



TESIS DE MASTER

Computación evolutiva y vida artificial

Sergio Sánchez

Máster Automática y Robótica

5/4/2010

- II. Computación evolutiva.

La computación evolutiva es una vertiente de la inteligencia artificial que usa como conceptos de inspiración los de la **evolución biológica** con la finalidad de resolver problemas de optimización generalmente de muy alta complejidad. Hasta hace relativamente poco tiempo se pensaba que el mayor resolutor de problemas era el ser humano, sin embargo, actualmente se conoce a la naturaleza junto con su selección natural que ha creado al hombre como el mayor resolutor. Por ello la computación evolutiva interpreta la naturaleza como una gran máquina de resolver problemas y trata de encontrar el origen de dicha potencialidad para utilizarla en nuestras propuestas.

Observando brevemente los inicios y la evolución a grandes rasgos de estas iniciativas, vemos que aún son técnicas muy jóvenes sobre las que constantemente se realizan nuevos estudios. Ya en 1932 **Cannon** visualizó la evolución natural como un proceso de aprendizaje. Posteriormente **Alan Turing** reconoció, en 1950, que debe haber una conexión obvia entre el aprendizaje de máquina y la evolución, y señaló que se podrían desarrollar programas para jugar ajedrez usando esta técnica. Más tarde en 1960 **Campbell** conjeturó que en todos los procesos que llevan a la expansión del conocimiento, se involucra un proceso ciego de variación y supervivencia selectiva.

Los primeros intentos de aplicar de manera formal la teoría de la evolución, apareció en las áreas de control de procesos estadísticos, aprendizaje de máquina y optimización de funciones. Tal vez el primer intento serio de este tipo se dio en el trabajo que realizaron **Box** y sus colegas en 1957, en el desarrollo de una técnica que denominaron operación evolutiva, la cual se aplicó a una planta de manufactura para manejarla.

Posteriormente **Friedberg** intentó, en 1958, hacer que un programa en lenguaje máquina se mejorara a sí mismo, seleccionando instrucciones que se asociaran más frecuentemente con un resultado exitoso. Poco antes, en el 1954, **Barricelli** ofreció, una de las primeras simulaciones que usaba principios evolutivos, utilizando los mismos procedimientos generales que se usan hoy en día en la disciplina estudiada anteriormente, la vida artificial.

Fogel el que introdujo la primera técnica evolutiva que realmente funcionó más o menos dentro de los parámetros con los que se conoce vincula la computación evolutiva. Su programación evolutiva consistía en hacer evolucionar autómatas de estados finitos por medio de mutaciones artificiales. Veremos más en detalle en apartados posteriores.

Rechenberg y Schwefel en el 1965 desarrollaron una técnica llamada estrategia evolutiva, se utilizó inicialmente para resolver problemas de ingeniería que desafiaban a los métodos de optimización tradicionales, con resultados muy buenos en algunas aplicaciones.

Como se aprecia en lo anteriormente comentado, la historia de estas técnicas es relativamente corta. Varias corrientes de investigación independientes comenzaron a formar lo que ahora se conoce como computación evolutiva y que a día de hoy ha derivado en tres vertientes:

- Programación evolutiva
- Estrategias Evolutivas
- Algoritmos genéticos

-2.1 Bases de la computación evolutiva

Las estructuras algorítmicas usadas en la computación evolutiva se conocen como algoritmos evolutivos. Dentro de la clasificación anterior se les vinculan algunos conceptos comunes.

En un Algoritmo Evolutivo se definirá una estructura de datos que admita todas las posibles soluciones a un problema. La forma de plantear esas posibles soluciones es lo que se conoce como la **representación del problema**, que puede hacer que la resolución sea factible o incluso que no sea posible, por lo que es necesario dedicarle tiempo a pensar en la más adecuada.

Cada uno de los posibles conjuntos de datos admitidos por esa estructura será una solución al problema. La calidad de la solución estará comprendida en un rango determinado, conocido a priori o no. Solucionar el problema consistirá en encontrar la solución óptima, dentro del rango definido como soluciones óptimas o dentro de unos parámetros o restricciones de computación determinados. Es por esto que se engloban en la clasificación de métodos de búsqueda.

Las soluciones al problema son capaces de reproducirse entre sí, combinando sus características y generando nuevas soluciones. En cada **ciclo** se seleccionan las soluciones que más se acercan al objetivo buscado, eliminando el resto de soluciones. Las soluciones seleccionadas se reproducirán entre sí, permitiendo de vez en cuando alguna modificación al azar durante la reproducción.

Por norma general, para generar una simulación de un proceso evolutivo será necesario:

- Codificar las estructuras o individuos.
- Determinar las operaciones que actuarán sobre cada uno de ellos.
- Definir y calcular la función de aptitud.
- Implementar un mecanismo de selección.

-2.2 Programación evolutiva

La programación evolutiva surge inicialmente de la mano de Lawrence J. Fogel y sus compañeros, los cuales concibieron el uso de la **evolución simulada** en la solución de problemas en su mayoría de predicción. Esta técnica que recibió el nombre de programación evolutiva estaba basada en hacer evolucionar autómatas de estados finitos que expuestos a una serie de cadenas de entrada que codificaban el ambiente, se esperaba que fueran capaces de predecir las futuras secuencias de símbolos que vendrían posteriormente.

Se utilizó para ello la función de **recompensa** que indicaba el grado de idoneidad de la respuesta del autómata para un determinado símbolo. Se aplicaba del mismo modo un operador de mutación para efectuar cambios en las mutaciones y eventualmente generar nuevos estados en los autómatas con tendencia a hacerlos progresivamente más aptos en la tarea de la predicción de las secuencia de símbolos. No se utilizaba ninguna técnica de reproducción sexual puesto que se pretendía evolucionar a nivel de especie y no a nivel de individuos.

Esta técnica se utilizó originalmente en problemas de **predicción**, de control automático, de identificación de sistemas y de teoría de juegos. Actualmente se llevan a cabo proyectos en las áreas de la generalización, entretenimiento, problemas de gestión de rutas, diseño y entrenamiento de redes neuronales y reconocimiento de patrones además de los anteriormente mencionados.

Donald W. Dearholt y algunos otros investigadores, experimentaron con programación evolutiva en la Universidad de Nuevo México en la década de los 70, de forma totalmente independiente a Fogel . Probablemente la programación evolutiva fue la **primera técnica** basada en la **evolución** aplicada a problemas de predicción, además de ser la primera en usar codificaciones de longitud variable ya que el número de estados de los autómatas podía variar tras la mutación. Constituye por tanto uno de los primeros intentos por simular la evolución.

-2.3 Estrategias evolutivas

Se conoce como estrategias evolutivas a los métodos computacionales que trabajan con una población de individuos que pertenecen al dominio de los números reales, que mediante los procesos de mutación y de recombinación evolucionan para alcanzar el óptimo de la función objetivo. Son, de este modo, algoritmos evolutivos enfocados hacia la **optimización de parámetros**, teniendo como características principales que utilizan una representación a través de vectores reales, una selección determinística y operadores genéticos específicos de cruce y mutación. Además, su objetivo fundamental consiste en encontrar el valor real de un vector de N dimensiones.

Cada individuo de la población es un posible óptimo de la función objetivo; la representación de cada individuo de la población consta de 2 tipos de variables: las variables objeto y las variables estratégicas. Las variables objeto son los posibles valores que hacen que la función objetivo alcance el óptimo global y las variables estratégicas son los parámetros mediante los que se gobierna el proceso evolutivo o, dicho de otro modo, las que indican de

qué manera las variables objeto son afectadas por la mutación. El genotipo en las estrategias evolutivas será por tanto el conjunto formado por las variables objeto y las variables estratégicas. Y el fenotipo por otro lado serán las variables objeto, ya que conforme se da la variación de éstas, se percibe un mejor o peor desempeño del individuo.

Las estrategias evolutivas fueron propuestas por Ingo Rechenberg y Hans-Paul Schwefel en la década de 1970 con el principal objetivo de optimizar de parámetros.

Las estrategias evolutivas pueden dividirse en dos tipos: simples y múltiples.

- Estrategias evolutivas simples

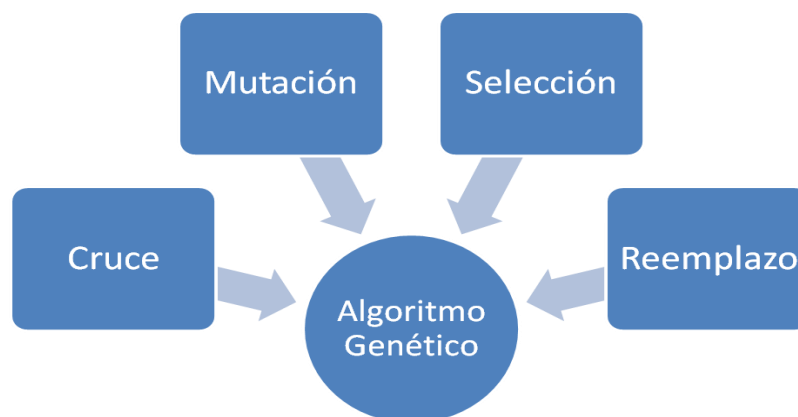
Son consideradas como procedimientos estocásticos de optimización paramétrica con **paso adaptativo**, esta característica las hace similares al temple simulado. En este caso, se hace evolucionar un solo individuo usando únicamente a la mutación como operador genético. Son relativamente sencillas, y se denominan también estrategias evolutivas de dos miembros. Esto es debido a que evoluciona un solo individuo a la vez, no son consideradas estrictamente como métodos evolutivos. A pesar de ser muy sencillas, son de gran utilidad práctica y han sido utilizadas, con algunas mejoras, para resolver problemas reales en diversas áreas.

- Estrategias evolutivas múltiples

Surgen como respuesta a las debilidades de las estrategias evolutivas simples, las cuales tienden a converger hacia óptimos locales. En esta área, existen múltiples individuos que se reproducen en cada generación generando varios nuevos individuos. Esta estrategia usa tanto mutación como cruce. Generalmente se usa el cruce **promedio entre dos padres**, el cual genera un único descendiente, promediando los valores de estos. En cuanto a los criterios de reemplazo, siempre se usa un esquema determinístico pudiéndose utilizar una estrategia de inserción o de inclusión.

-2.4 Algoritmos genéticos

Los algoritmos genéticos fueron propuestos por John H. Holland en 1975 y su motivación inicial fue la de proponer un modelo general de proceso adaptable.



El algoritmo básico de la programación evolutiva sigue el siguiente patrón, donde algunos de los operadores pueden o no actuar:

- Generar aleatoriamente una población inicial.
- Se aplica mutación.
- Calculo de la aptitud de cada hijo en base a uno o más enfoques
- Se selecciona mediante alguna técnica de selección las soluciones que se mantendrán.

El algoritmo genético tiene un gran número de variaciones, se presenta una síntesis de las técnicas más utilizadas junto con alguno de los conceptos que se tienen que tener en cuenta con vistas a ponerlas en práctica en la definición del proyecto y en la propia aplicación posterior.

- La representación

Una de las partes más importantes a tener en cuenta es la representación del problema, es decir, la forma en la que se codifican los individuos o las soluciones parciales de la población actual y que posteriormente el resto de elementos que intervienen se verán determinados y la solución puede verse modificada tanto en el proceso de obtención como en el propio resultado.

Entre las numerosas representaciones posibles del problema algunas se presumen como la **representación natural** y otras pueden resultar soluciones ilegales como aquellas que no se plantean con posibilidad de llegar a una solución factible.

Entre las representaciones posibles se tienen en cuenta los **códigos de Gray** que intentan homogeneizar las operaciones de cruce, selección y mutación para que se acerquen lo más posible a sus homólogas biológicas donde el grado de aplicación de la operación influye directamente en la variación de las características del individuo. Esto evita en la medida de lo posible el desvío producido por la representación binaria, real o de cualquier otro tipo, donde los elementos más altos de la cadena suelen venir ponderados por la base en la que se representen.

Se puede codificar bajo estructuras de información o cromosomas fijos, o de **longitud variable**, cada uno conlleva una representación diferente. Del mismo modo además de las representaciones naturales descritas generalmente como tipos básicos, es posible utilizar cualquier estructura de datos como listas, pilas o árboles entre otras, donde estas últimas cobran especial importancia en el ámbito de la computación evolutiva.

- Selección

La selección decide indiscutiblemente que criaturas formarán parte del siguiente paso evolutivo, es decir, de la población resultante de aplicar el algoritmo genético. La selección fija de forma unívoca aquellos elementos y características que perdurarán en el proceso de búsqueda.

Aunque se encuentran muchas técnicas de selección, las más importantes se muestran en el gráfico inferior. La finalidad no es la de describir en detalle cada uno, para lo que existe gran cantidad de material disponible de forma online, sino la de sintetizar aquellas que pueden tener más relevancia para el propio proyecto y pueden ser utilizadas en la parte práctica.



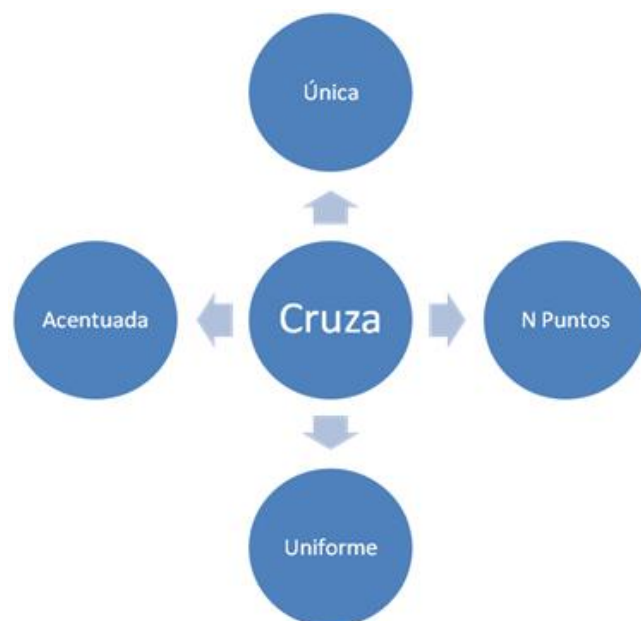
La selección puede definir tanto quien se implica en la reproducción como en cualquier penalización o proceso alternativo.

- Reproducción o cruza

Este operador tiene la finalidad de adquirir y fusionar características de dos o más elementos de la población generando nuevos individuos que tienen características comunes a las de sus padres.

Existe un gran número de variantes, del mismo modo que el operador anterior, se listan de forma genérica teniendo en cuenta que es importante definir la cruza para la representación concreta en la que se trabaja. En el plano genérico, las cruza serán las del gráfico de la derecha.

En el plano del proyecto que se planteará en apartados posteriores, el operador de cruza será aplicado a la representación real por lo que se aplicarán técnicas concretas de esta representación como las



anteriormente genéricas comentadas y las conocidas como intermedia, aritmética simple, aritmética total o el simulated binary crossover. Estas últimas son especialmente aplicables a los reales, que representan un gran número de aplicaciones y la mayoría de las variables que veremos posteriormente en el planteamiento del proyecto.

- Mutación

El operador de mutación modifica de forma moderada diferentes elementos de los individuos de la población. Generalmente se considera como un operador secundario dado que se aplica con moderación respecto al resto de operadores.

Existen varias técnicas de mutación dependiendo de la representación del problema, en este caso se resumen las que son aplicables al ámbito de la representación real.

Algunas técnicas a tener en cuenta se basan en aplicar tasas de mutación variable hasta el punto de incluso fijar una tasa de mutación diferente para cada individuo o en extremo para cada gen. Se tienen en cuenta modificaciones como hacer que sea más probable que se produzca una mutación en un gen si en su vecino ya se ha producido. En este caso y para el proyecto posterior, se tendrán en cuenta como veremos tasas de mutaciones propias a cada organismo como parte adicional de su metabolismo.



- Ajuste de parámetros

De la definición de los parámetros principales se desprende en la mayoría de las ocasiones el grado de desempeño del algoritmo por lo que se le da mucha importancia al ajuste de los mismos. Los principales elementos a ajustar son:

- El número de individuos de la población.
- El porcentaje de cruza.
- La tasa de mutación.

Estos parámetros interactúan de forma no lineal por lo que no pueden optimizarse Independientemente y es por eso que han surgido algunas técnicas de ajuste de parámetros para estos algoritmos.

Experimentos de De Jong y Goldberg establecieron el paralelismo entre la longitud de la representación y el tamaño de la población.

Uno de los mecanismos de ajuste de parámetros más útil es de la **autoadaptación**, que mediante determinadas técnicas añaden al espacio del problema, las variables correspondientes a los principales parámetros del algoritmo, teniendo en cuenta la aptitud media de los individuos con determinadas tasas.

Existe multitud de técnicas de ajuste de parámetros que requieren un grado añadido de complejidad al problema, sin embargo, es de extrema utilidad definir un mecanismo que optimice la adaptación puesto que en determinados problemas que cargan con un coste computacional muy alto este ajuste puede reducir ampliamente el tiempo de adquisición de una solución buena.

- Manejo de restricciones

En el algoritmo genético se plantean una serie de elementos que deben ponerse en juego para obtener la solución sin tener en cuenta que determinados procesos pueden llevar a soluciones no válidas que predominen sobre el resto. Es el caso de simulaciones de desplazamiento en cuerpos físicos donde simplemente mutar sobre el tamaño haciendo que el elemento sea más grande implicará un mayor desplazamiento inicial independiente de su aptitud real y por lo tanto se verá desarrollado en incrementos progresivos de tamaño dejando de lado el resto de características que realmente son importantes.

Es necesario por tanto aplicar **penalizaciones** sobre cada restricción que no sea respetada. Las penalizaciones pueden darse en base a 3 formas principales:

- Teniendo en cuenta su infactibilidad únicamente.
- Valorando lo lejos que está de una solución factible.
- Evaluando el esfuerzo de reparar al individuo para tonarlo factible.

Diseñar reglas de penalización implica conocer

- Es necesario conocer la 'distancia' a la solución factible.
- En problemas con pocas variables y pocas restricciones, penalizaciones que solo tienen en cuenta el número de restricciones no producirán resultados satisfactorios.
- El costo máximo previsto de hacer una función infactible en factible, el **costo de cumplimiento máximo**, debe ser tenido en cuenta.
- El **costo de cumplimiento esperado**, del mismo modo que el anterior, debe indicar cotas superiores de costes. No se debe caer frecuentemente por debajo de este coste puesto que las soluciones no serán alcanzadas.

Algunas de las restricciones que suelen aplicarse se muestran en resumen en el gráfico siguiente a las que sería conveniente agregar las técnicas de optimización multiobjetivo, quedarían fuera del ámbito del proyecto:



En el ámbito del trabajo será extremadamente importante tener en cuenta lo aprendido anteriormente puesto que al simular entornos físicos se deberán restringir un gran número de parámetros e incluso de comportamientos.

- Paralelismo

En el uso del algoritmo genético se espera generar una gran cantidad de requisitos de cálculo por lo que estas técnicas son muy importantes y se encuentran en continuo progreso. En ellas intervienen los conceptos de **deme** como entidad poblacional o la **granularidad** como el grado y aplicación de paralelismo en el algoritmo.

Puesto que en el proyecto práctica se requerirá gran capacidad de cálculo, se hará especial énfasis en este apartado y se propondrán algunas técnicas que pueden ser útiles.

- Usos directos

El uso de los algoritmos genéticos, a pesar de que es una técnica relativamente joven, se está empleando en numerosos proyectos de muy diferentes ámbitos como el diseño automatizado, la investigación en diseño de materiales y diseño multiobjetivo de componentes automovilísticos, mejor comportamiento ante choques, ahorros de peso, mejora de aerodinámica, etc.

No solo la industria automovilística, ya pionera en emplear técnicas de vanguardia, se beneficia de estas técnicas sino que son empleadas del mismo modo en diseño automatizado de equipamiento industrial, optimización de carga de contenedores, diseño de sistemas de distribución de aguas, diseño de topologías de circuitos impresos y diseño de topologías de redes computacionales entre otros muchos usos como la teoría de juegos.

- Resumen de los diferentes paradigmas

En definitiva, las características de cada una de las propuestas evolutivas son las que se pueden apreciar en la tabla inferior. Actualmente dada la multitud de usos y aplicaciones de complejidad creciente, resultan menos identificables y acaban tomando matices comunes.

	Estrategias Evolutivas	Programación Evolutiva	Algoritmo Genético
Representación	Real	Real	Binaria
Función de Aptitud	Valor de la función objetivo	Valor de la función objetivo ajustada	Valor de la función objetivo ajustada
Auto-Adaptación	Desviaciones Estándar y ángulos de rotación	Ninguna Varianzas (PE-estándar), Coeficientes de correlación (meta-PE)	Ninguna
Mutación	Gausiana, operador principal	Gausiana, operador único	Inversión de bits, operador secundario
Recombinación	Discreta e intermedia, sexual y panmítica, importante para la auto-adaptación	Ninguna	Cruza de 2-puntos, cruza uniforme, únicamente sexual, operador principal
Selección	Determinística, extintiva o basada en la preservación	Probabilística, extintiva	Probabilística, basada en la preservación
Restricciones	Restricciones arbitrarias de desigualdad	Ninguna	Límites simples mediante el mecanismo de codificación
Teoría	Velocidad de Convergencia para casos especiales, (1+1)-ES, (1+ λ)-ES, Convergencia global para ($\mu + \lambda$)-ES	Velocidad de Convergencia para casos especiales, (1+1)-PE, Convergencia global para meta-PE	Teoría de los Esquemas, Convergencia global para la versión elitista

Figura 6. Características de los paradigmas de computación evolutiva.

En definitiva la computación evolutiva se trata de otro método de búsqueda sobre el que se han ido generando paradigmas y propuestas. Los más relevantes a grandes rasgos quedan definidos en el siguiente esquema, que incluye la taxonomía de los diferentes procedimientos de búsqueda más utilizados en la actualidad:

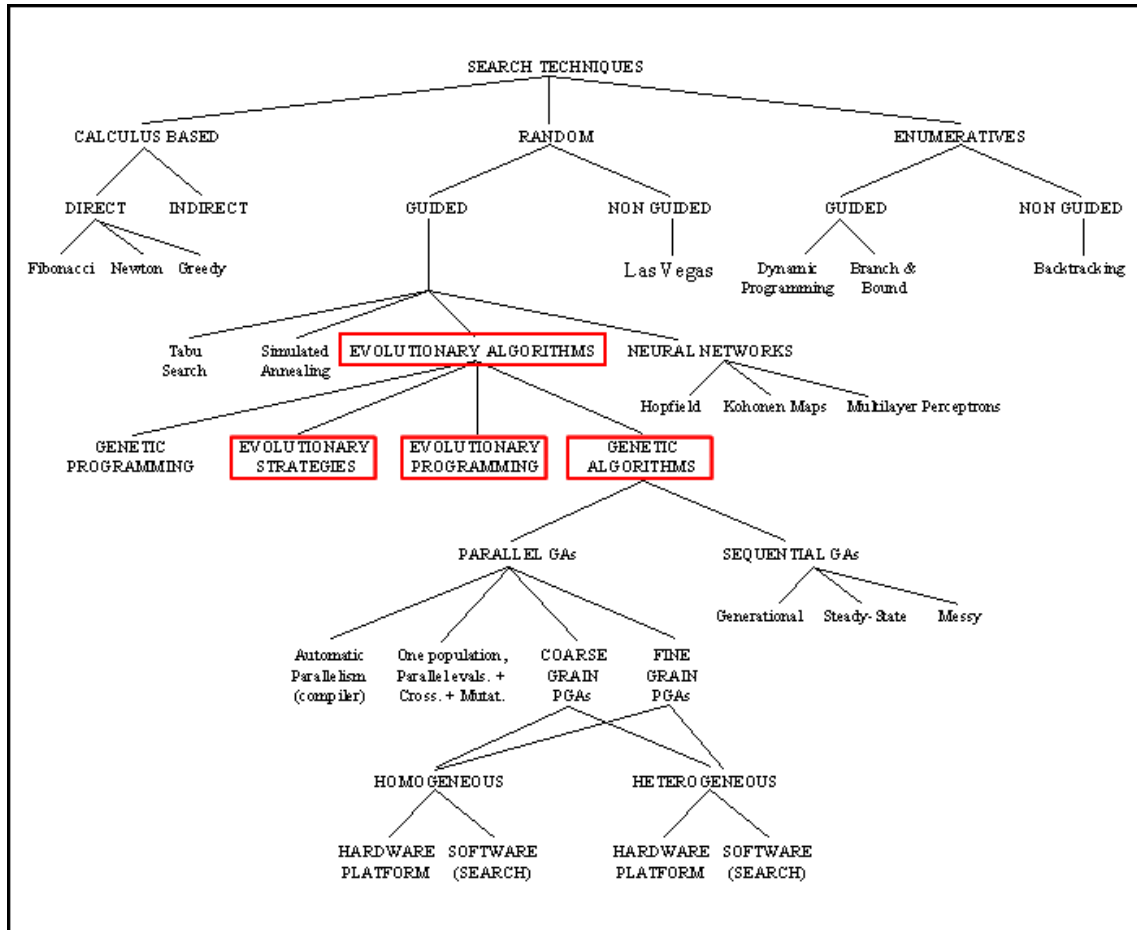


Figura 7. Distribución de las ramas estudiadas en el área de las técnicas de búsqueda.

Como se aprecia en la figura, los esfuerzos en técnicas genéticas se orientan tanto a profundizar en la eficiencia de la técnica como en aumentar o potenciar el grado de paralelismo de las mismas.

-2.5 En conclusión

La computación evolutiva representa una técnica de búsqueda muy útil en aquellos problemas en los que el **espacio** es muy **amplio** y las guías de resolución nos dan poca información del problema o son tan complejas que hacen inviable una búsqueda directa.

La programación genética o las estrategias evolutivas representan vertientes concretas de la computación evolutiva que son apropiadas para determinados problemas. Enfocando al proyecto que se propondrá en apartados posteriores, las técnicas más apropiadas se plantean las relativas al algoritmo genético y al paradigma propio que posee por lo que el resto de técnicas no serán utilizadas directamente.

En anteriores apartados se han comentado varias técnicas sobre cada operador relevante con la intención de explicitar aquellas que pueden ser de utilidad en el **desarrollo práctico**. Es por eso que en apartados posteriores se retomarán y aplicarán los conceptos anteriormente expuestos aplicados directamente sobre el propio proyecto.

En la aplicación de este proyecto se utilizarán los algoritmos genéticos como principal **motor de progreso** de las diferentes características de los individuos en base al grado de aptitud del que disponga cada una. Este grado de aptitud puede ser tanto la energía consumida, como el tiempo de supervivencia o el desplazamiento en caso de evaluar las habilidades motrices.

La investigación en esta materia crece de forma continua tanto que actualmente está siendo utilizada en **multitud de proyectos** tanto de investigación como comerciales. Entre ellos destacan las aplicaciones sobre el aprendizaje de comportamientos de robots, aprendizaje en reglas de lógica difusa, compresión de datos, predicción de plegamiento de proteínas entre otras muchas.

Resulta interesante conocer y aplicar estas técnicas. En el proyecto de apartados posteriores se utilizará en detalle todo este conocimiento adquirido y muchas de las técnicas propuestas en este apartado.