Introducción a la Computación Evolutiva y la Morfogénesis Artificial

Gabriela Ochoa

Automated Scheduling, Optimisation and Planning Group, School of Computer Science, University of Nottingham, Nottingham, UK

Grupo de Computación en Medicina y Biología, Universidad Simón Bolívar, Caracas, Venezuela

La computación evolutiva es una rama de la computación y la inteligencia artificial que comprende métodos de búsqueda y aprendizaje automatizado inspirados en los mecanismos de la evolución natural. Diversos enfoques a la computación evolutiva han sido propuestos: las estrategias evolutivas, los algoritmos genéticos, la programación genética y los clasificadores genéticos entre otros. A estos métodos se les denomina de manera colectiva como *algoritmos evolutivos*, entre los cuales los más conocidos son probablemente los *algoritmos genéticos*. Estos algoritmos han sido aplicados exitosamente en la resolución de problemas en distintas ramas de la ingeniería, el diseño, la industria, la economía y las ciencias naturales.

Este capítulo está estructurado en tres secciones principales. La primera sección describe el funcionamiento y los componentes principales de los algoritmos evolutivos, mientras que la segunda sección describe el formalismo matemático denominado *sistemas de Lindenmayer*, el cual se utiliza para modelar el desarrollo y crecimiento de sistemas biológicos. La tercera sección presenta una combinación de estas dos herramientas computacionales con inspiración biológica, específicamente, un ejemplo que utiliza un algoritmo evolutivo en el modelaje de la evolución de formas vegetales artificiales. Este ejemplo ilustrativo puede enmarcarse dentro de la disciplina de la *vida artificial*, un área relacionada con la computación evolutiva cuyo objetivo principal no es el de resolver problemas sino el de simular en el computador procesos relacionados con la vida con el fin de comprenderlos mejor. En particular, el modelo elegido combina el estudio de la morfogénesis y la simulación de la evolución. Concluimos el capítulo con algunas consideraciones finales en relación a las disciplinas de la computación evolutiva y la vida artificial.

La computación evolutiva

La computación evolutiva emula a la evolución natural en el diseño e implementación de herramientas computacionales para la resolución de problemas. A partir de 1960, varios modelos de computación evolutiva han sido propuestos y estudiados, a los cuales se les denomina colectivamente como algoritmos evolutivos (Eiben y Smith, 2003). Estos algoritmos han sido aplicados a una amplia variedad de problemas encontrados tanto en la industria y el comercio, como en la investigación científica de punta. Dado que otros capítulos en este volumen describen los principios de la evolución natural, estos no son mencionados aquí. Es suficiente decir que los algoritmos evolutivos emulan a la evolución natural y comprenden:

- Una representación o codificación de las soluciones potenciales al problema bajo estudio
- Una población (conjunto de individuos) de estas soluciones potenciales

- Mecanismos para generar nuevos individuos o soluciones potenciales al problema estudiado, a partir de los miembros de la población actual (los denominados operadores de mutación y recombinación)
- Una función de desempeño o evaluación (del inglés *fitness function*) que determina la calidad de los individuos en la población en su capacidad de resolver el problema bajo estudio
- Un método de selección que otorgue mayores chances de sobrevivir a las buenas soluciones

Naturaleza	**Algoritmos Evolutivos**
Individuo	Solución al problema
Población	Conjunto de soluciones
Adecuación (*fitness*)	Calidad de la solución
Cromosoma	Representación o codificación de una solución
Gen	Parte o componente de la representación de una solución
Mutación y recombinación	Operadores de búsqueda
Selección natural	Preservación o re-utilización de buenas soluciones (o sus componentes)

Tabla1: Analogías entre aspectos de la evolución natural y los algoritmos evolutivos

La Tabla 1 muestra las analogías entre la evolución natural y los algoritmos evolutivos, mientras que a Figura 1 ilustra el esquema general de un algoritmo evolutivo. Una población de *M* individuos es inicializada y sujeta a evolución simulada de una generación a la siguiente a través de la aplicación sucesiva de los operadores de selección de progenitores, recombinación, mutación, evaluación de la función de desempeño y selección de sobrevivientes. Este ciclo es interrumpido luego de un número fijo de iteraciones (generaciones), o cuando la solución encontrada es de calidad aceptable para el problema bajo estudio.

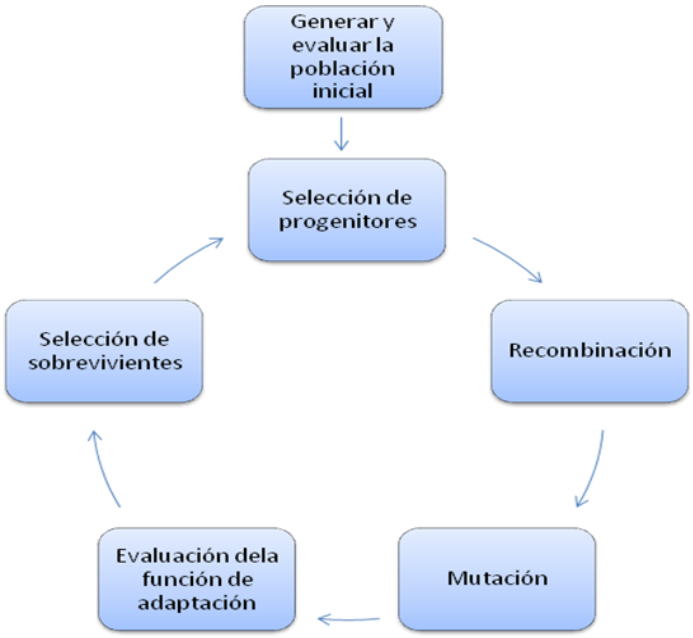


Figura 1: Esquema general de un algoritmo evolutivo

Históricamente, existen tres enfoques bien definidos, con origen independiente, a la computación evolutiva: la programación evolutiva (Fogel, Owens, y Walsh, 1966), las estrategias evolutivas (Rechenberg, 1973) y los algoritmos genéticos (Holland, 1975). Aunque similares a nivel conceptual, estos enfoques difieren en la manera de implementar el algoritmo evolutivo. Estas diferencias tocan distintos aspectos del algoritmo incluyendo la manera de representar los individuos, los mecanismos de selección, y el énfasis y tipo de operadores de variación genética utilizados. Aunque no es posible hacer en este capítulo una revisión rigurosa de todos los enfoques recientes a la computación evolutiva, cabe mencionar los algoritmos genéticos con representación ordinal (Goldberg, 1989), los clasificadores genéticos (Holland, 1986) y la programación genética (Koza, 1992), como ramificaciones de los algoritmos genéticos que se han desarrollado y aplicado ampliamente. Los algoritmos genéticos con representación ordinal se utilizan en la resolución de problemas de optimización combinatoria, donde el espacio de búsqueda se compone de permutaciones (ordenamientos de los números enteros), como por ejemplo el famoso problema del agente viajero (que debe recorrer una ruta de varias ciudades, volviendo a la ciudad de origen, mientras minimiza la distancia o el costo de la ruta). En este caso los individuos se codifican como permutaciones de los números enteros y los operadores genéticos (Ej. Inversión y reordenamiento) deben diseñarse de manera de producir permutaciones. Los sistemas clasificadores genéticos utilizan un algoritmo evolutivo para explorar el espacio de reglas de producción de un sistema de aprendizaje capaz de inducir nuevo conocimiento y generalizar. Finalmente, la programación genética aplica búsqueda evolutiva en el espacio compuesto por programas de computadora en un lenguaje de programación que puede modificarse utilizando operadores de mutación y recombinación. Cabe mencionar que en la actualidad hay gran interacción y comunicación entre los investigadores de los distintos enfoques a la computación evolutiva; las fronteras entre estos métodos han desaparecido hasta cierto punto.

A continuación se presenta una descripción más detallada de los algoritmos evolutivos más comúnmente utilizados, los denominados algoritmos genéticos:

Representación genética o codificación

Para poder aplicar algoritmos genéticos a un problema dado, es necesario poder representar a las soluciones potenciales al mismo como un conjunto de parámetros o componentes (llamados *genes*). Estos parámetros o componentes se juntan para formar una secuencia o cadena que representa una solución o individuo; a la que se denomina también *cromosoma* o *genotipo*. Nótese que la terminología de las ciencias biológicas es utilizada para denominar los componentes análogos (ampliamente simplificados) en los algoritmos genéticos. Tradicionalmente, la representación en los algoritmos genéticos consiste en cadenas de caracteres binarios. Esta cadena de dígitos binarios o bita es posteriormente decodificada pare constituir los parámetros o componentes del problema en cuestión (Figura 2). Sin embargo, muchas otras representaciones, como las mencionadas permutaciones, conjunto de reglas, estructuras ramificadas, vectores de números reales, programas por computadora, etc. han sido utilizadas exitosamente.

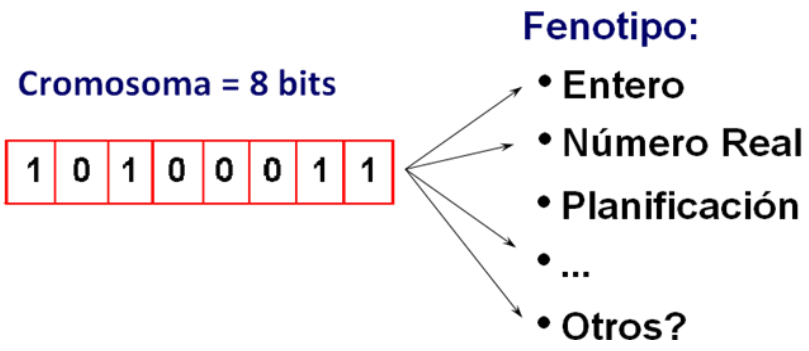


Figura 2: La representación genética puede basarse en valores discretos; binarios, enteros u otro sistema o alfabeto con un conjunto discreto de valores. El cromosoma o genotipo es posteriormente decodificado en los parámetros o componentes asociados al problema bajo estudio (fenotipo).

Función de evaluación o desempeño

Una función de desempeño (también llamada función de evaluación, función objetivo o función de *fitness*) debe diseñarse para cada problema a ser resuelto. El propósito de esta función es el de medir la calidad de los cromosomas en su capacidad de resolver el problema bajo estudio. Para algunos problemas (Ej. La optimización de funciones reales o la optimización combinatoria) la definición de la función objetivo resulta evidente y sencilla, pero esto no es así en aplicaciones del mundo real, donde la definición de la función de desempeño puede requerir la implementación de un complejo modelo de simulación.

Operadores genéticos

Los operadores genéticos introducen diversidad genética en la población; su propósito es generar nuevos individuos a partir de los individuos existentes en la población actual. Los algoritmos genéticos contienen dos tipos principales de operadores: la recombinación y la mutación los cuales están inspirados en los mecanismos análogos de la reproducción en la naturaleza. Cada operador tiene un parámetro asociado que controla la probabilidad de su aplicación en cada iteración (generación) del algoritmo. El cómo asignar estos parámetros es todavía tema de investigación, ya que es conocido que el funcionamiento del algoritmo es muy sensible a los valores de los mismos.

Mutación

Cuando la representación genética consiste en una cadena de bits, el operador de mutación simplemente altera un bit, es decir, lo cambia de 0 a 1 o viceversa. La probabilidad de que un bit sea alterado depende de un parámetro: la probabilidad o tasa de mutación. Los bits de la cadena son mutados de manera independiente, es decir, la mutación de un bit no altera la probabilidad de mutación de los otros bits (Figura 3). Para representaciones distintas a las cadenas de bits, otros operadores de mutación deben ser diseñados. La idea detrás de un operador de mutación es la de realizar una alteración o perturbación pequeña al genotipo, para generar un individuo ligeramente distinto pero relacionado con el progenitor. Tradicionalmente, la mutación se considera como un operador secundario en los algoritmos genéticos, cuyo rol es el de restaurar el material genético perdido, siendo la recombinación el principal operador de búsqueda. Sin embargo, algunos investigadores sostienen que un método basado en mutación y selección resulta un algoritmo de búsqueda poderoso; y que el rol de la mutación en los algoritmos genéticos ha sido sub-estimado mientras que el de la recombinación sobre-estimado.

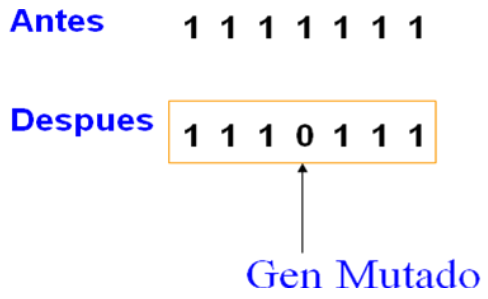


Figura 3: El operador de mutación sobre cadenas de bits, se aplica con probabilidad p_m para cada bit

Recombinación

La recombinación o *crossover* se considera el operador de búsqueda principal en los algoritmos genéticos. Este operador produce individuos descendientes a partir de combinar o mezclar el material genético de dos (o más) individuos progenitores. La motivación detrás de este operador es que la mezcla de sub-partes de los progenitores puede crear nuevos individuos con combinaciones favorables de genes. La aplicación de la recombinación es controlada por un parámetro (la probabilidad o tasa de cruce). Varios operadores de recombinación han sido propuestos en la literatura. Los más conocidos son los de un punto, dos puntos, múltiples puntos y uniforme. En la recombinación de un punto, un único punto de corte es seleccionado aleatoriamente en los progenitores; luego los segmentos antes y después del punto de corte son intercambiados. La recombinación de múltiples puntos es una generalización de esta idea que introduce varios puntos de corte e intercambia los segmentos entre dichos puntos. En la recombinación uniforme, los segmentos intercambiados se reducen a bits únicos. En lugar de puntos de corte, se sortea aleatoriamente de cual progenitor proviene cada bit. La figura 4 ilustra la operación de la recombinación de dos-puntos (izquierda) y uniforme (derecha) sobre cadenas de bits. Varios trabajos han comparado el desempeño de los distintos operadores de recombinación, sin embargo, no hay evidencia concluyente sobre cual es mejor. Es probable que la elección del operador más adecuado sea dependiente del problema. Parece haber, sin embargo, un consenso general respecto a las ventajas de los operadores de dos puntos y uniforme sobre el operador de un punto. Tal como ocurre con la mutación, para representaciones genéticas distintas de las cadenas de bits, operadores de recombinación especializados y capaces de preservar la estructura genética, deben ser diseñados.

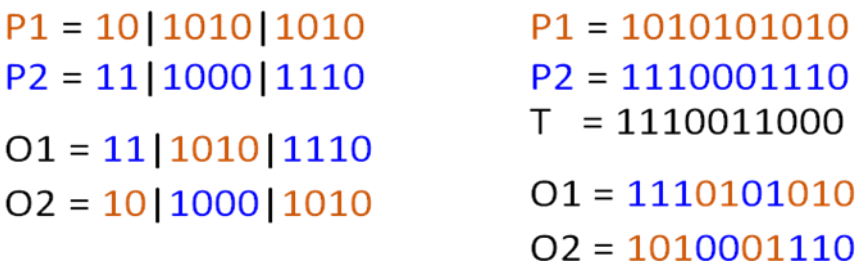


Figura 4: Ilustración de los operadores de recombinación más comúnmente utilizados para la representación binaria. **Izquierda**: recombinación de dos puntos, se seleccionan 2 puntos de corte, y se intercambian los segmentos entre ellos. **Derecha**: recombinación uniforme, se intercambian bits, para cada bit del hijo se selecciona aleatoriamente de qué padre procede.

Selección

La selección se encarga de asignar oportunidades de reproducción a cada uno de los miembros de la población. Mientras más alto sea el valor de la función de desempeño de un individuo, más

oportunidades tendrá de reproducirse. En los algoritmos evolutivos, la selección debe ser balanceada con los operadores de variación (mutación y recombinación). Todo algoritmo de búsqueda debe balancear estas dos fuerzas opuestas: la *exploración* de nuevas zonas en el espacio de búsqueda, y la *explotación* de zonas promisorias del mismo que ya han sido descubiertas. La selección es el componente que determina principalmente las características del proceso de búsqueda. Si la selección es muy fuerte, individuos de calidad sub-óptima pueden dominar la población y por lo tanto reducir la diversidad genética requerida para el cambio y el avance; por otro lado, una selección muy débil puede resultar en un proceso evolutivo muy lento. Varios esquemas de selección han sido propuestos en la literatura: selección proporcional, por escalamiento, por torneo, basada en rango, etc. No existe consenso respecto a cual esquema es preferible, aunque algunos son más sencillos conceptualmente que otros, y por tanto más fáciles de implementar.

Reemplazo poblacional

Pueden distinguirse dos modelos básicos para realizar el reemplazo de una población simulada de una generación o iteración del algoritmo a la siguiente: en el modelo *generacional*, toda la población es reemplazada en cada generación; mientras que en el modelo de *estado-estacionario*, solo pocos individuos son reemplazados en cada generación (típicamente uno o dos). Entre estos dos extremos, se encuentra el llamado "*gap" (brecha) generacional*, el cual define un porcentaje (gap) de los individuos que serán reemplazados cada generación.

Otros aspectos y consideraciones al diseñar algoritmos evolutivos

Para culminar esta sección, nos referimos a algunos componentes adicionales y consideraciones a la hora de diseñar e implementar algoritmos genéticos (o evolutivos).

Criterio de terminación: El ciclo evolutivo debe interrumpirse en algún momento para obtener las soluciones al problema estudiado; tradicionalmente el denominado *criterio de terminación* en los algoritmos genéticos consiste en un número fijo de iteraciones o generaciones del algoritmo. Frecuentemente un número fijo de evaluaciones de la función objetivo es considerado en lugar de un número fijo de generaciones. Esto permite la comparación en cuanto al desempeño de algoritmos bajo el modelo generacional y el modelo de estado-estacionario, así como la comparación de los algoritmos evolutivos con otros métodos heurísticos. Otros criterios de terminación comúnmente utilizados son: (i) cuando se alcanza un mínimo de diversidad genética en la población, (ii) un tiempo de cómputo fijo, (iii) cuando ha transcurrido cierto número de generaciones sin ninguna mejoría en la función de desempeño .

Medidas del funcionamiento del algoritmo: En vista de que los algoritmos evolutivos tienen un componente estocástico, el desempeño de los mismos no puede basarse en una única corrida. Generalmente deben considerarse estadísticas adecuadas a partir un número suficientemente grande de corridas del algoritmo. Las medidas de calidad del algoritmo son requeridas cuando quieren hacerse comparaciones entre distintos algoritmos, variaciones del mismo algoritmo (incluyendo distintos valores de los parámetros). Distintas medidas se han propuesto para evaluar la calidad de las soluciones obtenidas, así como para medir la velocidad de convergencia de los algoritmos.

Diseño y parametrización del algoritmo: Considerando las diversas posibilidades para los mecanismos de selección, esquemas de reemplazo poblacional, operadores genéticos, criterios de

terminación y valores de los parámetros evolutivos; el algoritmo genético no es un algoritmo único sino una familia de posibles algoritmos. Para complicar aún más el asunto, existen pocos resultados teóricos y sugerencias concretas sobre cómo tomar las decisiones de diseño y asignar los valores de los parámetros evolutivos.

Cuando se quiere aplicar un algoritmo evolutivo en la resolución de un problema dado, dos pasos principales son requeridos: (i) el diseño de una representación o codificación adecuada, y (ii) el diseño e implementación de la función de desempeño. Estos dos componentes conforman el vínculo entre el algoritmo y el problema. Otras decisiones que deben considerarse son: (a) el método de selección a utilizar y sus respectivos parámetros, (b) los operadores genéticos a utilizar, y (c) los valores de los parámetros evolutivos (tamaño de la población y tasas de aplicación de los operadores genéticos).

Los sistemas de Lindenmayer

Los sistemas de Lindenmayer o sistemas-L (del inglés *L-systems*) son conjuntos de reglas de producción (o gramáticas formales) que modelan procesos de crecimiento (Lindenmayer, 1968). Deben su nombre a su creador, el biólogo Aristid Lindenmayer, quien fue uno de los primeros en utilizar métodos sintácticos para modelar el crecimiento. Han encontrado aplicaciones en la comunidad de la Vida Artificial para el análisis del complejo proceso de la morfogénesis, pues constituyen una abstracción de la "forma lógica" de éste fenómeno natural. A partir de su aplicación original en el estudio de la ramificación de estructuras filamentosas en una dimensión, los sistemas-L se han utilizado especialmente en el modelaje realista del desarrollo de estructuras de ramificación en las plantas y en la descripción del crecimiento y desarrollo de membranas o tejidos celulares (Prusinkiewics y Lindenmayer, 1990).

La noción de *reescritura* es fundamental en los sistemas-L, cuya idea básica es la de producir objetos complejos a partir del reemplazo sucesivo de partes de un objeto simple utilizando un conjunto de reglas de reescritura o producción, proceso que puede llevarse a cabo de manera recursiva. Los sistemas de reescritura mejor comprendidos y ampliamente estudiados operan sobre cadenas de caracteres. El trabajo de Chomsky en gramáticas formales difundió un gran interés en los sistemas de reescritura. Posteriormente, comenzó un periodo de fascinación con la sintaxis, las gramáticas y sus aplicaciones a las ciencias de la computación, lo cual dio origen a la disciplina de los *lenguajes formales*.

El trabajo de Lindenmayer introdujo un nuevo tipo de reescritura en cadenas de caracteres que posteriormente se denominó Sistemas-L. La diferencia fundamental entre las gramáticas de Chomsky y los sistemas-L radica en el método de aplicar las reglas de producción. Mientras que en las gramáticas de Chomsky las reglas se aplican de manera secuencial, en los sistemas-L estas se aplican en paralelo, reemplazando simultáneamente todas las letras en una palabra dada. Esta diferencia refleja la motivación biológica de los sistemas-L. Las reglas de producción se diseñan con la intención de capturar las divisiones celulares en organismos unicelulares en los cuales muchas divisiones ocurren al mismo tiempo.

Los sistemas-L deterministas y libres de contexto (sistemas-D0L)

La clase más sencilla de sistemas-L son los denominados sistemas-D0L, donde D representa determinista y 0 corresponde a contexto 0, es decir, libre de contexto. A fin de comprender de manera intuitiva el funcionamiento de los sistemas-D0L, consideremos el siguiente ejemplo presentado en Prusinkiewics y Lindenmayer (1990; ver también la Figura 5):

> Consideremos las cadenas de caracteres constituidas de dos letras *a* y *b*, las cuales pueden ocurrir varias veces en una cadena. Para cada letra se especifica una regla de reescritura. La regla $a \rightarrow ab$ indica que la letra *a* debe ser reemplazada por la cadena *ab*, y la regla $b \rightarrow a$ que la letra *b* se sustituirá por *a*. El proceso de reescritura comienza a partir de una cadena particular conocida como el *axioma*. Asumamos que el axioma en este caso consiste de una única letra *b*. En el primer paso de derivación o reescritura el axioma *b* es reemplazado por *a* utilizando la regla $b \rightarrow a$. En el segundo paso, *a* es reemplazado por *ab* utilizando la regla $a \rightarrow ab$. La palabra *ab* contiene dos letras, las cuales son reemplazadas simultáneamente en el siguiente paso de derivación. Entonces, *a* es reemplazada por *ab* y *b* es reemplazada por *a*, resultando la cadena *aba*. De manera similar (a través del reemplazo simultaneo de todas las letras), la cadena *aba* produce *abaab* la cual a su vez resulta en *abaababa* luego *abaababaabaab* y así sucesivamente.

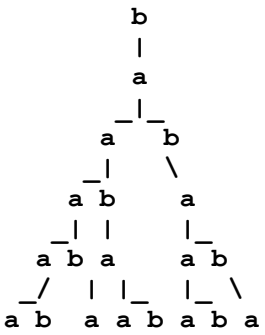


Figura 5: Definición formal y funcionamiento de un sistema-D0L, con alfabeto $\{a,b\}$ y reglas de producción: $a \rightarrow ab$, $b \rightarrow a$.

Fractales e interpretación gráfica de las cadenas de símbolos

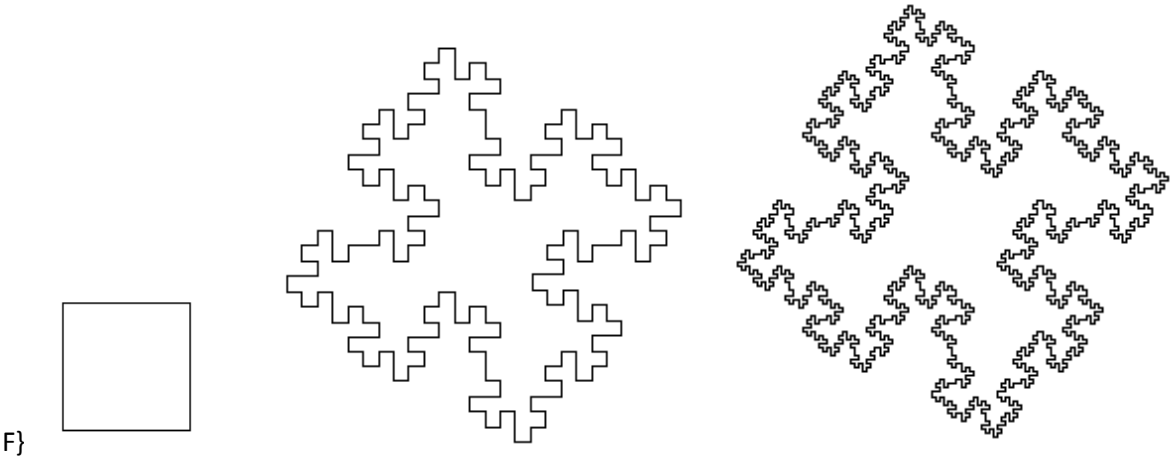
Los sistemas de Lindenmayer fueron concebidos como una teoría matemática del desarrollo biológico, por lo cual los aspectos geométricos estaban fuera del alcance de la teoría. Sin embargo, posteriormente se propusieron varias interpretaciones geométricas que transformaron a los sistemas-L en una herramienta versátil para la visualización y modelaje de curvas fractales y estructuras vegetales.

Muchos fractales (o al menos su aproximación finita) pueden considerarse como una secuencia finita de elementos primitivos (segmentos de línea). Para producir fractales, las cadenas producidas por los sistemas-L deben contener la información necesaria respecto a la geometría de la figura. En Prusinkiewics y Lindenmayer (1990) se describe una interpretación de las cadenas de símbolos basada en la geometría de *tortuga* (*turtle geometry*), la cual puede utilizarse para producir imágenes fractales.

Un estado de la tortuga se defina como una tripleta *(x, y, a),* donde las coordenadas Cartesians *(x,y)* representan la posición de la tortuga y el ángulo *a* ,es interpretado como la dirección a la que apunta la tortuga. Dado un tamaño de paso *d* y un incremento del ángulo *b*, la tortuga puede responder a los comandos indicados por los siguientes símbolos:

F	Mueve adelante un paso de longitud *d*. El estado de la tortuga cambia a *(x', y', a')*, donde *x'=x+d cos(a)* y *y'=y+d sen(a).* Se dibuja una línea entre los puntos *(x, y)* y *(x', y')*.
f	Mueve adelante un paso de longitud *d*, pero sin dibujar una línea. El estado de la tortuga cambia como se indica arriba.
+	Gira a la izquierda un incremento de ángulo *b*. El estado de la tortuga cambia a *(x, y, a+b)*.
+	Gira a la izquierda un incremento de ángulo *b*. El estado de la tortuga cambia a *(x, y, a+b)*.

Dados una cadena de caracteres v, el estado inicial de la tortuga *(xo, yo, ao)*, y valores para los parámetros *d* y *b*, la interpretación grafica de tortuga de la cadena v es la figura (conjunto de segmentos de línea) dibujados por la tortuga en respuesta a dicha cadena. Este método representa una manera rigurosa de generar figuras a partir de cadenas de caracteres. Puede aplicarse para interpretar cadenas generadas por sistemas-L. La Figura 6, representa cuatro aproximaciones a la curva conocida como la "isla cuadrática de Koch". Estas figuras se obtuvieron al interpretar la cadena de símbolos generada por el siguiente sistema-L: {w: F+F+F+F, p: F -> F+F-F-FF+F+F-



F}

Figura 6: Estas imágenes corresponden a las cadenas obtenidas al derivar el sistema-L {w: F+F+F+F, p: F -> F+F-F-FF+F+F-F}, con longitud de derivación *n* = 0, 1 y 2 respectivamente. El incremento de ángulo *b* es de 90º.

Sistemas-L con corchetes y modelos de arquitectura vegetal

Como se describe en la sección anterior, la tortuga interpreta una cadena de símbolos como una secuencia de segmentos de línea, que están conectados de la "cabeza a la cola". Dependiendo de los valores de longitud de segmento e incremento del ángulo, la figura resultante puede ser más o menos intrincada y replegada, pero permanece siendo una única línea.

Lindenmayer introdujo una notación para representar estructuras ramificadas utilizando cadenas con corchetes. La idea era describir formalmente las estructuras de ramificación encontradas en las plantas (desde algas a árboles terrestres) utilizando el marco de los sistemas-L. Posteriormente, se propuso una interpretación geométrica de las cadenas con corchetes, para modelar de manera realista plantas y estructuras vegetales. Por lo tanto, para representar estructuras ramificadas, el alfabeto de los sistemas-L es extendido con dos nuevos símbolos: '[' y ']' para delimitar una rama. Estos son interpretados de la manera siguiente:

[Guarda el estado de la Tortuga en una estructura tipo Pila.

] Recupera un estado de la Tortuga de la Pila y lo asigna como el estado actual de la misma.

La Figura 7 en la siguiente sección muestra un ejemplo de sistema-L con corchetes, donde se visualiza una estructura ramificada que asemeja a un arbusto.

Un modelo de la evolución de formas vegetales

Esta sección describe un modelo computacional que permite evolucionar artificialmente estructuras ramificadas bidimensionales (Ochoa, 1998). Estas estructuras evocan por su forma a los organismos vegetales naturales. El modelo puede enmarcarse dentro de la fascinante disciplina conocida como Vida Artificial, la cual es definida por Christopher G. Langton como "el estudio de sistemas hechos por el hombre que exhiben comportamientos característicos de los sistemas naturales vivos" (Langton, 1988). Un objetivo fundamental del área es crear y estudiar organismos artificiales que se asemejen a los organismos naturales.

Motivación y antecedentes

Dentro de los temas de estudio de la Vida Artificial se encuentra la simulación de la evolución biológica. Se han presentado modelos que realizan simulaciones de grandes poblaciones en un ambiente complejo a través de varias generaciones. Sin embargo, los organismos modelados pertenecen en su mayoría al reino animal. El interés se ha centrado en modelar comportamientos y no tanto la morfología física de estos organismos. Entre las líneas de investigación que se han interesado en el aspecto morfológico y el desarrollo de los organismos biológicos se destaca el trabajo realizado por el biólogo Richard Dawkins . En su libro *The Blind Watchmaker* (Dawkins, 1986) el autor demuestra el poder del Darwinismo con una simulación de la evolución de estructuras ramificadas en dos dimensiones. Estas estructuras, que Dawkins ha denominado *biomorfos*, son construidas a partir de un conjunto de parámetros genéticos. El usuario selecciona los biomorfos que sobreviven y se reproducen para formar cada nueva generación. Otros trabajos que pueden mencionarse, en este sentido son la "colección de animales" artificiales de P. Oppenheimer (Oppenheimer, 1988) y las criaturas virtuales de Karl Sims (Sims, 1994). Este último autor, ha enfocado su trabajo más en la perspectiva de la disciplina de computación gráfica. Describe cómo técnicas evolutivas de variación y selección pueden utilizarse para crear estructuras complejas, texturas y movimientos para el uso en computación gráfica y animación (Sims, Artificial Evolution for Computer Graphics, 1991).

Cabe mencionar también, dentro de las metodologías para estudiar la morfología de los organismos biológicos, al formalismo matemático conocido como los sistemas de *Lindenmayer* o *sistemas-L*, descritos en la sección anterior.

Un antecedente importante al modelo aquí descrito es el trabajo titulado *Computer-simulated Plant Evolution*, del biólogo evolucionista Karl Niklas (Niklas, 1985). En este artículo se describe un modelo computarizado que simula al desarrollo evolutivo de las primeras plantas terrestres. La simulación se fundamenta en hipótesis de la biología evolutiva, estas hipótesis son afirmaciones sobre qué factores han tenido la mayor influencia en la evolución de las plantas. Una de estas hipótesis es que la mayoría de las plantas pueden considerarse como soluciones estructurales a las restricciones

impuestas por el proceso bioquímico de la fotosíntesis. Puede predecirse entonces que las plantas con patrones de ramificación que recolecten la mayor cantidad de luz, serán las más exitosas. Sin embargo, para ser competidoras efectivas por la luz y el espacio, las plantas deben desempeñar ciertas otras tareas. En particular deben ser capaces de mantenerse erguidas, esto es, sostener el estrés mecánico comprometido en el crecimiento vertical. Una segunda hipótesis sería, entonces, que la evolución de las plantas fue impulsada por la necesidad de reconciliar la habilidad de recolectar luz con la habilidad de sustentar las estructuras verticales de ramificación. Una tercera hipótesis afirma que la evolución de las plantas es guiada por la magnitud de su éxito al reproducirse, siendo favorables los patrones de ramificación que permitan una mejor diseminación de las semillas.

En el modelo de Niklas, cada planta es representada a través de solamente tres características o factores "genéticos". En la terminología de la computación evolutiva, diríamos que los cromosomas en el modelo de Niklas se componen de tres parámetros numéricos. La simulación del desarrollo de una planta ocurre por etapas. Durante cada etapa, una planta sufre varias fases de alteración. Primero cada eje crece una pequeña distancia. Luego el modelo selecciona, según el primer factor - probabilidad de ramificación - , cuantos y cuales ejes van a ramificarse. En aquellos que se ramifican las direcciones de los nuevos ejes son determinadas por los otros dos factores genéticos. Uno es el ángulo de ramificación entre los nuevos ejes; y el otro el ángulo de rotación: ángulo entre dos puntos de ramificación sucesivos. Se considera que las plantas completan su desarrollo luego de 10 ciclos de ramificación. Al covariar los tres rasgos geométricos, que hemos denominado factores genéticos, se constituye un dominio morfológico teórico de las estructuras ramificadas. Este dominio puede visualizarse como un espacio tridimensional o cubo.

La simulación de una trayectoria evolutiva, se realiza de la siguiente manera. Inicialmente se determina un patrón de ramificación primitivo en la escala evolutiva (un patrón que se asemeje a los de las plantas más antiguas, de acuerdo a los registros fósiles). Luego, el modelo selecciona del conjunto de vecinos más cercanos a la planta (en el dominio morfológico), la especie más eficiente según las hipótesis evolutivas mencionadas. Este proceso es iterado hasta que el modelo identifique un conjunto de características morfológicas que resulte más eficiente que cualquiera los vecinos inmediatos en el espacio tridimensional. Con este modelo Niklas encontró trayectorias evolutivas que tienen bastante semejanza con las tendencias observadas en los registros fósiles.

El universo de formas en el dominio morfológico teórico de Niklas, a pesar de ser amplio, tiene sus limitaciones. En particular, nótese que los puntos de ramificación en las estructuras, pueden ser a lo sumo binarios. Los sistemas-L, por otro lado, constituyen una herramienta mucho más poderosa en cuanto al espacio de formas que pueden ser modeladas. Una ventaja adicional de los sistemas-L es que son más realistas desde el punto de vista biológico ya que evocan al proceso natural de la morfogénesis. Otra limitación del modelo de Niklas es que no simula los operadores genéticos naturales de mutación y recombinación o *crossover*. La exploración en el espacio se hace inspeccionando los vecinos inmediatos de un organismo según alteraciones discretas y prefijadas de sus factores genéticos. Tampoco contempla el concepto de una población de individuos, concepto que es fundamental en el estudio de la evolución tanto natural como simulada.

Estas últimas consideraciones sobre el trabajo de Niklas, así como la investigación en vida artificial y computación gráfica mencionadas en esta sección, inspiraron el diseño del modelo que se describe a continuación.

Descripción general

Más que sus comportamientos, lo interesante en las plantas es la gran diversidad y variabilidad de sus formas y estructuras. El modelo propuesto se enfoca en la morfología de los organismos vegetales, y en la posibilidad de generar artificialmente éstas formas reduciendo la participación humana en su diseño. Para este fin, y siguiendo los lineamientos de la Vida Artificial la metodología de trabajo es simular los procesos naturales de variación y selección natural que ocurren en los organismos biológicos. La representación genética o genotipo de los organismos se fundamenta en el mencionado formalismo denominado sistemas-L. Dos tipos de evolución artificial pueden realizarse con el modelo. El primer tipo es análogo a la simulación de la selección artificial que hacen; Richard Dawkins con sus biomorfos, y Karl Sims con sus diversos objetos y estructuras. Las estructuras ramificadas bidimensionales pueden ser modificadas interactivamente, utilizándose los operadores de variación genética, u operadores de reproducción, diseñados. Se deja al criterio estético del "reproductor" humano (el usuario del programa) la selección de los organismos más aptos que podrán permanecer en la población y reproducirse. El segundo tipo es una simulación de la evolución natural de las plantas. Para este fin se utiliza un algoritmo evolutivo, donde la función de evaluación o desempeño encuentra su justificación en las hipótesis de la biología evolutiva descritas por Niklas en su estudio. El algoritmo evolutivo constituye una poderosa herramienta de búsqueda y exploración en el espacio de estructuras representado por los sistemas-L.

La representación de los organismos

La escogencia de la representación de los organismos es un aspecto esencial en todo modelo de computación evolutiva y vida artificial. Cuando se discute sobre evolución artificial, es conveniente hacer la distinción entre el genotipo y el fenotipo de los organismos. En general, en los modelos de vida artificial y computación evolutiva, en concordancia con la Naturaleza, los operadores de variación (mutación, recombinación, inversión, etc.) actúan sobre el genotipo; y la selección o supervivencia del organismo está relacionada con su fenotipo.

El genotipo

En el modelo propuesto, el cromosoma que codifica a un organismo es un sistema-L determinista y libre de contexto, sistema-D0L descritos previamente, con una única regla de producción. A manera de ejemplo se muestran dos cromosomas:

$$F \rightarrow F[++F]F[-F][F],$$
$$F \rightarrow [FFF+F[-F-F]F[--F]]FF$$

En otras palabras, un cromosoma es un sistema de reglas de producción, donde el axioma es el mismo para todos los individuos, el símbolo 'F', y el conjunto de producciones se reduce a una única regla cuya longitud es variable. Para ser más precisos, el cromosoma codifica al sucesor de la única producción del sistema D0L, ya que el predecesor es siempre el símbolo 'F'. Así, los cromosomas antes mostrados son realmente:

$$F[++F]F[-F][F],\ [FFF+F[-F-F]F[--F]]FF.$$

El modelo manejara palabras de este lenguaje cuyas longitudes se encuentran en un rango prefijado. Se introducen parámetros que determinan las longitudes máxima y mínima aceptadas de los cromosomas.

El fenotipo

Como hemos mencionado, el fenotipo de un organismo viene dado por la estructura ramificada bidimensional que se produce luego de derivar el sistema D0L e interpretar geométricamente la cadena de caracteres producto de esta derivación. En este proceso de ir del genotipo al fenotipo intervienen también otros parámetros, a saber; la longitud de la derivación (*n*), el paso de la tortuga (*d*) y el incremento del ángulo de la tortuga (δ), que no forman parte del genotipo. Tenemos, entonces, que para un genotipo particular hay más de un fenotipo asociado.

La figura 7, muestra la secuencia de la derivación e interpretación geométrica de un sistema-D0L.

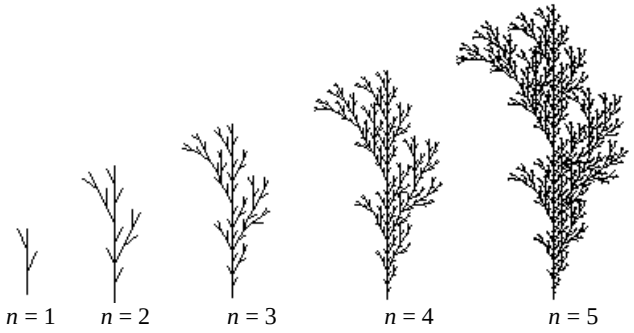


Figura 7: Visualización de la derivación de una estructura ramificada generada por el sistema-D0L: F, F → F[-F]F[+F][F]. El incremento del ángulo es δ = 27° , y n indica el número de pasos de la derivación.

La figura 8 muestra la interpretación geométrica de la derivación de longitud n = 4 del mismo sistema-D0L, para distintos valores del parámetro δ (incremento del ángulo).

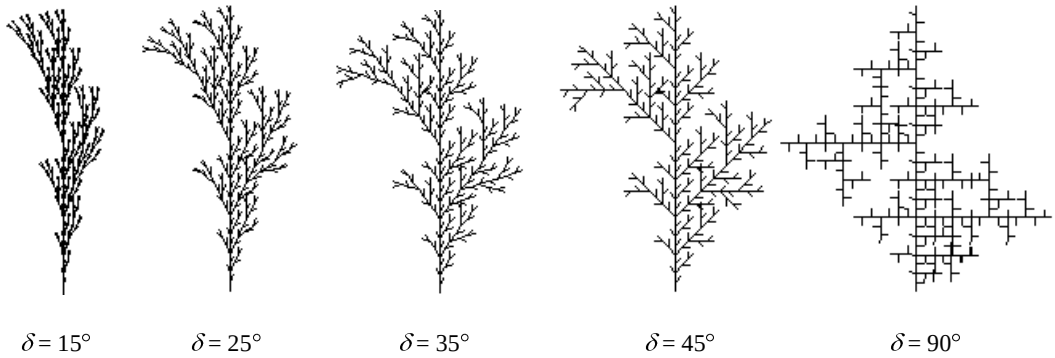


Figura 8: Interpretación geométrica del sistema D0L: F, F → F[-F]F[+F][F], para distintos valores del parámetro δ.

Operadores genéticos

Los sistemas-L, a diferencia de los cromosomas utilizados tradicionalmente en los algoritmos genéticos convencionales, son estructurados. Por esta razón deben diseñarse operadores genéticos especiales que produzcan descendencia conforme con su estructura sintáctica. En el paradigma denominado Programación Genética creado por John Koza (Koza, 1992), los cromosomas son programas en el lenguaje de programación denominado LISP. Una característica importante de LISP es que todos los programas tienen sólo una construcción sintáctica; la expresión simbólica o

expresión S (del inglés S-expression). Los programas en LISP son expresiones S, y una expresión S es, en efecto, el árbol de derivación del programa. La estructura sintáctica de los sistemas-L también puede visualizarse como árboles, por esta razón los operadores de reproducción diseñados están inspirados en los operadores de la programación genética. Tres operadores genéticos fueron diseñados, a saber, una recombinación o crossover y dos tipos de mutación. A continuación presentamos el diseño y visualización estos operadores.

Recombinación o "Crossover"

La operación de "crossover" o recombinación sexual genera variación en la población al producir nueva descendencia que se compone de partes tomadas de cada progenitor. Además permite que estrategias exitosas que hayan evolucionado separadamente en individuos distintos, puedan combinarse en un sólo individuo.

El operador de recombinación implementado está inspirado en la recombinación utilizada en Programación Genética (Koza, 1992). Los segmentos a intercambiar corresponden a bloques gramaticalmente correctos (con un numero balanceado de corchetes). De esta manera, los descendientes de la recombinación son individuos sintácticamente correctos. La Figura 9 ilustra dos ejemplos de recombinación. Resulta interesante observar como la morfología de los progenitores es combinada en la de su descendencia.

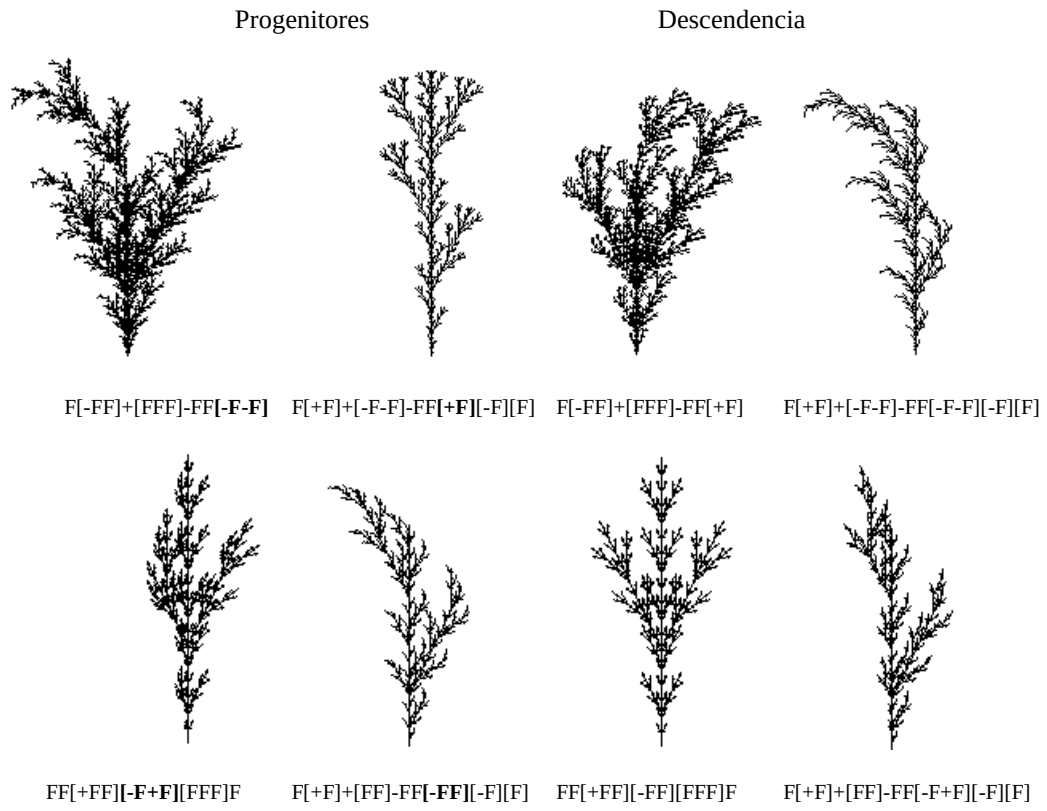


Figura.9: Visualización de progenitores y descendencia de dos "crossovers". Los puntos de cruce corresponden a los bloques resaltados en los progenitores. La longitud de la derivación es n = 4 y el incremento del ángulo δ = 25° para todos los organismos.

Mutación

La operación de mutación introduce cambios aleatorios en las estructuras de la población. La mutación es asexual en el sentido que opera sobre un solo cromosoma progenitor.

En el modelo, se diseñaron dos tipos de mutación. La distinción entre ellas tiene que ver con la porción o localidad del cromosoma que se ve afectada por la mutación.

- **Mutación con localidad de símbolo:** La operación comienza seleccionando aleatoriamente, con distribución de probabilidad uniforme, una posición del cromosoma en el conjunto {F,+,-} . El símbolo seleccionado es sustituido por una cadena de símbolos del sistema L, generada aleatoriamente, pero manteniendo la legalidad sintáctica.
- **Mutación con localidad de bloque:** La operación comienza seleccionando aleatoriamente, utilizando una distribución de probabilidad uniforme, no una posición sino un bloque del cromosoma. A continuación, el bloque escogido es sustituido por una cadena de símbolos del sistema L, generada aleatoriamente, pero manteniendo la legalidad sintáctica.

La figura 10 ilustra un ejemplo de cada tipo de mutación.

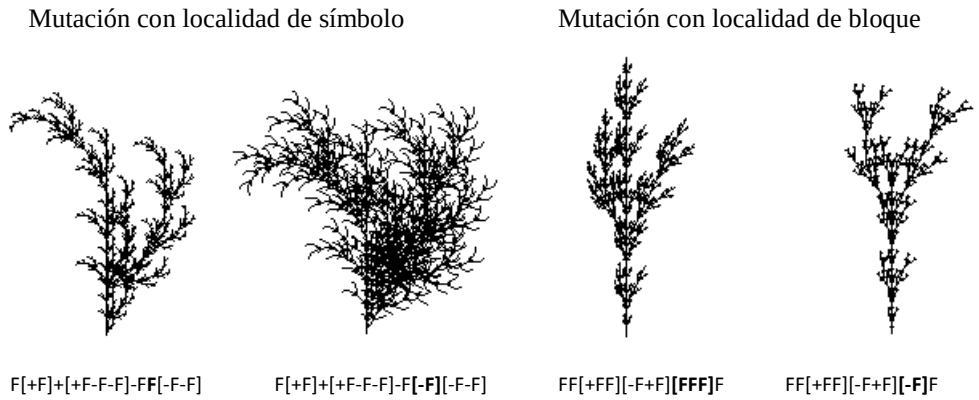


Figura 10: Visualización del organismo original y el resultante de la mutación. La localidad de la mutación aparece resaltada en el progenitor, y en el descendiente aparece resaltado el nuevo fragmento que la sustituye. El número de pasos de la derivación es n = 4 y el ángulo $\delta = 25°$ para los cuatro organismos.

La función de desempeño

Para diseñar la función que guiará la evolución de las estructuras vegetales de manera que éstas evoquen a las plantas naturales, deben plantearse hipótesis sobre cuáles fueron las presiones selectivas en la naturaleza que actuaron sobre las distintas variaciones menores que surgen entre los organismos progenitores y su descendencia. El trabajo titulado “Computer-simulated Plant Evolution” por Karl Niklas (Niklas, 1985), descrito anteriormente, considera que la mayoría de las plantas son soluciones estructurales a las restricciones impuestas por el proceso bioquímico de la fotosíntesis. Entonces, las plantas con patrones de ramificación que reciben una mayor cantidad de luz serán las más exitosas. Además , para competir efectivamente por la luz y el espacio, las plantas deben realizar otras funciones, tales como mantenerse erguidas y facilitar la diseminación de semillas o esporas.

El diseño de la función de evaluación se fundamenta en las hipótesis mencionadas anteriormente. Para modelar los distintos aspectos contenidos en estas hipótesis; a saber, (1) capacidad de absorción de la luz, (2) estabilidad de la estructura, y (3) mejor diseminación de semillas o esporas;

se estructuró la función de evaluación en cinco componentes que hemos denominado: (a) fototropismo positivo[1], (b) simetría, (c) capacidad de absorción de la luz, (d) cantidad de ramas de la estructura, y (e) estabilidad de los nodos de ramificación. Se desarrollaron técnicas matemáticas para cuantificar el aporte de cada una de estas componentes. Más adelante describiremos en mayor detalle cada una de ellas. Como vemos, hay una diferencia numérica (3 contra 5) entre los aspectos a modelar (*1*, *2*, *3*) y las componentes de la función de evaluación (*a*, *b*, *c*, *d* y *e*). Esto es porque los aspectos a modelar se corresponden con más de una componente de la función. En particular, el aspecto *1*, se modela con las componentes *a*, *b*, *c* y *d*; el *2* con las componentes *a*, *b* y *e*; y el *3* con las componentes *a*, *b* y *d*. Esta estructuración en componentes es, por supuesto, arbitraria y constituye una simplificación de la realidad. Obedece, además, a las características de la representación e implantación particulares.

Como hemos mencionado, la función de desempeño actúa sobre el fenotipo, esto es, la estructura ramificada bidimensional. Por tanto, antes de evaluar un sistema L este debe ser derivado e interpretado geométricamente según parámetros prefijados de longitud de la derivación, incremento del ángulo y paso de la tortuga. La función de evaluación no actúa directamente sobre la figura de la planta, lo hace sobre estructuras de datos que mantienen la información geométrica de la estructura, en otras palabras, no hay que graficar la planta para evaluarla. Cada componente de la función de adaptación es calculada por un procedimiento que recibe como parámetros los datos con la información geométrica particular, y devuelve un valor real en el rango [0,1] que indica la calidad de la estructura en el aspecto que el procedimiento cuantifica. Las estructuras de datos internas que mantienen la información geométrica de una "planta" o estructura ramificada son las siguientes; (1) las coordenadas (*x, y*) de cada vértice en un sistema de referencia cartesiano de dos dimensiones cuyo origen es el punto de "nacimiento" de la estructura; y (2) una representación del grafo de tipo árbol axial (secc. 3.5.1) que puede asociarse a la estructura ramificada. A continuación se describe someramente la cuantificación de cada componente.

a) Fototropismo positivo: Se observa el valor máximo de la coordenada *y* de la "planta", en el sistema de referencia cartesiano, favoreciéndose las estructuras ramificadas con un valor alto y positivo de la mencionada coordenada. Al mismo tiempo se desfavorecen las estructuras que contengan alguna coordenada *y* negativa.

b) Simetría: Se calcula la distribución del "peso" en la "planta". Donde el "peso" de un nodo o vértice de la estructura está en proporción directa a su distancia al eje de referencia vertical (*Y*), esto es, al valor absoluto de su coordenada *x*. Lo que se hace es acumular los pesos de los vértices, a la izquierda y a la derecha del eje vertical y comparar estas cantidades. Se favorecen las estructuras con una distribución del "peso" más balanceada.

c) Capacidad de absorción de la Luz: Se cuantifica la superficie de las hojas de la "planta" (aristas finales del grafo asociado a la estructura) que no reciben "sombra" de otras hojas. Considerando que el eje de luz principal tiene dirección vertical y sentido hacia el "suelo". En este caso se favorecen aquellas plantas que contienen una mayor superficie iluminada (sin sombra).

[1] Fototropismo es el movimiento de crecimiento de determinados elementos de un organismo fijo, provocado por la luz. Este movimiento puede ser positivo (dirigido hacia la fuente luminosa) o negativo. Por ejemplo el tallo presenta fototropismo positivo y la raíz fototropismo negativo.

d) Estabilidad de los puntos de ramificación: Este componente se agregó con la finalidad de simular el aspecto de la hipótesis referente a la rigidez de la estructura. La premisa es que un nodo del árbol con un *grado-saliente*[2] muy alto es inestable o poco rígido. El valor considerado como muy alto es parametrizable. Para calcular este componente, se cuantifica el número de nodos para cada *grado-saliente*, en el grafo dirigido asociado a la estructura ramificada. Se favorecen las "plantas" que posean un alto número de nodos "más estables", con *grado-saliente* bajo, y un bajo número de nodos "inestables", con *grado-saliente* alto.

e) Proporción de ramas de la "planta": Simplemente se cuenta el número de nodos del grafo asociado con *grado-saliente* mayor que uno. Pues hay una relación directa entre este número y el número de ramas de la estructura. Se favorecen las estructuras con un número alto de ramas.

Finalmente, a cada una de estas componentes se le asocia un peso o proporción dentro de la función de evaluación final (p_a, p_b, p_c, p_d y p_e). Es así, que la función de evaluación *E* normalizada se calcula según la fórmula mostrada en la Ecuación 1.

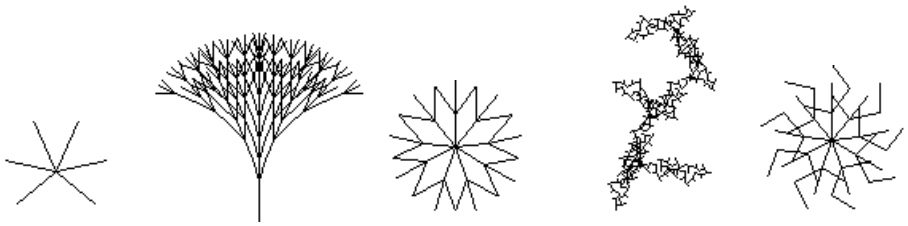
$$E(\,fenotipo) \;=\; \frac{a\,p_a + b\,p_b + c\,p_c + d\,p_d + e\,p_e}{p_a + p_b + p_c + p_d + p_e}$$

Ecuación 1: Fórmula de la función de evaluación o adaptación

Resultados de la simulación

Con los diseños finales de los operadores y de la función de adaptación, fueron realizadas numerosas evoluciones para distintos valores de los parámetros que controlan los pesos de cada componente de la función de adaptación (Ecuación 1). Los pesos para las componentes están en una escala del 0 al 100.

En los experimentos realizados, una población de organismos frecuentemente converge hacia la homogeneidad, esto es, uno o pocos individuos muy bien adaptados terminan dominando en la población. Sin embargo, corridas separadas de la evolución pueden producir estructuras ramificadas completamente diferentes, aún para el mismo conjunto de valores de parámetros de la función de adaptación. Por esta razón, varias evoluciones separadas, cada una de 40 a 70 generaciones, fueron realizadas. En los experimentos realizados, para la componente e), el rango para el grado-saliente de los nodos se mantuvo fijo en el intervalo [3,4]. Las estructuras más exitosas de cada evolución fueron inspeccionadas. Una selección de éstas se muestra en las figuras 11 a la 15. Debajo de las figuras aparece el valor arrojado por la función de adaptación en cada caso.



[2] El *grado-saliente* (del inglés *out-degree*) de un nodo o vértice en grafo dirigido, es el número de aristas que salen de dicho nodo

0.738538 0.541917 0.534641 0.423098 0.415396

Figura 11: Estructuras producidas con una función de adaptación con un peso de 50 para todas las componentes.

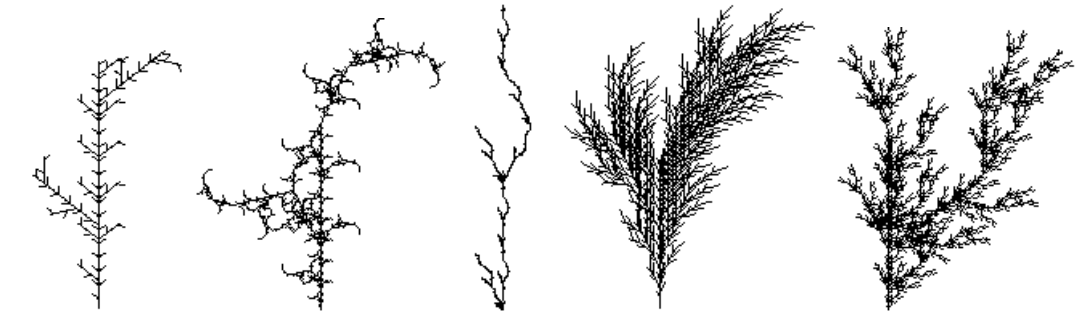


Figura 12: Estructuras producidas con una función de adaptación con pesos para las componentes; a) 100, b) 80, c) 20, d) 20 y e) 20.

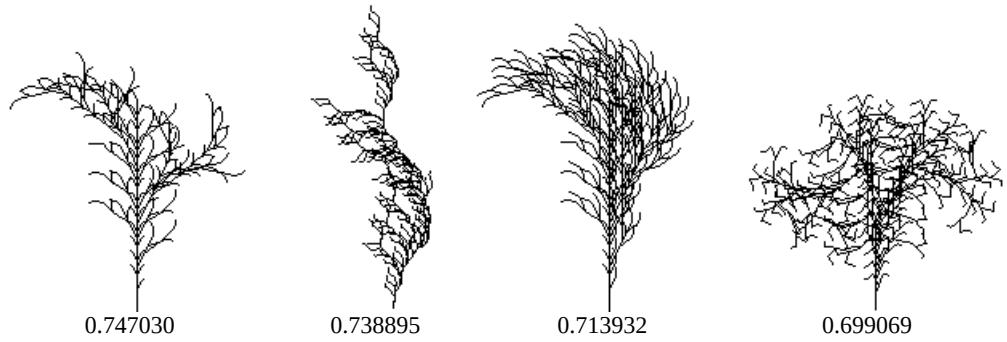


Figura 13: Estructuras producidas con una función de adaptación con pesos para las componentes; a) 100, b) 90, c) 40, d) 20 y e) 30.

Pudo observarse que, para obtener estructuras que evoquen a los organismos vegetales naturales, las componentes a) y b) (fototropismo positivo y simetría) deben ser considerablemente altos. Aproximadamente de 3 a 4 veces más altos que el resto de los pesos. Más aún, la componente a) debe ser ligeramente mayor que la b).

Respecto al tipo de evolución artificial interactiva que puede realizarse con el modelo, presentamos como resultado algunas de las figuras producidas.

El siguiente conjunto de estructuras que hemos denominado *Equinodermos*[3], (Figura 14) fue producida realizando interactivamente cruces y mutaciones, permutaciones, a partir de los organismos 3 y 5 de la figura 11. La selección de los organismos se realizó de acuerdo a criterios estéticos. El ordenamiento de las figuras no tiene ningún significado particular.

[3] Los *Equinodermos* constituyen un grupo de animales invertebrados marinos de piel espinosa. Se caracterizan, además, por presentar simetría radial en su fase adulta. Comprenden las estrellas de mar, erizos de mar, pepinos de mar y lirios de mar.

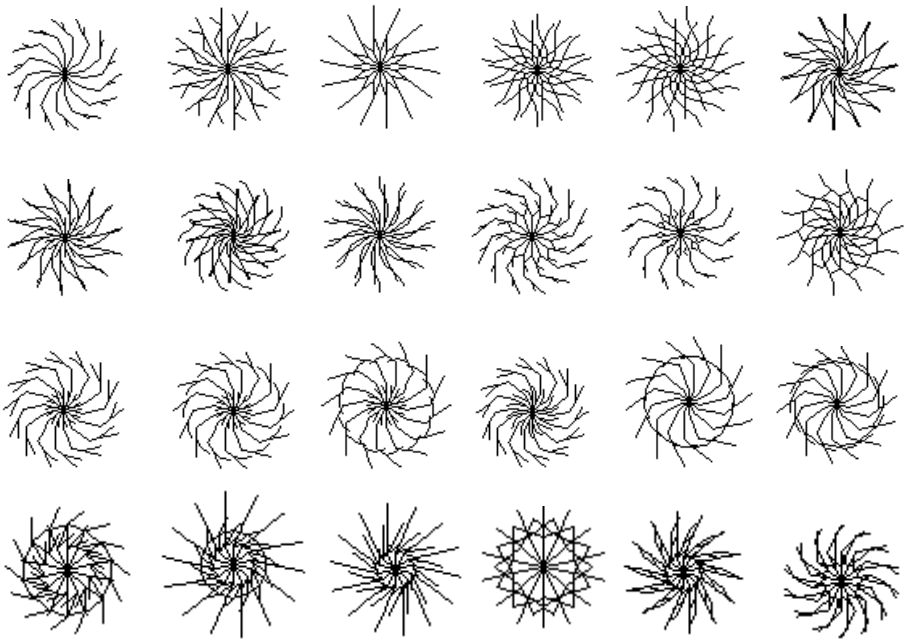


Figura 14: *Equinodermos*

Aunque la idea original fue explorar el espacio de estructuras que evocan a las formas naturales vegetales, pudimos observar la evolución de estructuras que sugieren a las formas de ciertos organismos animales, en particular los denominados *equinodermos* (Figura 14) y algunos insectos (Figura 15). También se obtuvieron figuras que evocan a ciertos objetos artificiales, y figuras novedosas que resultan llamativas (Figura 15).

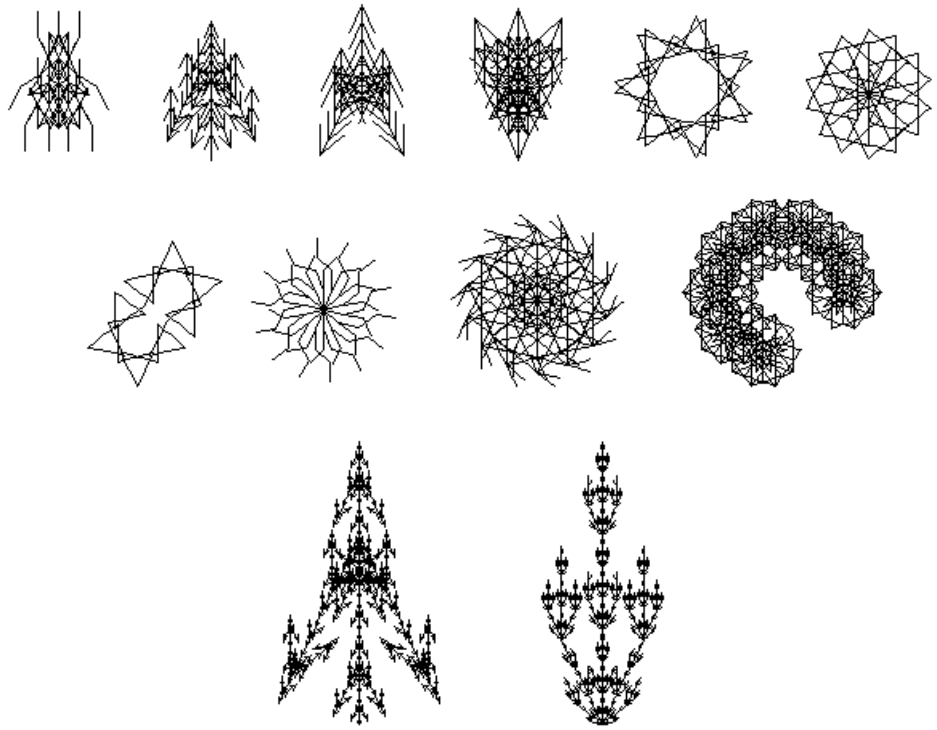


Figura 15: Figuras llamativas, que no evocan organismos vegetales, obtenidas con el modelo

Conclusiones

Este capítulo presenta una introducción general a los algoritmos evolutivos, seguida de una aplicación de estos en un modelo de simulación de la evolución de la estructura de las plantas. El modelo descrito puede enmarcarse dentro de la disciplina de la *vida artificial*, cuyo propósito es el de utilizar modelos computacionales como herramientas en el estudio de los fenómenos de la vida natural. El formalismo conocido como sistemas de Lindenmayer, *sistemas-L*, es también presentado en detalle como ejemplo del uso de herramientas matemáticas en el estudio de la morfología y desarrollo de los organismos naturales. La combinación de estos sistemas de Lindenmayer con los algoritmos evolutivos, es utilizada como ejemplo ilustrativo del poder de la computación evolutiva como herramienta para la exploración de un espacio de búsqueda y el diseño de estructuras novedosas. El modelo fue capaz de generar complicadas estructuras ramificadas bidimensionales sin requerir, por parte del usuario, de grandes especificaciones, esfuerzo de diseño o detalles algorítmicos. El modelo permitió constatar que el espacio de estructuras ramificadas, aún en dos dimensiones, es muy amplio y variado.

La computación evolutiva constituye una de las áreas más activas de la llamada "inteligencia computacional", que agrupa además las redes neuronales y los sistemas difusos (del Inglés *fuzzy systems*). Estas técnicas han logrado un gran número de aplicaciones exitosas en la industria y el comercio. Los algoritmos evolutivos se han utilizado también como herramientas de modelaje y resolución de problemas en las ciencias naturales.

A modo de cierre, quisiera argumentar que la computación en sus distintas sub-disciplinas, y en particular la llamada inteligencia computacional, tiene el potencial de aportar a la Humanidad una ayuda invalorable en el modelaje y solución de problemas difíciles y de importancia vital en temas como la salud, la producción de alimentos, el manejo de los recursos naturales, y los fenómenos climáticos. A modo de ejemplo, en la era post-genómica, la biología se transforma en una disciplina del estudio, manipulación e interpretación de información de índole digital. Es nuestra responsabilidad como computistas, el intentar tender puentes entre las disciplinas y sugerir la aplicación de nuestras tecnologías y algoritmos en la manipulación y comprensión de este gran volumen de datos, en vías de conformar soluciones integrales a estos grandes retos.

Bibliografía

Dawkins, R. (1986). *The Blind Watchmaker.* London: Harlow Longman.

Eiben, A. E., y Smith, J. .. (2003). *Introduction to Evolutionary Computing, .* Springer.

Fogel, L. J., Owens, A. J., y Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution.* New York: Wiley Publishing.

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

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor: University of Michigan Press.

Holland, J. H. (1986). Escaping brittleness: the possibilities of general purpose algorithms applied to parallel rule-based systems. En *Machine Learning, an Artificial Intelligence Approach* (págs. 593-623). San Mateo, CA: Morgan Kaufmann.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: MIT Press.

Langton, C. G. (1988). Artificial Life. *Artificial Life, SFI Studies in the Sciences of Complexity.* Addison-Wesley.

Lindenmayer, A. (1968). Mathematical models for cellular interaction in development. Parts I and II. *Journal of Theoretical Biology* , 280-315.

Niklas, K. J. (1985). Computer Simulated Plant Evolution. *Scientific American* .

Ochoa, G. (1998). On Genetic Algorithms and Lindenmayer Systems. *Parallel Problem Solving From Nature (PPSN V), Lecture Notes in Computer Science 1498,* (págs. 335-344). Berlin: Springer.

Oppenheimer, P. (1988). The Artificial Menagerie. *Artificial Life, SFI Studies in the Sciences of Complexity.* Addison-Wesley .

Prusinkiewics, P., y Lindenmayer, A. (1990). *The Algorithmic Beauty of Plants.* New York: Springer-Verla.

Rechenberg, I. (1973). *Evolution Strategy: Optimization of Technical Systems by Means of Biological Evolution.* Stuttgart: Fromman-Holzboog.

Sims, K. (1991). Artificial Evolution for Computer Graphics. *Proceedings of SIGGRAPH in Computer Graphics* (págs. 319-32). New York: ACM SIGGRAPH.

Sims, K. (1994). Evolving Virtual Creatures. *Proceedings of SIGGRAPH in Computer Graphics* (págs. 15-22). New York: ACM SIGGRAPH.