algoritmo de 50 a 500 o más veces. El conjunto completo de generaciones se llama *serie*[6]. Al concluir una serie, a menudo se encuentran entre la población uno o más cromosomas con elevada capacidad. Como la aleatoriedad juega un importante papel en cada serie, dos series con diferentes números aleatorios en el origen darán lugar generalmente a comportamientos diferentes. Los investigadores en algoritmos genéticos acostumbran a reportar estadísticas (tales como el mejor potencial hallado en una serie, y la generación donde se encontró el individuo con la mejor capacidad, etc.) evaluadas sobre muchas series diferentes del algoritmo genético sobre el mismo problema.

El procedimiento arriba descrito es en realidad la base de la mayoría de las aplicaciones de los Algoritmos Genéticos. Desde luego, quedan muchos detalles importantes en los que se debería profundizar, como cuál ha de ser el tamaño de la población, y cuáles las probabilidades de cruce (p_c) y de mutación (p_m). De esos "detalles" dependerá, en gran parte, el éxito o fracaso del Algoritmo Genético que estemos aplicando. Asimismo, podemos encontrar otros Algoritmos Genéticos mucho más complejos (y eficaces) que el que se ha utilizado, como aquellos que trabajan sobre otras representaciones además de las cadenas de bits, o los que emplean otros operadores de cruce y mutación, pero se pretende realizar una primera aproximación del lector al campo de los Algoritmos Genéticos, no desarrollar un completo estudio del mismo.

5. Algunas aplicaciones de los Algoritmos genéticos

Aunque, como se ha comentado, el Algoritmo que se utilizó en el apartado anterior es muy simple, ha servido para que los estudios realizados en torno a él, se hayan aplicado a diversos problemas y modelos en ingeniaría, y en la ciencia en general[7]. Cabe destacar entre ellos:

- **Optimización:** Se trata de un campo especialmente abonado para el uso de los Algoritmos Genéticos, por las características intrínsecas de estos problemas. No en vano fueron la fuente de inspiración para los creadores estos algoritmos. Los Algoritmos Genéticos se han utilizado en numerosas tareas de optimización, incluyendo la optimización numérica, y los problemas de optimización combinatoria.
- **Programación automática:** Los Algoritmos Genéticos se han empleado para desarrollar programas para tareas específicas, y para diseñar otras estructuras computacionales tales como el autómata celular, y las redes de clasificación.
- **Aprendizaje máquina:** Los algoritmos genéticos se han utilizado también en muchas de estas aplicaciones, tales como la predicción del tiempo o la estructura de una proteína. Han servido asimismo para desarrollar determinados aspectos de sistemas particulares de aprendizaje, como pueda ser el de los pesos en una red

[6] En inglés la palabra es *run*, que en computación se asocia a *correr* un programa. En este caso se ha traducido por *serie* para indicar el conjunto de todas las iteraciones del programa.

[7] En Goldberg (1989) se puede encontrar un capítulo dedicado a las aplicaciones, tanto históricas como actuales, de los Algoritmos Genéticos, junto con una tabla en la que se detalla una relación de las investigaciones, el campo al que pertenecen, el año, y los investigadores que las desarrollaron.

neuronal, las reglas para sistemas de clasificación de aprendizaje o sistemas de producción simbólica, y los sensores para robots.

- **Economía:** En este caso, se ha hecho uso de estos Algoritmos para modelizar procesos de innovación, el desarrollo estrategias de puja, y la aparición de mercados económicos.
- **Sistemas inmunes:** A la hora de modelizar varios aspectos de los sistemas inmunes naturales, incluyendo la mutación somática durante la vida de un individuo y el descubrimiento de familias de genes múltiples en tiempo evolutivo, ha resultado útil el empleo de esta técnica.
- **Ecología:** En la modelización de fenómenos ecológicos tales como las carreras de armamento biológico, la coevolución de parásito-huesped, la simbiosis, y el flujo de recursos.
- **Genética de poblaciones:** En el estudio de preguntas del tipo *"¿Bajo qué condiciones será viable evolutivamente un gene para la recombinación?*"
- **Evolución y aprendizaje:** Los Algoritmos Genéticos se han utilizado en el estudio de las relaciones entre el aprendizaje individual y la evolución de la especie.
- **Sistemas sociales:** En el estudio de aspectos evolutivos de los sistemas sociales, tales como la evolución del comportamiento social en colonias de insectos, y la evolución de la cooperación y la comunicación en sistemas multi-agentes.

Aunque esta lista no es, en modo alguno, exhaustiva, sí transmite la idea de la variedad de aplicaciones que tienen los Algoritmos Genéticos. Gracias al éxito en estas y otras áreas, los Algoritmos Genéticos han llegado a ser un campo puntero en la investigación actual.

6. Bibliografía

Banzhaf W. Reeves C. (editors) (1999) Foundations of Genetic Algorithms.5 Morgan Kaufmann Publishers

Bauer, R.J. (1994) Genetic Algorithms and investment strategies Wiley Finance Edition

Goldberg D.E. (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley

Langdon W.B., Poli R. (2002) Foundations of Genetic Programming. Springer

Mitchell M. (1998) An introduction to Genetic Algorithms. MIT Press