



**OBJETO VIRTUAL DE APRENDIZAJE PARA PROGRAMACIÓN POR EXPRESIÓN
GENÉTICA**

ORLANDO COSSIO MURILLO

**UNIVERSIDAD DEL VALLE
ESCUELA DE INGENIERÍA SISTEMAS Y COMPUTACIÓN
PROGRAMA ACADÉMICO DE INGENIERÍA DE SISTEMAS
SANTIAGO DE CALI
2010**

OBJETO VIRTUAL DE APRENDIZAJE PARA PROGRAMACIÓN POR EXPRESIÓN

ORLANDO COSSIO MURILLO

Código 0644126

E-mail: orlandoc@univalle.edu.co

**Trabajo de grado para optar al título de
Ingeniero de Sistemas**

Director trabajo de grado

ANGEL GARCIA BAÑOS

Ph D. en Ingeniería de Telecomunicación

**UNIVERSIDAD DEL VALLE
ESCUELA DE INGENIERÍA SISTEMAS Y COMPUTACIÓN
PROGRAMA ACADÉMICO DE INGENIERÍA DE SISTEMAS
SANTIAGO DE CALI
2010**

AGRADECIMIENTOS

A mi familia por estar siempre a mi lado apoyándome incondicionalmente, motivándome a terminar con esta meta que me he trazado.

A mi compañeros por los buenos momentos vividos y todas sus enseñanzas, a mis buenos profesores por guiarme en este camino y todos la familia de la EISC que fue mi querida casa durante este largo trayecto.

A la Universidad del Valle por darme la oportunidad de alcanzar el sueño de ser un profesional,

Índice de contenido

1. INTRODUCCIÓN.....	6
1.1 Planteamiento del Problema.....	7
1.1.1 En la educación.....	7
1.1.2 En la programación.....	7
1.2 Propuesta de solución.....	8
1.3 Justificación.....	9
1.4 Objetivos.....	10
1.4.1 Objetivo Principal.....	10
1.4.2 Objetivos específicos.....	10
1.4.3 Objetivos estratégicos.....	10
.....	10
.....	11
2. MARCO TEÓRICO.....	11
2.1 Algoritmos Genéticos.....	11
2.2 Programación Genética (GP).....	12
2.3 Programación por Expresión Génica (GEP).....	12
2.4 Objeto Virtual de Aprendizaje (OVA).....	17
2.5 Derivada Simbólica.....	17
2.6 Integración Simbólica.....	17
2.7 Programación Extrema.....	18
2.8 ADL.....	19
2.9 SCORM.....	19
2.10 IMS.....	20
2.11 LOM.....	20
2.12 Moodle.....	21
2.13 JavaFx.....	21
2.14 Conclusiones del capítulo.....	23
3. ESTADO DEL ARTE.....	24
3.1 En Programación por Expresión Genética.....	24
3.1.1 Gene Expression Programming .NET.....	24
3.1.2 PyGEP.....	24
3.1.3 GeneXproTools.....	24
.....	24
3.2 En Objetos Virtuales de Aprendizaje.....	24
3.3 Conclusiones del capítulo.....	26
4. DESCRIPCIÓN DE LA SOLUCIÓN.....	27
4.2 Historias de usuario.....	28
4.3 Casos de Uso.....	32
4.4 Casos de uso en formato extendido.....	33
4.5 Detalles de Implementación.....	36
4.6 Conclusiones del capítulo.....	41
5. PRUEBAS Y RESULTADOS.....	42
6. CONCLUSIONES.....	51

6.1 Conclusiones.....	51
6.2 Trabajos futuros.....	52
7. Referencias.....	53

Índice de ilustraciones

Ilustración 1: Algoritmo Genético, flujo de datos.....	10
Ilustración 2: Árbol de Expresiones (ET).....	11
Ilustración 3: Representación lineal y por ET de los genes en un cromosoma.....	12
Ilustración 4: Ciclo de desarrollo en programación extrema.....	15
Ilustración 5: Representación gráfica del estándar de SCORM.....	16
Ilustración 6: Representación gráfica de la plataforma Moodle.....	18
Ilustración 7: Diagrama de clase, modulo algoritmo GEP.....	23
Ilustración 8: Diagrama de clase.....	24
Ilustración 9: Representación gráfica de los casos de uso.....	25
Ilustración 10: Gráfico de la función deriva.....	33
Ilustración 11: Gráfico del calculo de la integral.....	34

Índice de tablas

Tabla 1: C.U. en formato extendido: Ingresar datos.....	30
Tabla 2: C.U en formato extendido: Realizar simulación.....	31
Tabla 3: C.U en formato extendido: Revisar ayudas.....	32

1. INTRODUCCIÓN

Vivimos en una época en que la tecnología ha avanzado mucho y lo sigue haciendo en forma exponencial gracias a la gran variedad de adelantos que se realizan cada día y estos a su vez son transmitidos en los diferentes medios de comunicación, los cuales nos acercan cada día más y rompen las barreras de las distancias, esto facilita compartir información para lograr hacer trabajos de manera colaborativa en muchas partes del mundo, eso hace que siga este crecimiento en el desarrollo de las tecnologías que cada vez, de manera más rápida, es aceptada por gran parte de la población. En los países en vías de desarrollo especialmente los países de centro y sur América donde existen diferencias grandes en las clases sociales, la mayoría de la población pertenece a la clase de menor poder adquisitivo.

Por otra parte, actualmente la computación hace parte importante de nuestra vida diaria ya que gracias a ella y los aparatos electrónicos que manejamos con ella, podemos hacer nuestra vida mucho más sencilla a la hora de realizar distintas tareas o actividades. Para que esto siempre sea posible los desarrolladores y los investigadores de la ciencias de la computación se deben enfrentar cada día nuevos retos, debido a la complejidad de algunos sistemas, por esto muchas personas especializadas en las ciencias de la computación o la informática deben resolver muchos problemas de todo tipo para hacer que los sistemas sean cada vez mejores.

En la ciencias de la computación siempre se están investigando mejores formas de resolver problemas y creando nuevos lenguajes de programación que no tengan las desventajas que presentan otros. Sin embargo la forma de programar o de hacer algoritmos es similar en la mayoría de los casos, esto hace que las soluciones a los problemas sean similares en los diferentes lenguajes de programación (al menos en los lenguajes imperativos) ya que el algoritmo subyacente es el mismo independientemente del lenguaje empleado.

A continuación se presentarán seis capítulos: Marco Teórico, en el que se abordarán los conceptos necesarios para entender el planteamiento de la solución y el desarrollo del proyecto; 3. Estado del Arte, en el cual se describirán algunas herramientas y trabajos realizados en programación por expresión genética y los avances de los objetos virtuales de aprendizaje, los cuales serán explicados y se permitirá entender la aplicación que se va a realizar; 4. Descripción de la Solución, mediante la cual se explicará la manera como se afrontó el problema, como se realizó que clases se emplearon; Pruebas y Resultados, en el que se verán las pruebas que se realizaron a la aplicación implementada y los resultados que se obtuvieron de dichas pruebas; Conclusiones y Referencias.

1.1 Planteamiento del Problema

Para explicar mejor el problema se ha dividido en dos campos:

1.1.1 En la educación

En la educación, en especial en las instituciones académicas publicas (enseñanza donde un profesor transmite los conocimientos a sus estudiantes por medio de clases presenciales), la herramienta de uso más común son los libros, los cuales son poco interactivos y muchas veces los profesores no cuentan con otras herramientas que los puedan ayudar en el proceso de la enseñanza de los diferentes contenidos de una materia.

Por otra parte las personas que no pueden acceder a la educación formal necesitan también de herramientas para poder profundizar en algunos conocimientos de su agrado sin la necesidad de pagar a un tutor particular. De igual forma existe la necesidad de contar con mas herramientas que ayuden al proceso de la educación.

Particularmente en la materia de computación evolutiva donde se puede entender los nuevos conceptos de programación con herramientas que ayuden a su comprensión por medio de casos específicos.

1.1.2 En la programación

En la actualidad existen muchas técnicas de programación en distintos lenguajes que son utilizados para resolver variados tipos de problemas. Aunque se pueden resolver muchos de esos, existen otros muy difíciles de resolver en los cuales hay que recurrir a métodos no tradicionales de programación. En este caso existe la necesidad de conocer otros métodos de programación que sirva para encontrar soluciones a los problemas que no podemos resolver con los métodos tradicionales

1.2 Propuesta de solución

Los OVAs (Objetos Virtuales de Aprendizaje) son una herramienta vital para el desarrollo de conocimiento en nuestras comunidades ya que solo el 27.7% de toda la población nacional de 18 a 24 años asisten a un establecimiento educativo formal según cifras del DANE en el 2005, la población restante necesita mas herramientas para poder acceder a los diferentes unidades de información de forma interactiva. Los OVAs pueden ayudar a la solución al problema de la educación en nuestro país sirviendo como herramienta educativa para las personas menos favorecidas y que puedan acceder a una conexión a internet.

La *programación genética* trabaja con un paradigma de programación diferente el cual rompe con el estilo de resolución de problemas paso a paso típico de los lenguajes imperativos. Para hallar la solución a un problema sólo necesita saber algunos datos iniciales. Con esa información simula procesos evolutivos hasta que encuentra la solución o una aproximación a la solución. Esta fue la razón por se escogió este proyecto “Objeto Virtual de Aprendizaje para Programación Genética por Expresión” ya que será una herramienta que ayudará a reforzar la comprensión de este proceso a los estudiantes que deseen profundizar en el paradigma de la *programación genética en especial a los estudiantes de la asignatura computación evolutiva*.

El propósito de este proyecto es crear una aplicación que permita ver el uso de la programación por expresión genética en casos específicos y la solución de estos problemas. El usuario podrá ver gráficamente la solución de problemas numéricos como la interpolación simbólica, predicción de secuencias, la integral simbólica y la derivada simbólica.

1.3 Justificación

La computación evolutiva es una rama de la informática que también hace parte de la inteligencia artificial, presenta soluciones a problemas de todo tipo, basándose en la idea darwiniana de la evolución por selección natural. El tema es bastante distinto a las técnicas de programación habituales, por lo que se necesitan herramientas didácticas que expliquen de forma interactiva como funciona, para poder masificar, llevar de una manera clara y sencilla el entendimiento de la computación evolutiva a los estudiantes, profesionales e interesados. Se proporcionará una herramienta de fácil acceso para su enseñanza, en un ambiente visual que muestra la forma como funciona la programación evolutiva, y como se abordan los problemas y sus soluciones.

Se busca con este proyecto incentivar el interés por la programación por expresión genética y por la asignatura de computación evolutiva. También se busca que se continúe el desarrollo de aplicaciones que permitan fortalecer los procesos educativos contando con herramientas de este tipo para que haya una mayor comprensión del conocimiento en un tema de estudio.

Este proyecto quedará disponible en la pagina del profesor de computación evolutiva para el uso de los estudiantes de la asignatura computación evolutiva, para que reforzar lo aprendido en el tema de computación por expresión genética vista en clase.

1.4 Objetivos

1.4.1 Objetivo Principal

- Desarrollar un programa interactivo que ayude en el aprendizaje del tema "programación genética por expresión "

1.4.2 Objetivos específicos

- Definir que tipo de estándar de objeto virtual se va a utilizar.
- Decidir que partes del proceso irán en el cliente y cuales en el servidor.
- Desarrollar la interface gráfica , amigable con el usuario.
- Desarrollar el algoritmo de programación genética y aplicarlo a la resolución de los siguientes problemas:
 - Interpolación simbólica.
 - Predicción de secuencias numéricas.
 - Integrales y derivadas simbólicas.
- Implementar ayudas interactivas .

1.4.3 Objetivos estratégicos

- Mejorar los aspectos pedagógicos de la asignatura Computación Evolutiva.
- Mejorar la visibilidad del grupo EVA (el código fuente quedará bajo licencia GPL).

2. MARCO TEÓRICO

2.1 Algoritmos Genéticos

Los algoritmo genético fue ideado por John Holland en 1969 – 1975, es un algoritmo de búsqueda de óptimos, muy robusto, el cual se basa en el concepto de la evolución. Los algoritmos genéticos sirven principalmente para resolver problemas numéricos (con ciertas modificaciones, también para combinatorios). Es de búsqueda orientada, explorador/explotador, con soluciones múltiples y para problemas generales.

Los algoritmos genéticos se pueden aplicar en muchos casos como en la minimización de funciones multidimensionales, el descubrimiento de estrategias óptimas en juegos, en la clasificación entre otros.

Para diseñar de un algoritmo genético se debe definir un cromosoma (secuencia de símbolos), cada gen del cromosoma (posición) almacena un símbolo y cada gen puede tener varios alelos. Un cromosoma debe representar una posible solución al problema dado, la posición de cada gen debe definir alguna característica relevante de la solución.

Luego definir una función de evaluación que, aplicada a un cromosoma, nos diga lo buena que es la solución representada por él, definir la manera de normalizar la función de evaluación para conseguir la función de aptitud (fitness), definir los operadores de reproducción y también definir la forma en que los nuevos cromosomas reemplazarán a los viejos, en la población de N cromosomas. [AGBGA]

Diagrama de flujo de datos:

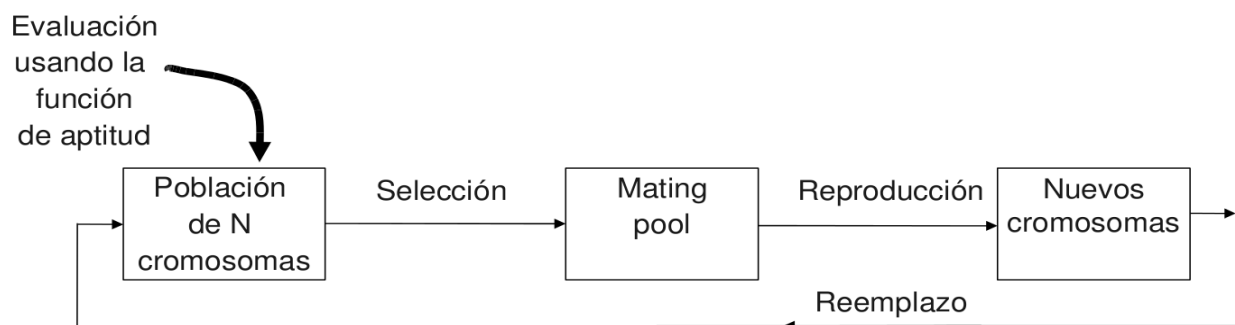


Ilustración 1: Algoritmo Genético, flujo de datos

2.2 Programación Genética (GP)

La programación genética (GP por las siglas del inglés Genetic Programming) es una metodología automatizada inspirada por la evolución biológica para encontrar los

programas informáticos que mejor realicen una tarea definida por el usuario. Es por esto una técnica de aprendizaje de máquinas particular que utiliza un algoritmo evolutivo para optimizar una población de programas informáticos. Los primeros en informar sobre los experimentos con GP fueron Stephen F. Smith (1980) y Michael L. Cramer (1985), tal y como se describió al famoso libro *Genetic Programming: On the Programming of Computers by Means of Natural Selection* ("Programación genética: Sobre la programación de computadoras mediante la selección natural") de John Koza (1992) [LVIAR].

Los programas de ordenador en GP pueden ser escritos en una variedad de lenguajes de programación. En las primeras (y tradicionales) implementaciones de GP, las instrucciones de programa y los valores de los datos estaban organizados en estructuras en árbol, favoreciendo la utilización de lenguajes que de forma natural trataban esta estructura (un ejemplo importante que innovó Koza es Lisp)[LVIAR].

2.3 Programación por Expresión Génica (GEP)

La programación por expresión génica (GEP) como lo son los algoritmos genéticos (AG) y la programación genética (GP), es un algoritmo genético, ya que utiliza una población de los individuos, los selecciona de acuerdo a la aptitud, e introduce la variación genética con uno o más operadores genéticos. Las diferencias fundamentales entre los tres algoritmos reside en la naturaleza de los individuos: en los algoritmos genéticos los individuos son cadenas lineales comúnmente de longitud fija (los cromosomas), en la GP de los individuos son entidades no lineal de diferentes tamaños y formas (árboles de análisis); y en la GEP los individuos se codifican como cadenas lineales de longitud fija (el genoma o cromosomas) que están después expresados como entidades no lineal de diferentes tamaños y formas (es decir, representaciones en diagramas simples o árboles de expresión)[CFR].

La organización estructural de los genes en GEP es mejor entendida en términos de marcos de lectura abierta (ORFs en inglés -Open Reading Frames-) En biología un ORF comienza cuando inicia el codon, continua con el aminoácido y termina cuando termina el codon, la diferencia con GEP es que no tiene que terminar cuando termina el codon, lo puede hacer antes, una cosa en común es que a través del codon en el gen de GEP también existen regiones sin codificar, en GEP serían datos que no se incluyen en la solución [AGBPGE].

En GEP se diferencia el genotipo del fenotipo. En AGs y GPs no se hace esa diferenciación (funcionan como los primitivos ARN). En GEP se mantiene una población de cromosomas, compuestos por genes de tamaño fijo codificados como cadenas de símbolos. Los cromosomas se reproducen como en GA (cruce, mutación). Luego cada gen se convierte a un ET (*expression tree*, es decir, árbol de expresión), que consiste en un programa de computador. [AGBPGE]

Ejemplo se considera la siguiente expresión algebraica:

$$\sqrt{(a+b) \times (c-d)},$$

La cual se representa en el siguiente árbol de expresión :

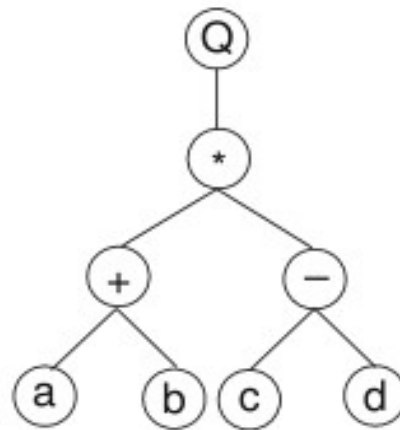


Ilustración 2: Árbol de Expresión (ET)

Donde Q representa la función raíz cuadrada. Este tipo de representación del diagrama es, de hecho, el fenotipo de GEP, siendo el genotipo fácilmente deducible del fenotipo de la siguiente manera:

1234567
Q*+-abcd

Esta cadena es leída en el árbol (ET) de izquierda a derecha y de arriba hacia abajo, la expresión es un ORF que comienza en “Q “ (en la posición 0) y termina en “d” (en la posición 7). Ver las ventajas en la estructura GEP ORFs es difícil excepto quizás por su simpleza o elegancia, Sin embargo, lo analizamos en el contexto de gen, las ventajas de su representación son obvias.

En este cuadro se ve el genotipo o contenido del genoma el cual representa la codificación del fenotipo mostrando cada una de sus posiciones. [CFR]

En GEP un gen no es un símbolo, sino una secuencia de símbolos de longitud fija. Lo que usualmente ocurre es que quedan símbolos no usados (no expresados) en el gen. De

esta manera, cualquier secuencia de símbolos en un gen da lugar a un ET sintácticamente correcto.[AGBPGE] Por ejemplo:

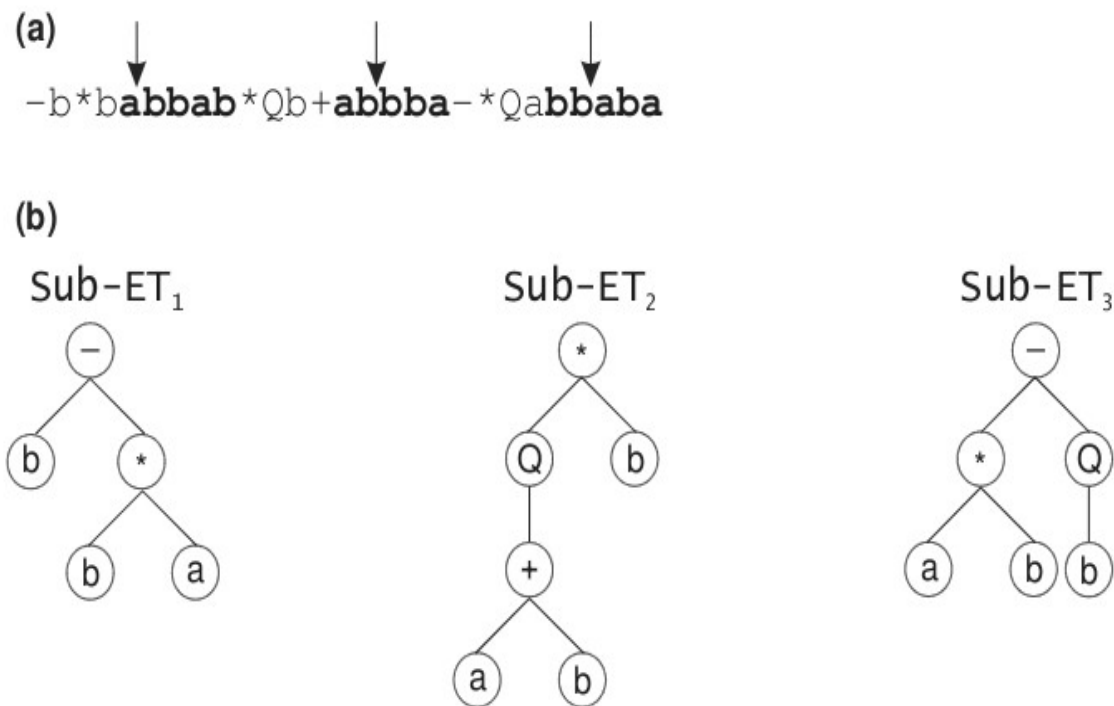


Ilustración 3: Representación lineal y por ET de los genes en un cromosoma

En lo gráficos anteriores se puede ver la representación en árboles de expresiones (ET) de tres genes en un cromosoma y como quedan símbolos no usados.

El gen en GEP está compuesto por una cabeza y una cola. La cabeza contiene la representación de ambos símbolos funciones y terminales, la cola solo contiene símbolos terminales. A pesar de su longitud fijada el gen tiene un potencial para codificar un árbol de expresión de diferentes tamaños o formas[CFR].

En la naturaleza, el fenotipo tiene varios niveles de complejidad, los organismos más complejos tienen células, proteínas, ribosomas y así sucesivamente, son también producto de la expresión y todos ellos están al final codificados en el cromosoma. En todo caso la expresión de la información genética comienza con una transcripción (la síntesis de ARN) y procesos con traslación (la síntesis de proteína)[CFR].

La función de aptitud es una de las aplicaciones más importantes de GEP, donde el objetivo es encontrar una expresión que realiza bien el cálculo de la aptitud para todos los casos para descubrir una buena solución[CFR].

La selección se hace de acuerdo a la función de aptitud y en la mayoría de los caso por muestreo por ruleta (con o sin el elitismo) aunque también se puede hacer por torneo (con o sin el elitismo) u otras técnicas deterministas[CFR].

La reproducción un cromosoma puede ser escogido por ninguno o por varios operadores genéticos para producir variación en la población, esto distingue a GEP de GP donde una cromosoma nunca es modificado por mas de un operador en un tiempo. En consecuencia, en GEP, la acumulación de varios operadores genéticos durante el proceso de reproducción da como resultado descendientes muy diferentes de sus padres[CFR].

Operadores genéticos en GEP:

Replicación: Es uno de los menos interesantes ya que no contribuye en mucho a la diversificación de la población ya que los cromosomas son copiados fielmente a la siguiente generación[CFR].

Mutación: Las mutaciones pueden ocurrir en cualquier parte del cromosoma, sin embargo la organización de la estructura de quedar intacta, en la cabeza los símbolos pueden variar a funciones o terminales pero en la cola solo pueden varias a terminales, esta estructura en mantenida para que los nuevos cromosomas conserven las reglas[CFR].

Transposición: Los fragmentos de un cromosoma pueden ser activados y movidos de un lugar a otro en el cromosoma. En GEP hay tres tipos de elementos transponibles. (1) Fragmentos cortos con una función o un terminal en la primera posición que transpone la cabeza de los genes excepto la de la raíz (IS Elements). (2) Fragmentos cortos con una función en la primera posición que traspone la raíz de los genes (RIS Elements). (3) Todo el gen transpone al inicio del cromosoma[CFR].

Recombinación: En GEP hay tres tipos de recombinación: un-punto, dos-puntos y recombinación gen. Durante la recombinación un-punto los cromosomas se cruzan un punto elegido al azar para formar hija a partir de dos cromosomas, En recombinacion dos-puntos los cromosomas se aparean y los dos puntos de recombinación se eligen al azar. El material entre los puntos de recombinación es posteriormente ex-cambiado entre los dos cromosomas, formando dos nuevos cromosomas hijas. En gen recombinación, el gen completo es cambiado durante el cruce. El cambio de genes es escogido al azar y ocupa la misma posición que en los cromosomas padres. El nuevo gen creado tiene genes de ambos padres[CFR].

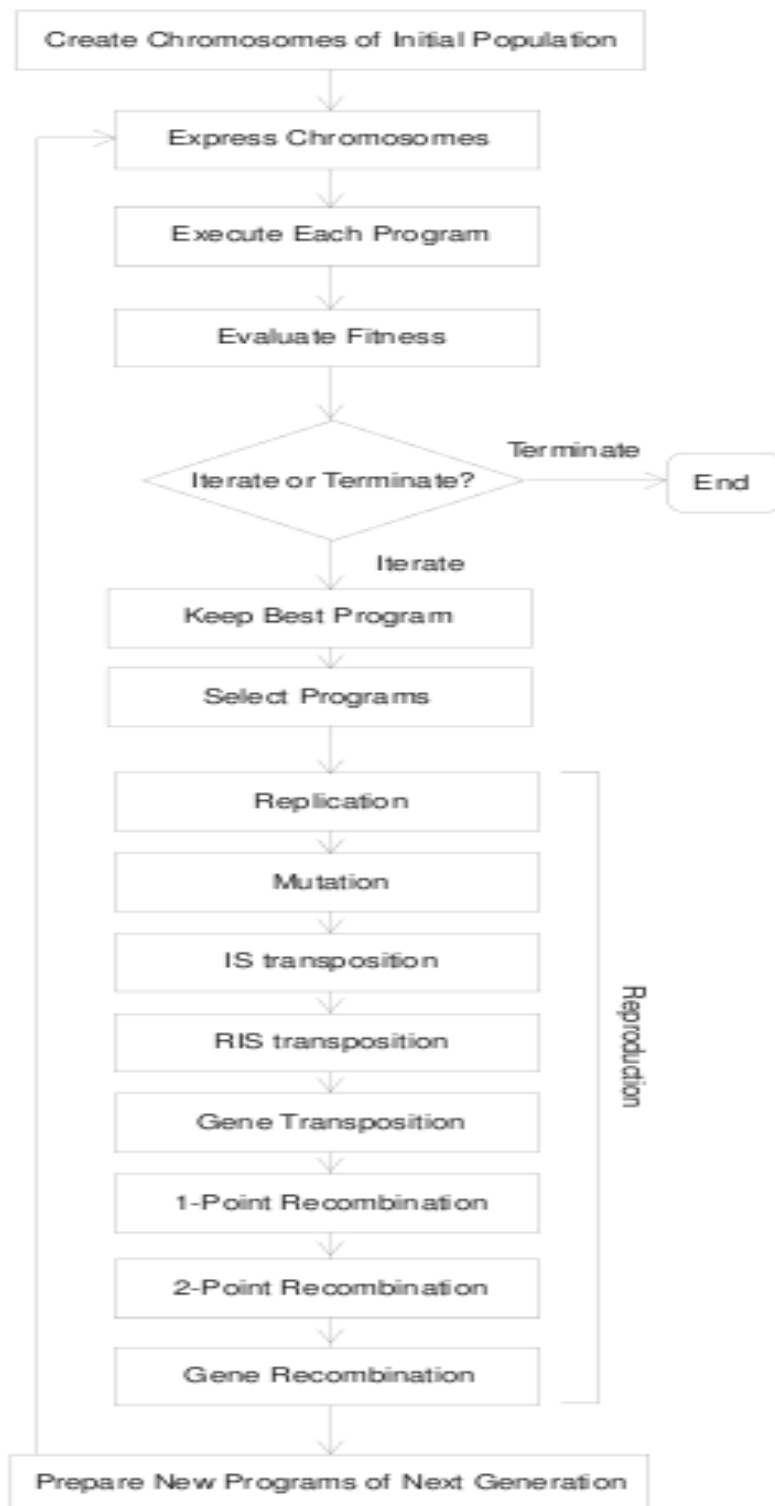


Ilustración 1: The flowchart of a gene expression algorithm

La ilustración anterior muestra un diagrama de flujo del algoritmo de programación por expresión genética, donde se muestran los operadores genéticos anteriormente explicados

2.4 Objeto Virtual de Aprendizaje (OVA)

El término “objeto de aprendizaje” generalmente se aplica a materiales educativos diseñados y creados en pequeñas unidades con el propósito de maximizar el número de situaciones de aprendizaje en las cuales puedan ser utilizados. Es un concepto reciente, que surge en los años 90 (learning objects), aunque hay muchos precedentes en software didáctico de diversas funcionalidades [CFDAE].

Se denominan Objetos Virtuales de Aprendizaje (OVA), cuando corresponden a archivos o unidades digitales de información, dispuestos con la intención de ser utilizados en diferentes propuestas y contextos pedagógicos. Se trata de archivos digitales o elementos con cierto nivel de interactividad e independencia, que podrán ser utilizados o ensamblados, sin modificación previa, en diferentes situaciones de enseñanza-aprendizaje. [CFDAE]

Los Objetos Virtuales de Aprendizaje (OVA) son ayudas pedagógicas que sirven como una herramienta útil en el proceso de enseñanza de cualquier asignatura ya que proporcionan ayudas interactivas y de usabilidad para los estudiantes, el uso de los objetos virtuales de aprendizaje puede ser asincrónico o sincrónico con el contenido de una asignatura debido a que el estudiante puede realizar procesos autónomos de aprendizaje que sirvan como refuerzo a lo visto en una asignatura. Además los Objetos Virtuales de Aprendizaje presentan ayudas de multimedia que explotan el factor visual lo cual hace más fácil los procesos de aprendizaje.

2.5 Derivada Simbólica

Una expresión formada por funciones elementales (que admiten derivada formada por funciones elementales) permite establecer un conjunto de reglas llamadas las “*reglas de derivación*” que trivializan el proceso de derivar una expresión dada. Basta seguir cuidadosamente todos y cada uno de los pasos que determinan las reglas. [JJDS]

Un método para poder encontrar la derivada simbólica en computación se debe construir un algoritmo reciba varios puntos de la función y a partir de estos se deben derivar en cada punto de la función y a partir de eso puntos construir la expresión matemática que más se aproxima a esos puntos.

2.6 Integración Simbólica

En muchos problemas de matemáticas, física, e ingeniería en los que participa la integración es deseable tener una fórmula explícita para la integral. Con esta finalidad, a lo largo de los años se han ido publicando extensas tablas de integrales. Con el desarrollo de los ordenadores, muchos profesionales, educadores y estudiantes han recurrido a los sistemas de cálculo algebraico por ordenador, que han sido diseñados específicamente

para desarrollar tareas tediosas o difíciles, entre las cuales se encuentra la integración. La integración simbólica presenta un reto especial en el desarrollo de este tipo de sistemas.

Una dificultad matemática importante de la integración simbólica es que, en muchos casos, no existe ninguna fórmula cerrada para la primitiva de una función. Por ejemplo, se

sabe que las primitivas de las funciones: $e^{(x^2)}$, x^x y $\frac{\sin(x)}{x}$

no se pueden expresar con una fórmula cerrada en las que participen sólo funciones racionales, exponenciales, logarítmicas, trigonométricas, inversas de las funciones trigonométricas, y las operaciones de suma, multiplicación y composición. Es decir, ninguna de estas tres funciones dadas es integrable con funciones elementales. La teoría de Galois diferencial proporciona criterios generales para determinar cuándo la primitiva de una función elemental es a su vez elemental. Por desgracia, resulta que las funciones con expresiones cerradas para sus primitivas son la excepción en vez de ser la regla. En consecuencia, los sistemas de cálculo algebraico, no pueden tener la seguridad de poder encontrar una primitiva para una función elemental cualquiera construida de forma aleatoria. En el lado positivo, si se fijan de antemano los "bloques constructivos" de las primitivas, aún es posible decidir si se puede expresar la primitiva de una función dada empleando estos bloques y las operaciones de multiplicación y composición, y hallar la respuesta simbólica en el caso de que exista. [WKIS]

2.7 Programación Extrema

La programación extrema es una metodología de desarrollo ligera (o ágil) basada en una serie de valores y de buenas prácticas que persigue el objetivo de aumentar la productividad a la hora de desarrollar programas.

Este modelo de programación se basa en una serie de metodologías de desarrollo de software en la que se da prioridad a los trabajos que dan un resultado directo y que reducen la formalismos que hay alrededor de la programación.

Una de las características principales de este método de programación, es que sus ingredientes son conocidos desde el principio de la informática. Los autores de XP han seleccionado aquellos que han considerado mejores y han profundizado en sus relaciones y en cómo se refuerzan los unos con los otros. El resultado de esta selección ha sido esta metodología eficiente y compacta. Por esto, aunque no está basada en principios nuevos, sí que el resultado es una nueva manera de ver el desarrollo de software.

El objetivo que se perseguía en el momento de crear esta metodología era la búsqueda de un método que hiciera que los desarrollos fueran más sencillos aplicando el sentido común.

Extreme Programming Project

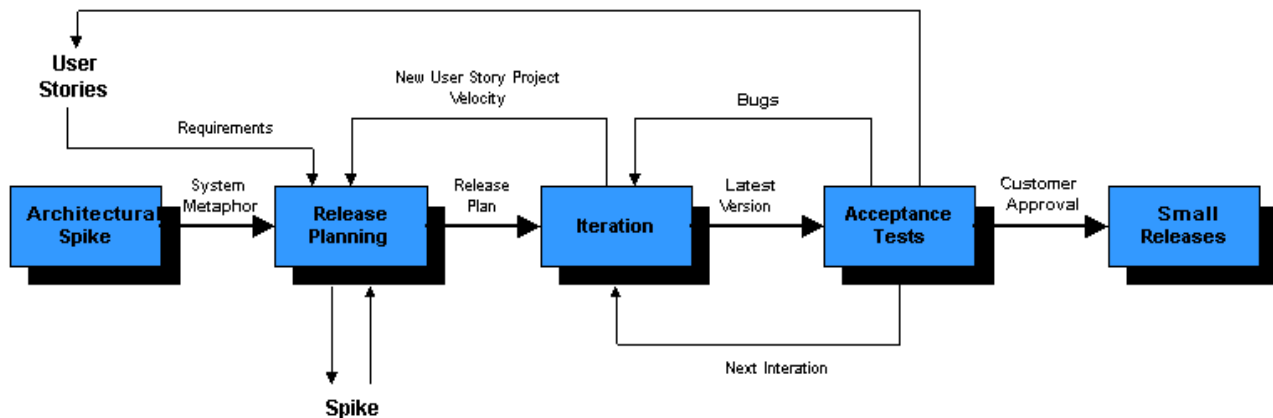


Ilustración 4: Ciclo de desarrollo en programación extrema

La ilustración 4, es un diagrama del ciclo del desarrollo de un proyecto con programación extrema, y como en esta metodología fluyen los procesos hasta pasar la etapa de aceptación del aplicativo.

2.8 ADL

ADL (Advanced Distributed Learning) es una iniciativa del Departamento de Defensa (DoD) de los Estados Unidos para implementar y desarrollar herramientas y tecnologías de aprendizaje.

Fue creada con la apoyo de la Oficina del Subsecretario de Defensa para Personal y Preparación (Office of the Under Secretary of Defense for Personnel and Readiness. OUSD P&R) para potenciar el uso de las tecnologías de la información para modernizar el aprendizaje estructurado.

La ADL es la impulsora del estándar de e-learning internacional SCORM. [WKADL]

2.9 SCORM

SCORM (del inglés *Sharable Content Object Reference Model*) es una especificación que permite crear objetos pedagógicos estructurados. Los sistemas de gestión de contenidos en web originales usaban formatos propietarios para los contenidos que distribuían. Como resultado, no era posible el intercambio de tales contenidos. Con SCORM se hace posible el crear contenidos que puedan importarse dentro de sistemas de gestión de aprendizaje diferentes, siempre que estos soporten la norma SCORM.

Los principales requerimientos que el modelo SCORM trata de satisfacer son:

- **Accesibilidad:** capacidad de acceder a los componentes de enseñanza desde un sitio distante a través de las tecnologías web, así como distribuirlos a otros sitios.
- **Adaptabilidad:** capacidad de personalizar la formación en función de las necesidades de las personas y organizaciones.

- Durabilidad:** capacidad de resistir a la evolución de la tecnología sin necesitar una reconcepción, una reconfiguración o una reescritura del código.
- Interoperabilidad:** capacidad de utilizarse en otro emplazamiento y con otro conjunto de herramientas o sobre otra plataforma de componentes de enseñanza desarrolladas dentro de un sitio, con un cierto conjunto de herramientas o sobre una cierta plataforma. Existen numerosos niveles de interoperabilidad.
- Reusabilidad:** flexibilidad que permite integrar componentes de enseñanza dentro de múltiples contextos y aplicaciones. [WKS]

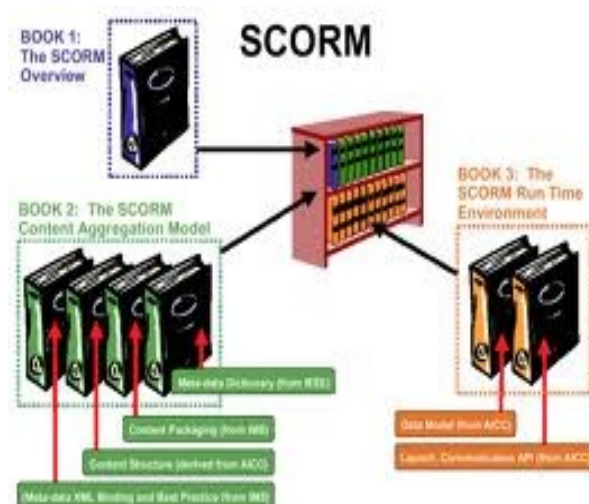


Ilustración 5: Icon of standard SCORM

2.10 IMS

Las especificaciones IMS (2004) son el resultado de una activa iniciativa que está desarrollando y proponiendo especificaciones basadas en tecnologías abiertas (XML) para facilitar las actividades de aprendizaje sobre tecnología web, principalmente para el intercambio de contenidos y de información de los estudiantes. Es una propuesta ambiciosa que cubre, entre otros rubros, accesibilidad y adaptación del estudiante, la definición de competencias, el empaquetamiento de contenidos, información de agentes del proceso educativo, el diseño del aprendizaje a través de un lenguaje para expresar diferentes modelos pedagógicos, así como la formación de repositorios de contenidos digitales. [EIMS]

2.11 LOM

LOM, (del inglés Learning Object Metadata «metadatos para objetos de aprendizaje») es un modelo de datos, usualmente codificado en XML, usado para describir un objeto de aprendizaje y otros recursos digitales similares usados para el apoyo al aprendizaje. Su propósito es ayudar a la reutilización de objetos de aprendizaje y facilitar su interactividad, usualmente en el contexto de sistemas de aprendizaje en línea. [WKLOM]

2.12 Moodle

Moodle es un paquete de software para la creación de cursos y sitios Web basados en Internet. Es un proyecto en desarrollo diseñado para dar soporte a un marco de educación social constructiva.

Moodle se distribuye gratuitamente como software libre (Open Source) (bajo la Licencia Pública GNU). Básicamente esto significa que Moodle tiene derechos de autor (copyright), pero que usted tiene algunas libertades. Puede copiar, usar y modificar Moodle siempre que acepte: proporcionar el código fuente a otros, no modificar o eliminar la licencia original y los derechos de autor, y aplicar esta misma licencia a cualquier trabajo derivado de él.

Moodle puede funcionar en cualquier ordenador en el que pueda correr PHP, y soporta varios tipos de bases de datos (en especial MySQL).

La palabra Moodle era al principio un acrónimo de Modular Object-Oriented Dynamic Learning Environment (Entorno de Aprendizaje Dinámico Orientado a Objetos y Modular), lo que resulta útil para programadores y teóricos de la educación. También es una la palabra que describe el proceso de deambular perezosamente a través de algo, y hacer las cosas cuando se te ocurre hacerlas, una placentera herramienta que a menudo te lleva a la visión y la creatividad. Las dos acepciones se aplican a la manera en que se desarrolló Moodle y a la manera en que un estudiante o profesor podría aproximarse al estudio o enseñanza de un curso en línea. Todo el que usa Moodle es un Moodler.

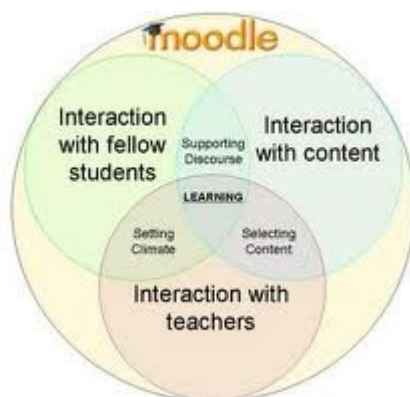


Ilustración 6: Icon of Moodle plataforma

2.13 JavaFx

Es una familia de productos y tecnologías de Sun Microsystems, adquirida por Oracle Corporation, para la creación de Rich Internet Applications (RIAs), que son aplicaciones web que tienen las características y capacidades de aplicaciones de escritorio, incluyendo aplicaciones multimedia interactivas. Las tecnologías incluidas bajo la denominación

JavaFx son JavaFx Script y JavaFx Mobile, aunque hay más productos JavaFx planeados.

JavaFx amplía la experiencia del desarrollador web al proporcionarle medios y contenidos ricos para todas las pantallas que visite. Como usuario, podrá ejecutar las aplicaciones de JavaFx en un explorador o arrastrarlas y colocarlas en el escritorio. Es una interfaz perfecta.

JavaFx cuenta con la tecnología Java, JavaFx amplía la potencia de Java al permitir que los desarrolladores puedan utilizar todas las librerías de Java en las aplicaciones de JavaFx. De esta forma, los desarrolladores pueden ampliar su capacidades en Java y utilizar la innovadora tecnología de presentación que suministra JavaFx para crear experiencias visuales atractivas. [OJFX]

2.14 Conclusiones del capítulo

En este capítulo se mostraron los principales conceptos que nos ayudarán a entender el trabajo de grado: los algoritmos evolutivos (particularmente con el que se va a trabajar, que es la programación por expresión genética); se tuvieron en cuenta conceptos matemáticos con los cuales se trabajará en este trabajo de grado como lo son la integración simbólica y la derivación simbólica.

También se trató el tema de los objetos virtuales de aprendizaje, y se tuvieron en cuenta conceptos sobre términos utilizados en los objetos virtuales de aprendizaje. Todo esto servirá para entender los términos utilizados para el desarrollo de esta la aplicación.

Por último se vieron diferentes tecnologías empleadas, que serán de vital importancia en la realización del proyecto, ya que permitirán una implementación rápida y extensible, dando las bases para la obtención de una aplicación confiable y estandarizada.

3. ESTADO DEL ARTE

3.1 En Programación por Expresión Genética

En programación por expresión genética se han construido varias herramientas para uso académico y comercial. Entre ellas contamos con el software elaborada por Candida Ferreira quien fue la inventora de esta técnica de programación evolutiva.

3.1.1 Gene Expression Programming .NET

En una herramienta para trabajar con algoritmos de programación genética por expresión hecho por Candida Ferreira con el apoyo de Microsoff y elaborada en el framwork de .NET

3.1.2 PyGEP

Es una librería adecuada para el estudio académico del GEP está hecha en Python 2.5 , con el objetivo de facilidad de uso y rápida implementación. La librería usa el elitismo y la ampliación de la aptitud para la selección, los operadores de mutación, crossover y la transposición, y algunas funciones estándar de GEP. [PPG]

3.1.3 GeneXproTools

GeneXproTools es una herramienta de análisis de datos creada para predecir modelos desde datos numéricos. La principal ventaja de GeneXproTools es la facilidad de uso para el usuario y su poderoso algoritmo. El principal enfoque de GeneXproTools es un proceso racional que comienza con la carga de datos. Sigue con la creación automática de datos y concluye con la conversión del modelo a cualquiera de los deiciseis lenguajes soportados. [GSGX]

3.2 En Objetos Virtuales de Aprendizaje

Los Ovas tuvieron sus inicios en los años 90s con el concepto de Objetos de Aprendizaje hasta la fecha han tomado varias interpretaciones y descripciones en los diferentes lugares en los que se desarrollan este tipo de objetos. Afortunadamente, se han podido definir estándares para una mejor implementación tales como IMS, LOM y SCORM.

Los Objetos Virtuales de Aprendizaje en Colombia son apoyados por el Ministerio de Educación y desde su sitio web [www.colombiaaprende.edu.co] estimulan su creación. Allí hay un enlace a la página del banco de objetos a nivel nacional. En esta página se pueden encontrar muchos tipos de objetos que han sido realizados en diferentes universidades del país aunque vale recalcar que los objetos encontrados en ese sitio no son muy interactivos. Muchos son de tipo pdf y páginas web planas. Es decir, son objetos estáticos, sin interactividad.

En la Universidad del Valle este tema se han venido desarrollando en los últimos años por la Dirección de Nuevas Tecnologías y Educación Virtual (DINTEV) en la cual la definición de objeto virtual de aprendizaje ha tenido varias mejoras. En el campus virtual hay cursos donde enseñan a los profesores; cómo crear objetos virtuales de aprendizaje por medio de herramientas de uso gratuito; cómo subir esos OVAs a campus virtual el cual está hecho en la plataforma Moodle; cómo se puede integrar fácilmente el objeto creado al sitio web (ya que los objetos los empaquetan con el estándar de SCORM).

Las herramientas libres más conocidas para la creación y empaquetamiento para los diferentes estándares para objeto de aprendizaje son EXE, XERTE, Courselab, eLML, Myudutu entre otras. Estas herramientas permiten crear OVAs con presentaciones con cierto grado de interactividad cumpliendo con los requisitos para ser empaquetados en los formatos estándar.

3.3 Conclusiones del capítulo

En este capítulo se pudieron ver algunas herramientas que presentan diferentes soluciones para algunos tipos de problemas empleando la programación por expresión genética. Como se ve no hay ninguna herramienta que tenga como objetivo ser un objeto virtual de aprendizaje, además de ser aplicaciones de uso propietario, solo PyGep es libre pero solo es una librería.

Otro punto a tener en cuenta es que estas herramientas de carácter propietario solo funcionan en el sistema operativo Windows por lo cual no pueden utilizar en otros sistemas operativos y esto limita su uso.

El uso de estándares en los OVA supuso un gran avance, pero limita la interactividad que se puede obtener con el desarrollo de estos objetos en diferentes lenguajes de programación distintos a java ya que no hay herramientas que permitan empaquetar aplicaciones desarrolladas en otros lenguajes de programación en algún estándar para cumplir con los aspectos básicos de un objeto virtual de aprendizaje.

4. DESCRIPCIÓN DE LA SOLUCIÓN

La aplicación está enmarcada dentro del área de la computación evolutiva y será capaz de resolver problemas de interpolación simbólica, predicción de secuencias numéricas, interpolación de integrales y derivadas simbólicas por medio de la programación por expresión genética. Con esta herramienta se pretende a partir de nubes de puntos encontrar las funciones matemáticas si las hay que pasan por cada punto.

Con tratar de resolver estos problemas de interpolación simbólica se busca que los estudiantes de esta materia comprendan un poco como funciona la programación por expresión genética por medio del control de la ejecución del algoritmo y la visualización en cada momento de la solución propuesta. Finalmente se desea construir un objeto virtual de aprendizaje que refuerce los conocimientos aprendidos en la asignatura de computación evolutiva especialmente en el tema de programación por expresión genética.

4.1 Metodología Utilizada

El presente estudio tiene un carácter aplicativo, pues por medio del uso conceptos teóricos en una herramienta de software, se pretende simular el funcionamiento de la programación por expresión genética.

Para el desarrollo de la aplicación se han seleccionado técnicas pertenecientes a la programación extrema (xp) debido a que es una metodología de ágil de desarrollo de productos de software, no se ha implementado toda la metodología pero si la cosas que fueran más relevantes para la aplicación.

4.2 Historias de usuario

Las historias de usuario fueron realizadas con el profesor de la asignatura de computación evolutiva

Historia de Usuario # 1							HU1-DI
Nombre	Diseño de la interfaz						
Prioridad en proyecto	X			Riesgo en desarrollo			X
	Alta	Media	Baja		Alto	Medio	Bajo
Cliente	Profesor Angel Garcia Baños						
Descripción	Para hacer uso de la aplicación el usuario debe contar con un interfaz de fácil manejo con la cual interactuar sin necesidad de entrar en muchas pantallas y no se pierda a la hora de realizar una función de la aplicativo						
Observaciones:	Se realizó un bosquejo de la interfaz						
Historia de usuario relacionada							

Historia de Usuario # 2							HU2-IA
Nombre	Interactividad del Algoritmo						
Cliente	Profesor Angel Garcia Baños						
Prioridad en proyecto	X			Riesgo en desarrollo		X	
	Alta	Media	Baja		Alto	Medio	Bajo
Descripción	Esta característica permitirá a los estudiantes ver de una mejor manera el funcionamiento del algoritmo GEP ya que podrá de manera controlada observar como va el algoritmo encontrando las soluciones						
Observaciones:	Se creará un panel donde el usuario podrá controlar la ejecución por medio de botones.						
Historia de usuario relacionada							

Historia de Usuario # 3							HU3-MFS
Nombre	Manejo de Funciones simbólicas						
Prioridad en proyecto	X			Riesgo en desarrollo		X	
	Alta	Media	Baja		Alto	Medio	Bajo
Cliente	Profesor Angel Garcia Baños						
Descripción	El aplicativo contará con la ejecución del algoritmo para resolver cuatro tipo de problemas: interpolación simbólica, predicción simbólica, integración simbólica y derivación simbólica.						
Observaciones:	El estudiante podrá escogerlo el tipo de problema a resolver por medio de un selector						

Historia de usuario relacionada	
--	--

Historia de Usuario # 4							HU4-VFMG
Nombre	Visualizar funciones mediante gráficos						
Prioridad en proyecto		X		Riesgo en desarrollo		X	
	Alta	Media	Baja		Alto	Medio	Bajo
Cliente	Profesor Angel Garcia Baños						
Observaciones:	El estudiante podrá visualizar las funciones escogidas en un cuadro donde aparecerá la gráfica en la parte izquierda						

Historia de Usuario # 5							HU5-VAA
Nombre	Visualizar ayudas de la aplicación						
Prioridad en proyecto		X		Riesgo en desarrollo		X	
	Alta	Media	Baja		Alto	Medio	Bajo
Descripción	En cualquier momento el usuario debe poder contar con ayudas que la faciliten el manejo de la aplicación y el entendimiento del algoritmo GEP						
Observaciones:	El estudiante podrá visualizar las ayudas dentro la interfaz principal.						

Historia de Usuario # 6							HU6-RAA
Nombre	Realizar animación del algoritmo						
Prioridad en proyecto		X		Riesgo en desarrollo		X	
	Alta	Media	Baja		Alto	Medio	Bajo
Cliente	Profesor Ivan Mauricio Cabezas						
Descripción	EL usuario podrá ver una animación del algoritmo cuando esté funcionando para mayor comprensión del mismo						
Observaciones:	El estudiante podrá visualizar la animación dentro la interfaz principal.						

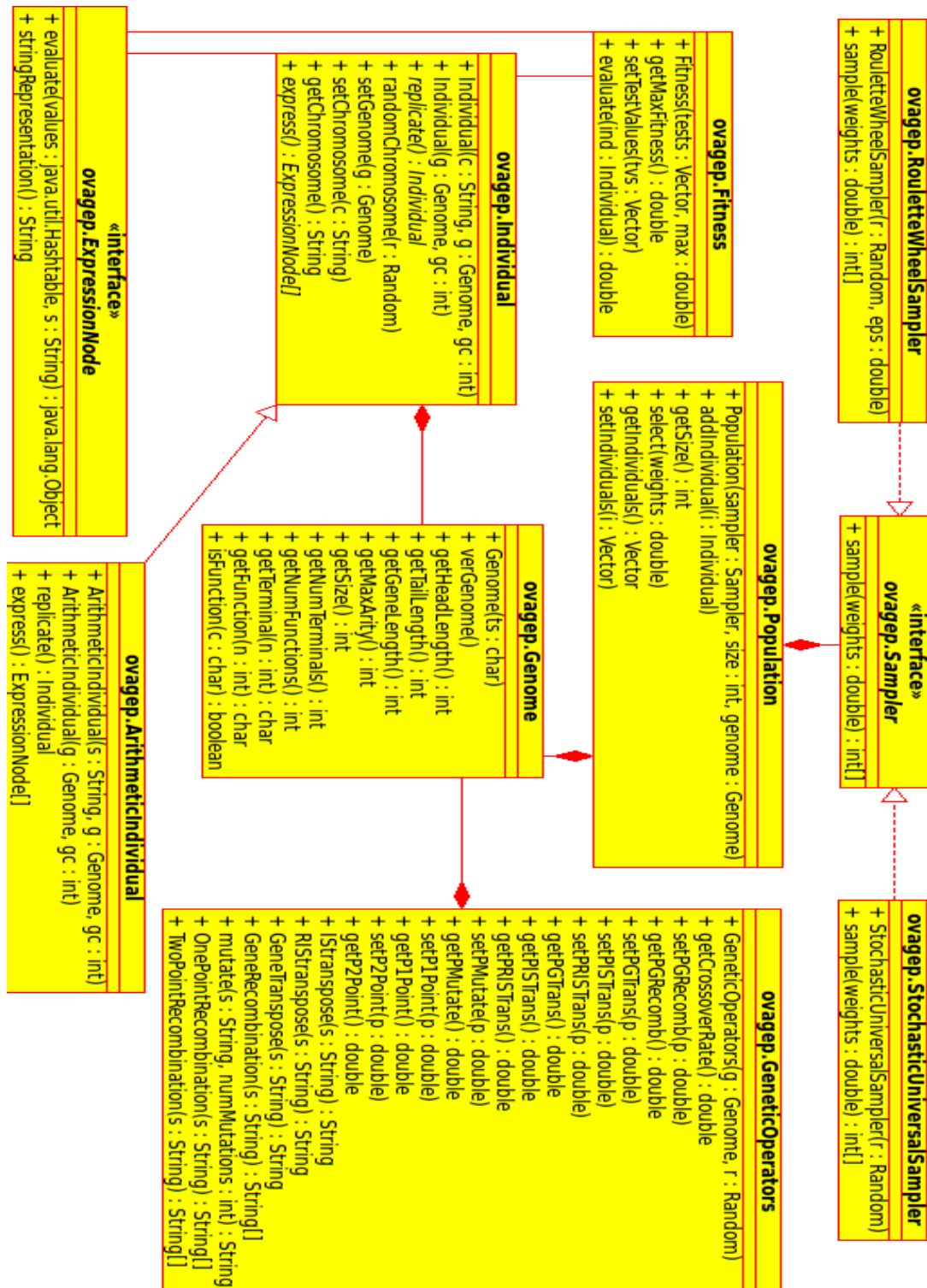


Ilustración 2: Diagrama de clase, modulo algoritmo GEP

4.3 Casos de Uso

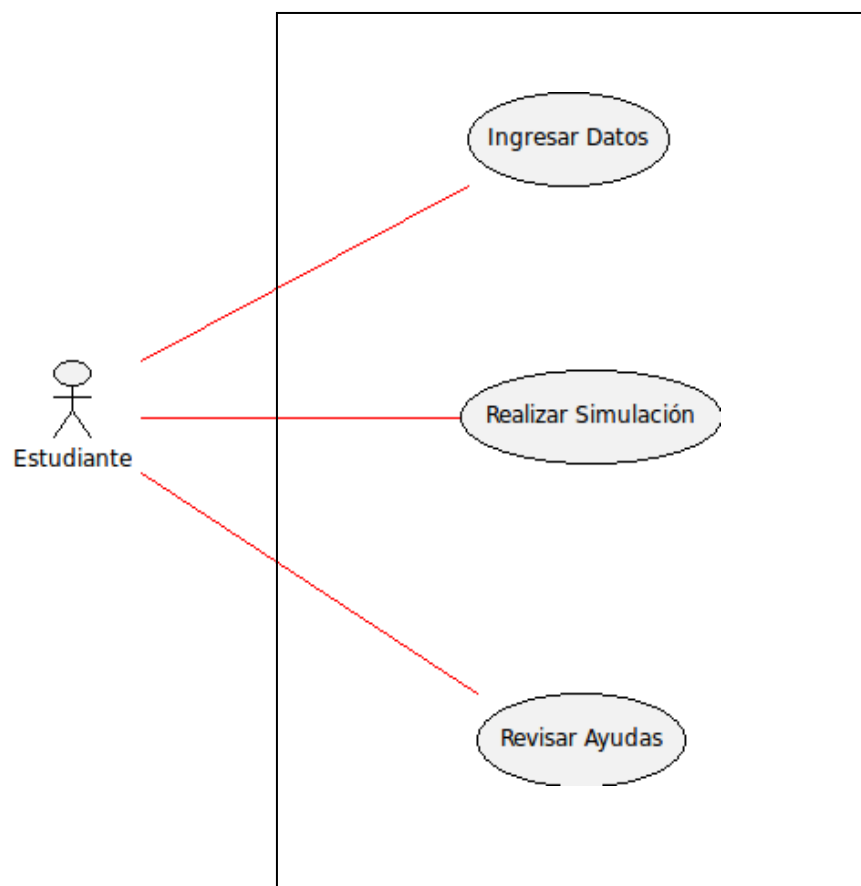


Ilustración 9: Representación gráfica de los casos de uso

4.4 Casos de uso en formato extendido

Caso de uso	CU 1 Ingresar datos	
Actor	Usuario	
Propósito	Permitir el ingreso de los datos de las funciones para la ejecución de la aplicación	
Descripción	En esta caso de uso se describen las acciones correspondientes para llevar a cabo el ingreso de los datos al programa como los son la función matemática, el dx y los rangos para las funciones.	
Tipo	Esencial y primario.	
Referencia Cruzada		
Pre condiciones	ninguna	
Post condiciones	ninguna	
Flujo De Eventos		
Flujo Normal		
Acción Del Usuario	Respuesta del Sistema	
1. El usuario presiona en la lista de métodos de ejecución.	2. Despliega un listado con los diferentes métodos de ejecución.	
3. Selecciona un método de su interés 4. El usuario presiona el botón Ecuación	5. Muestra un ventana interna la cual contiene números y funciones y espacio de resultados.	
6. Ingresar la función 7. El usuario presiona el botón cerrar	8. Cierra la ventana interna 9. Actualiza el campo de la ecuación	
10. Ingresar el valor en el campo dx 11. ingresa el valor de los rangos.		

Tabla 1: C.U. en formato extendido: Ingresar datos

Caso de uso	CU 2 Realizar Simulación	
Actor	Usuario	
Propósito	Permitir al usuario ver la ejecución del algoritmo y permitir la interacción con los diferentes resultados del algoritmo genético	

Caso de uso	CU 2 Realizar Simulación	
	por expresión	
Descripción	En esta caso de uso se describen las acciones correspondientes a la ejecución del algoritmo para llevar a cabo el proceso de simulación, interacción y muestra de resultados del mismo.	
Tipo	Esencial y primario	
Referencia Cruzada		
Pre condiciones	Los datos de las funciones deben estar escritos	
Flujo De Eventos		
Flujo Normal		
Acción Del Usuario	Respuesta del Sistema	
1. Presiona el botón comenzar	2. Ejecuta los cálculos correspondientes para convertir la función en arreglos de puntos. 3. Muestra un panel en la parte inferior derecha con un botón Iniciar y dos botones inactivos. 4. Muestra en la parte central izquierda un panel con un gráfico que contiene la función ingresada.	
5. Presiona el botón Iniciar	6. Inicia la ejecución del algoritmo 7. Desactiva el botón Iniciar 8. Activa el botón Parar.	
9. Presiona el botón Parar	10. Muestra un listado con las funciones “cercanas” o la solución del problema. 11. Desactiva el botón Parar. 12. Activa el botón Continuar.	
13. Presiona en un función de la lista mostrada.	14. Muestra en la ventana del gráfico de la parte izquierda la linea de la función presionada además de la linea de la unción original.	
15. Presiona en la parte de abajo donde dice animación.	16. Aparece un cuadro mostrando una animación sobre el funcionamiento de algoritmo por expresión genética	
17. Presiona el botón Continuar	18. Reanuda la ejecución del algoritmo.	

Caso de uso	CU 2 Realizar Simulación
	19. Activa el botón Parar.
	20. Si el algoritmo encuentra la solución aparece un mensaje en una ventana diciendo que ya encontró la solución.

Tabla 2: C.U en formato extendido: Realizar simulación

Caso de uso	CU 3 Revisar Ayudas
Actor	Usuario
Propósito	Permitir al usuario la visualización de las ayudas de la aplicación
Descripción	En esta caso de uso se describen las acciones correspondientes a la visualización de las ayudas que presenta el sistema para su ejecución y mejorar el entendimientos de la programación por expresión genética.
Tipo	Trazo fino y secundario
Pre condiciones	Ninguna
Post condiciones	Ninguna
Flujo De Eventos	
Flujo Normal	
Acción Del Usuario	Respuesta del Sistema
1. Pasa el ratón por encima del botón "Ayuda GEP"	2. Muestra una ventana con la información sobre GEP.
3. Presiona el scroll inferior de la ventana	4. La ventana muestra más contenido mostrando la información que tiene en la parte inferior.
5. Pasa el ratón por encima del botón "Ayuda Aplicación"	6. Muestra una ventana con la información sobre la aplicación y que pasos seguir para su funcionamiento.
7. Presiona el scroll inferior de la ventana	8. La ventana muestra más contenido mostrando la información que tiene en la parte inferior.
9. Pasa el ratón por encima del botón "Acerca de "	10. Muestra una ventana con la información de los créditos de la aplicación.

Caso de uso CU 3 Revisar Ayudas	
11. Presiona el scroll inferior de la ventana	12. La ventana muestra más contenido mostrando la información que tiene en la parte inferior.

Tabla 3: C.U en formato extendido: Revisar ayudas

4.5 Detalles de Implementación

A continuación se describen los aspectos más relevantes sobre la implementación del algoritmo de programación por expresión genética. Se recomienda al lector revisar los conceptos descritos en el Capítulo 2.

La aplicación se creó a partir de una librería de GEP para Java llamada “Java GEP toolkit” la cual presenta una estructura que sirve de base para realizar la implementación de clases para la buscar soluciones empleando programación por expresión genética, estos archivos fueron modificados en su mayoría para poder adaptarlos a el problema de la interpolación simbólica. A partir de esta librería se construyeron el resto de las clases necesarias para el funcionamiento de la aplicación incluyendo las interfaces necesarias y la interacción entre Java y JavaFx.

Interacción entre clases y funcionamiento del algoritmo.

La clase **Genome** representa a los elementos que componen a los genes es decir funciones y terminales. En esta clase la cola sólo puede contener caracteres terminal ello es un requisito para garantizar que todo el cromosoma resultante después de aplicar cualquier operador genético tenga un cromosoma válido. Esta clase principalmente interactúa con la clase **GeneticOperators** que donde se realizan las distintas operaciones y utiliza a la clase **Population** que representa la población de individuos y también a la clase **Individual** la cual contiene cada uno de los cromosomas.

La clase **Population** representa una población. Esto incluye el almacenamiento de todos cromosomas, el manejo de selección y la reproducción. En la fase selección para cada población se da una serie de pesos que están determinados por la clase **Fitness** (de modo que el mejor individuo tiene el mayor peso) y el índice del mejor individuo. Esto nos permite llamar a una clase **Sampler** que es la encargada de utilizar los esos pesos, y así asegurar que las mejores instancias de la clase Individual en la población.

La clase **Individual** es donde se guarda el cromosoma generado por la clase **Genome**, el cual se puede replicar y mandar la información del cromosoma a otras clases como la clase **ArithmeticIndividual**.

La clase **ArithmeticIndividual** es la base de la clase **Individual** la cual evoluciona en expresiones que representan las operaciones aritméticas sobre las múltiples variables, también contiene los operadores o funciones matemáticas.

La clase **ArithmeticExpressionNode** es una implementación de la clase interna de la expresión específica del nodo del objeto de la clase **Individual** que va a utilizar. Describe a este cromosoma como un árbol de expresión para las operaciones aritméticas.

La clase **ExpressionNode** contiene el método de evaluación además tiene una tabla hash donde almacena los valores. La clase **ExpressionNode** maneja a la clase **Individual** como un mapa de almacenamiento de tuplas representada por la tabla hash de una dimensión de objetos.

La clase **Fitness** está diseñada para servir como evaluación de aptitud de la población a partir de un conjunto de casos de prueba y los objetos de la clase **Individual**. Cada elemento del vector debe ser una tabla hash donde se asigna un nombre y un valor del tipo de esperado por un objeto **ExpressionNode** de cada objeto **Individual**. En esta clase se evalúa cada punto de la función, se calcula que tan lejos está del punto deseado y se le asigna un valor, el cual resulta de la suma de la diferencias de cada punto de la función encontrada con cada punto de la función original. Para mejorar la función de aptitud se aplico a cada sumatoria el cálculo del error absoluto.

La clase **GeneticOperators** contiene todos los operadores en los genes ocurren de forma individual (como la mutación) o durante la reproducción (por ejemplo, la recombinación). Las probabilidad que tiene cada operador es calculada en esta clase, esto permite cada operador tenga asociado a él una probabilidad de ser utilizado.

Esto permite realizar las dos operaciones principales:

- **Mutación:** Dada una cadena y el número de mutaciones, devolverá una cadena con ese número de mutaciones. Cada objeto de la clase **Individual** tendrá una probabilidad de mutación que a su vez será guardado en una variable que informe sobre el numero de mutaciones.
- **Cruce:** Dada una matriz de dos elementos que contienen dos cromosomas de cadena, la cual lleva a cabo una recombinación de un punto entre ellos y devolver una nueva matriz de dos elementos con los nuevos cromosomas. Esto es llamado por un objeto externo que ha utilizado la probabilidad almacenada en este objeto para determinar si la recombinación en un punto debe ocurrir.

Para calcular la probabilidad se utilizan las siguientes clases:

- La clase **RouletteWheelSampler** proporciona el Método de rueda de ruleta de muestreo para la etapa de selección en los programas de GEP. Dado un conjunto de pesos (entre 0 y 1), devuelve una lista de igual longitud que contiene los índices de los individuos seleccionados. La Selección por ruleta (roulette-wheel selection) es también conocida como selección proporcional a la función de desempeño. Sea N el número de individuos existentes y f_i el desempeño del i -ésimo individuo. La probabilidad asociada a su selección esta dada por:

$$p_i = \frac{f_i}{\sum_{j=1}^N f_j}$$

- La clase **StochasticUniversalSampler** proporciona un método de muestreo estocástico universal para la etapa de selección en los programas de GEP.

- La clase **Sampler** es una Interfaz que define un muestreo genérico. Un muestreo toma un conjunto de ponderaciones para un conjunto de elementos y devuelve una matriz de índices correspondientes a las personas seleccionadas. La longitud de la matriz devuelta coincide con la longitud de la matriz de peso.

La clase **HiloGep** es la encargada de hacer la conexión con todas las clases involucradas en la GEP, en esta se configura los parámetros con que se va a trabajar, los terminales y funciones matemáticas con las que se va a dar las soluciones en este caso los terminales son:

- '1', 'x', '2', 'x', '3', 'x', '4', 'x', '5', 'x', '6', 'x', '7', 'x', '8', 'x', '9', 'x'

El terminal x lo repito varias veces para asegurar que se encuentre presente en la mayoría de los cromosomas.

Y los operadores o funciones matemáticas son:

- '+', '-', '/', '*', 'r', 's', 'c', 't', 'l', '^'

Se encuentran los operadores básicos de suma (+), resta (-), multiplicación (*), división (/) y otros un poco más complejos como raíz cuadrada (r), seno (s), coseno (c), tangente (t), logaritmo (l) y exponente (^).

En esta clase también se implementa el hilo con el cual se encarga de interactuar con cada una de la clase que hace posible evolucionar los cromosomas para buscar la función este hilo termina cuando se encuentra la solución, es decir se encuentra la función deseada.

La clase **InterfazGep** se encarga presenta una interfaz para el control del hilo de la clase **HiloGep**, con esta interfaz gráfica podemos realizar tres acciones:

- **Iniciar:** Inicia el hilo de la clase **HiloGep** y empieza la ejecución del algoritmo

- **Parar:** Para el hilo de la clase **HiloGep** y muestra en pantalla la lista de funciones encontradas hasta el momento, las cuales pueden seleccionar y ser mostradas en el graficador de la interfaz gráfica principal.

- Continuar: Esta opción hace que el hilo se continúe con su ejecución hasta que encuentre la solución.

La clase **Derivada** sirve para a partir de punto iniciales buscar la derivada de cada punto para crear un arreglo de puntos derivados los cuales van a servir para hallar la función derivada mediante el algoritmo GEP. La formula utilizada de la es la formula general donde en la parte superior tenemos la diferencia entre dos imágenes evaluadas de x y abajo el diferencial de x:

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$$

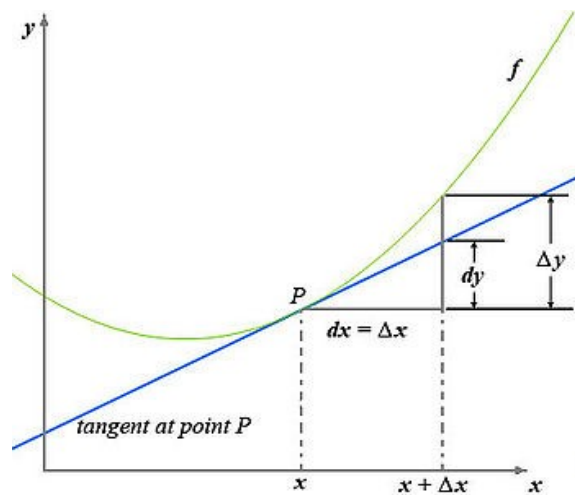


Ilustración 10: Gráfico de la función derivada

La traducción en el algoritmo es la siguiente :

$$f'(x) = \frac{f(x+dx) - f(x)}{dx}$$

Donde dx es un valor ingresado por el usuario

La clase **Integral** sirve para a partir de punto iniciales buscar la integral de cada punto para crear un arreglo de puntos integrados los cuales van a servir para hallar la función de integración mediante el algoritmo GEP. La formula de la del método de la integral de Riemann la cual se define en términos de sumas de Riemann de funciones respecto de *particiones etiquetadas* de un intervalo. Sea $[a,b]$ un intervalo cerrado de la recta real; entonces una *partición etiquetada* de $[a,b]$ es una secuencia finita, esto divide al intervalo $[a,b]$ en n sub-intervalos $[x_{i-1}, x_i]$, cada uno de los cuales es "etiquetado" con un punto especificado t_i de; $[x_{i-1}, x_i]$. En este caso multiplicamos el diferencial de x, por la suma entre dos imágenes evaluadas de x , con distancia igual al diferencial de x y luego dividimos por 2:

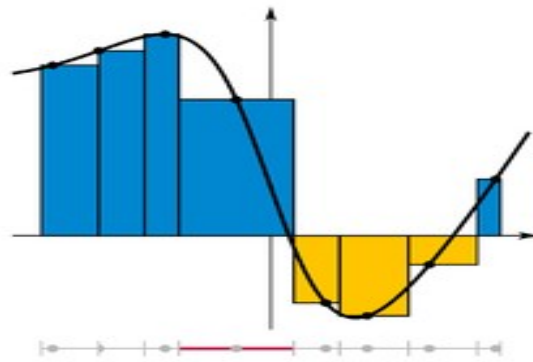


Ilustración 11: Gráfico del calculo de la integral

Para la implementación en el algoritmo es la siguiente se utilizó la formula del trapecio :

$$\int f(x) = \frac{dx * (f(x+dx) + f(x))}{2}$$

La clase **Puntos** sirve para crear puntos a partir de una formula, se enlaza con la clase Parseador para generar puntos los cuales son almacenados en arreglos que luego son utilizados en los diferentes problemas para encontrar por medio del algoritmo de GEP una función.

La clase **Graficas** es una interfaz gráfica que permite visualizar funciones de diferente tipo en un plano x,y, la cual también permite aumentar una región en el plano para analizar mejor los datos, esto lo hace gracias a una librería libre de Java llamada jfreechart.

La clase **PanelCalculadora** es una interfaz que sirve para escribir la función con la cual se va a trabajar esta se conecta con la clase **Parseador** para validar si la función creada con la con esa interfaz es válida.

La clase **Parseador** que parsea una expresión matemática, para parsear expresión se escribe `Obj.parsear(Expr)`. La función devuelve un String con la Expresión en notación post-fija, además el programa también guarda de manera automática la última expresión parseada. La variable utilizada que se utilizó en la en la funciones es x. Esta clase también es utilizada como evaluadora de expresiones simbólicas.

Las clases **InfoItem**, **InfoNodo** y **Mensaje** proveen de una Interfaz gráfica la cual muestra una serie de información que sirve de ayuda al estudiante para una mejor comprensión de los algoritmos genéticos en especial el algoritmo de programación por expresión genética, como también de ayudas para el uso de la aplicación. Esta ayuda fue creada para que funcionará por medio de ventana flotantes para que se mostrarán en la interfaz principal si necesidad de abrir otras ventanas o pestañas y el usuario final no se confundiera en el manejo de la aplicación.

La clase **InterfazPrincipal** es la interfaz gráfica en donde se visualizan todas las demás interfaces esta clase al igual que: **PanelCalculadora**, **InfoItem**, **InfoNodo** y **Mensaje** fueron desarrolladas en el lenguaje JavaFx para mejorar un poco más la apariencia las interfaces en comparación con las que se podrían hacer con Java.

4.6 Conclusiones del capítulo

En este capítulo se pudieron apreciar de manera detallada todos los elementos utilizados para construir la aplicación como lo fueron los diagramas de clase, los casos de uso, las historias de usuario, las librerías y lenguajes utilizados, también se hizo un resumen del funcionamiento de la aplicación y la interacción de sus clases.

Como se vio en este capítulo el desarrollo de la aplicación fue elaborado en dos lenguajes de programación de la misma familia pero con sintaxis muy diferente, en Java fueron creadas las diferentes clases que sirvieron para el procesamiento de datos y en JavaFx se crearon la mayoría de la interfaces.


También se puede apreciar que el modulo del algoritmo se relaciona con el resto la aplicación razón por la cual puede ser usado en otros desarrollos con mucha facilidad ya que no se creó dependencia hacia de las demás clases.

Por último el manejo de hilos permitió la interacción del usuario con el algoritmo aparte de que esta interacción pudiera ser acompaña de un gráfico de la función el cual puede cambiar cuando se escoge un función diferente.

5. PRUEBAS Y RESULTADOS

La aplicación fue probada realizando distintas pruebas con diferentes funciones y viéndolos tiempos de espera para encontrar la solución en los diferentes métodos que provee la aplicación.


A continuación se mostrarán varias pruebas de ejecución del objeto.

 Universidad del Valle	INSTITUCIÓN ISO 9001-2000		Documento:	Rev.: 1
	Aplicación /Módulo: Interfaz			Página: 1 de #
	Nombre: Orlando Cossio Murillo			

DOCUMENTOS DE REFERENCIA

Historia de usuario	HU1-DI
Objetivo	Conocer si la interfaz es usable
Datos de Prueba	El Dx con valor 5, el valor mínimo igual 0 y el valor máximo iguala 50, la función fue $3x^2$
Procedimiento de Prueba	Escogió el método, ingreso los valores dx , mínimo y máximo luego ingreso la ecuación y luego dio clic en el botón comenzar y inicio el proceso le dio para para ver si la población estaba cerca, luego le dio continuar hasta encontrar la solución
Resultado Esperado	Que el usuario pueda manejar la intefaz sin muchos inconvenientes
Resultado Obtenido	Prueba Exitosa SI(x) NO()
Comentarios	la pareció usable al usuario


Realizado por:	Firma	Fecha
Jhon Jairo Escobar		18/02/11
Aprobado por:	Firma	Fecha

	INSTITUCIÓN ISO 9001-2000		Documento:	Rev.: 1
	Nombre: Orlando Cossio Murillo			Página: 1 de #

DOCUMENTOS DE REFERENCIA

Historia de usuario	HU2-IA
Objetivo	Saber la interactividad con el algoritmo
Datos de Prueba	El Dx con valor 5, el valor mínimo igual 0 y el valor máximo iguala 50, la función fue $3x^2$
Procedimiento de Prueba	Le parece conveniente porque puede parar la ejecución y ver si el algoritmo se está acercando a la solución
Resultado Esperado	Que el usuario pueda interactuar con el algoritmo
Resultado Obtenido	Prueba Exitosa SI(x) NO()
Comentarios	Le parecio apropiado este uso para el algoritmo

Realizado por:	Firma	Fecha
Jhon Jairo Escobar		18/02/11
Aprobado por:	Firma	Fecha

	INSTITUCIÓN ISO 9001-2000	Documento:	Rev.: 1
	Nombre: Orlando Cossio Murillo		Página: 1 de #

DOCUMENTOS DE REFERENCIA

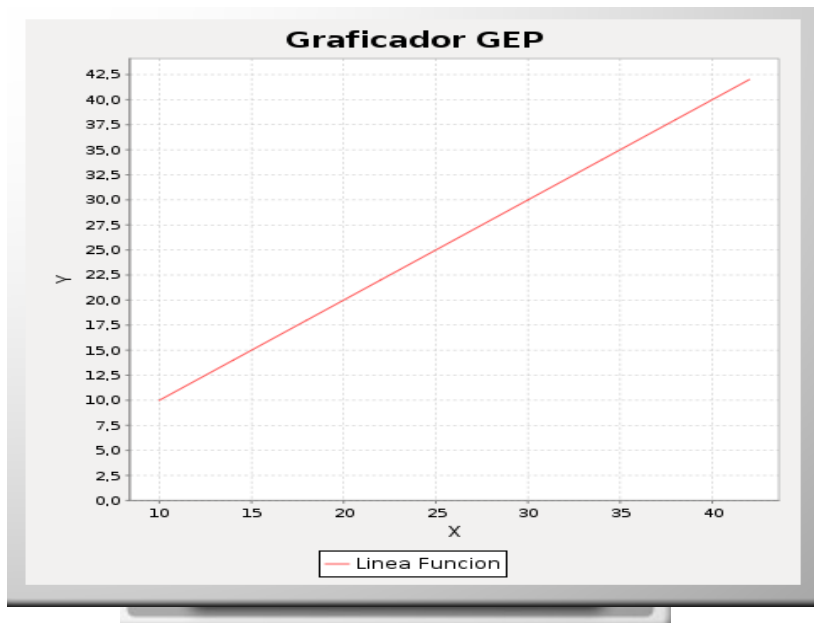
Historia de usuario	HU2-IA
Objetivo	Saber la interactividad con el algoritmo
Datos de Prueba	El Dx con valor 5, el valor mínimo igual 0 y el valor máximo iguala 50, la función fue $3x^2$
Procedimiento de Prueba	Le parece conveniente porque puede parar la ejecución y ver si el algoritmo se está acercando a la solución
Resultado Esperado	Que el usuario pueda interactuar con el algoritmo
Resultado Obtenido	Prueba Exitosa SI(x) NO()
Comentarios	Le parecio apropiado este uso para el algoritmo

Realizado por:	Firma	Fecha
Jhon Jairo Escobar		18/02/11
Aprobado por:	Firma	Fecha

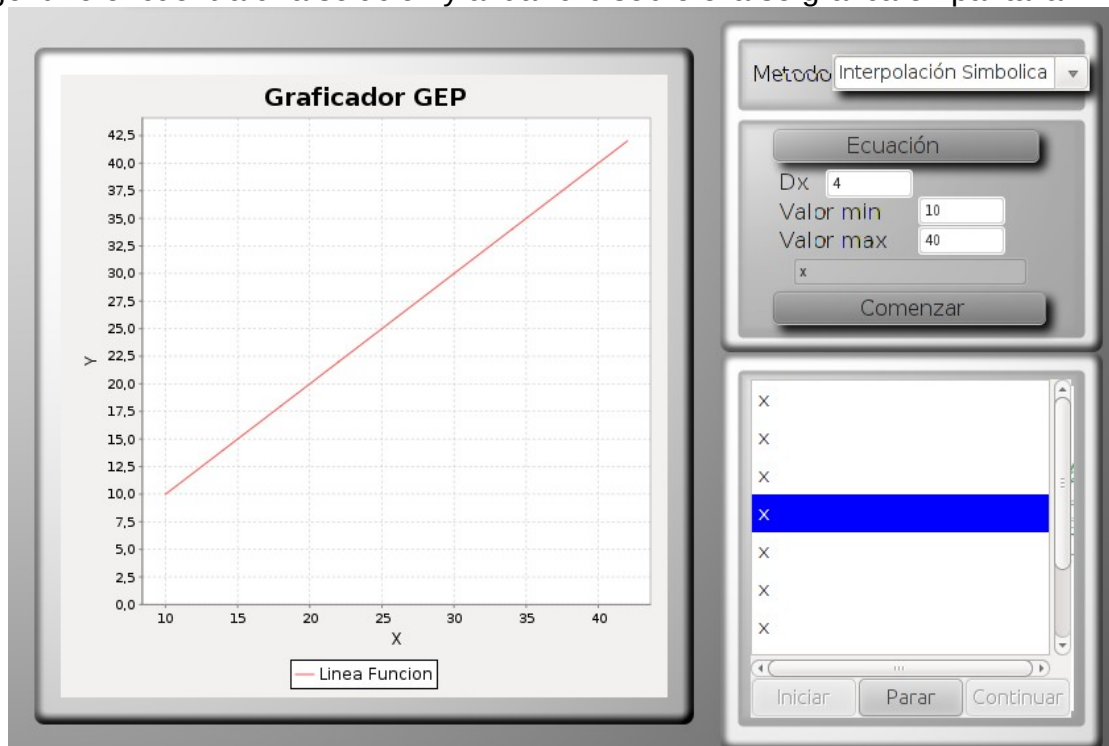
Prueba 1

1. Se ingresan los datos:

2. Se muestra la gráfica de la función inicial:

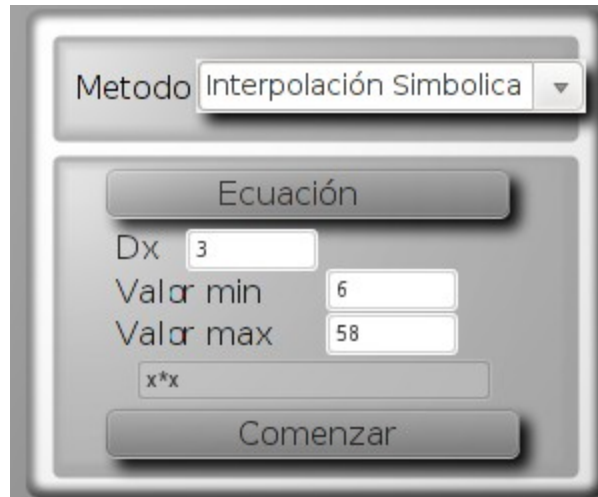


3. El algoritmo encuentra una solución y al dar clic sobre ella se gráfica en pantalla



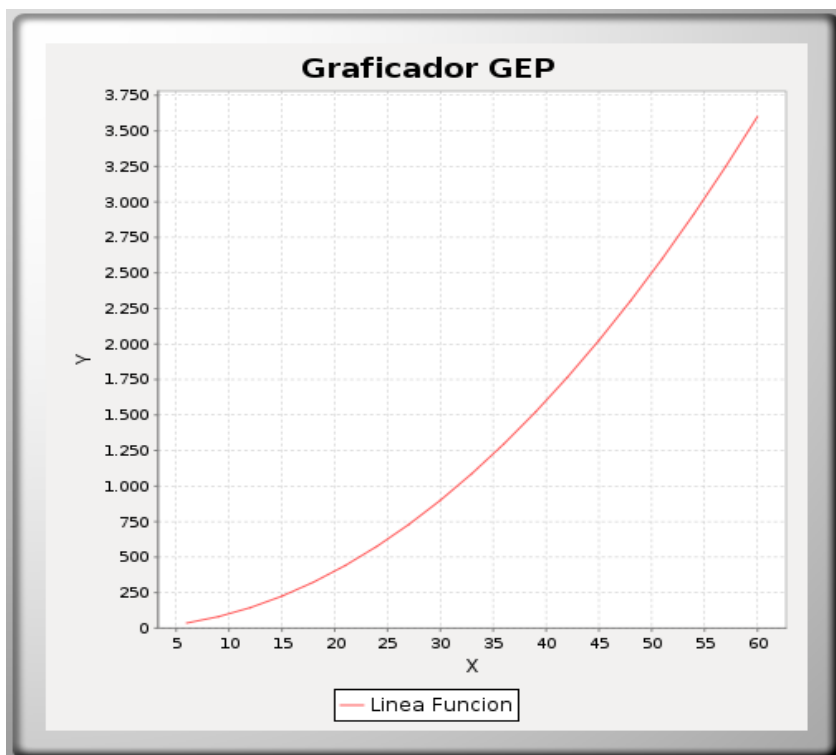
Prueba 2

1. Se ingresan los datos:

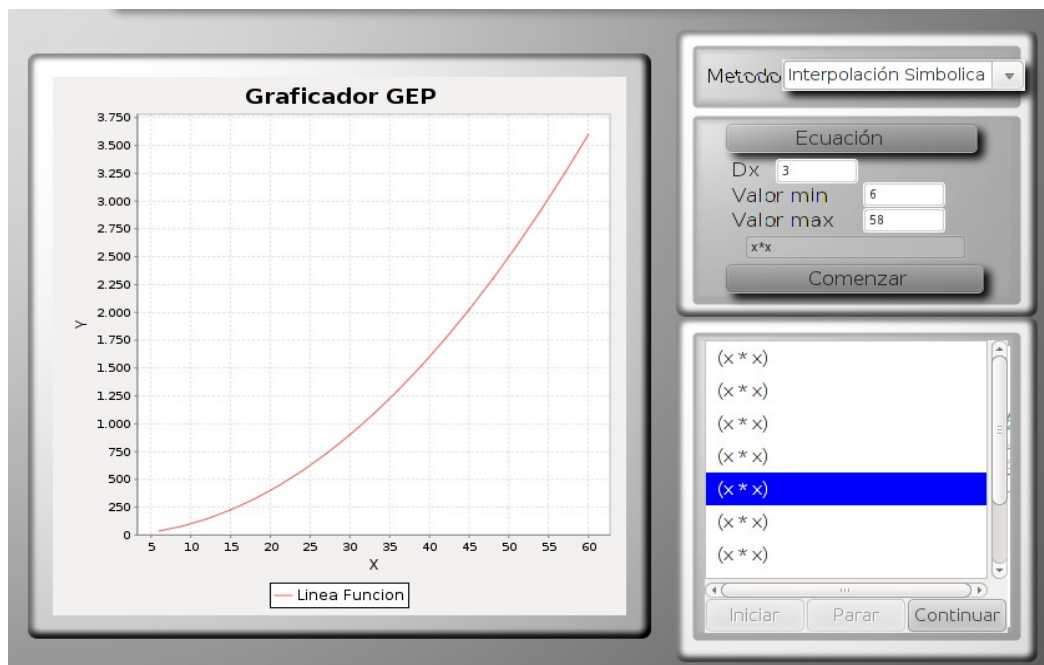


The screenshot shows a software window titled 'Ecuación' (Equation). At the top, there is a dropdown menu labeled 'Metodo' (Method) with 'Interpolación Simbolica' (Symbolic Interpolation) selected. Below this, there are three input fields: 'Dx' with the value '3', 'Valor min' (Minimum Value) with the value '6', and 'Valor max' (Maximum Value) with the value '58'. Below these fields is a text input area containing the expression 'x*x'. At the bottom of the window is a button labeled 'Comenzar' (Start).

2. Se muestra la gráfica de la función inicial:



3. El algoritmo encuentra una solución y al dar clic sobre ella se gráfica en pantalla:

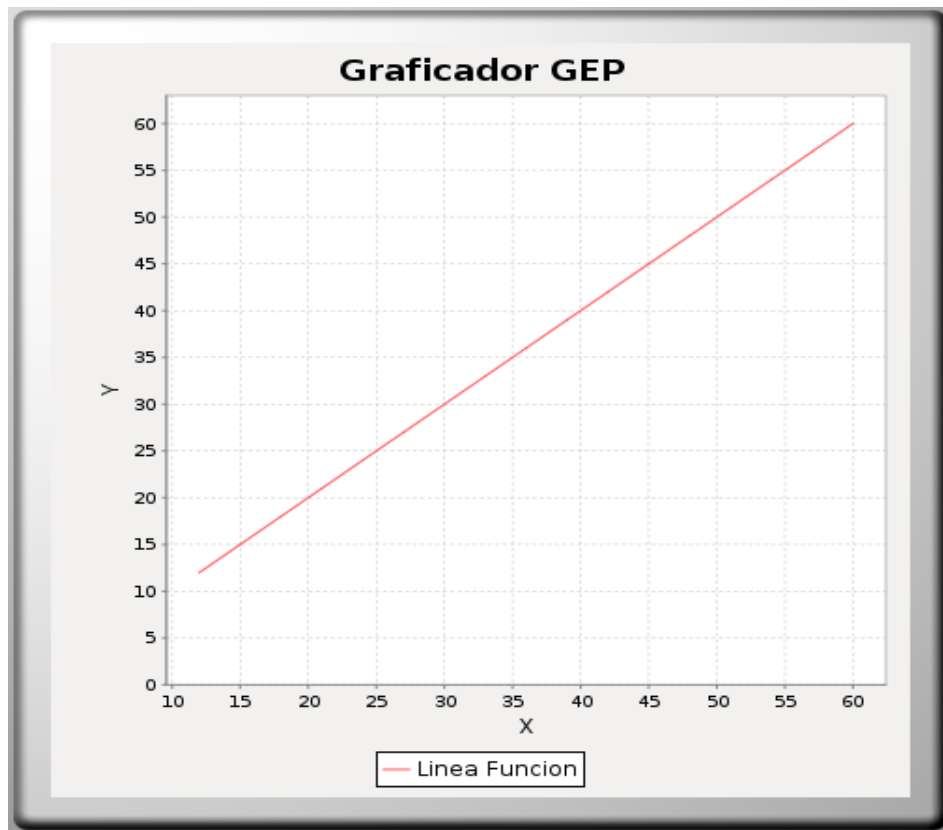


Prueba 3

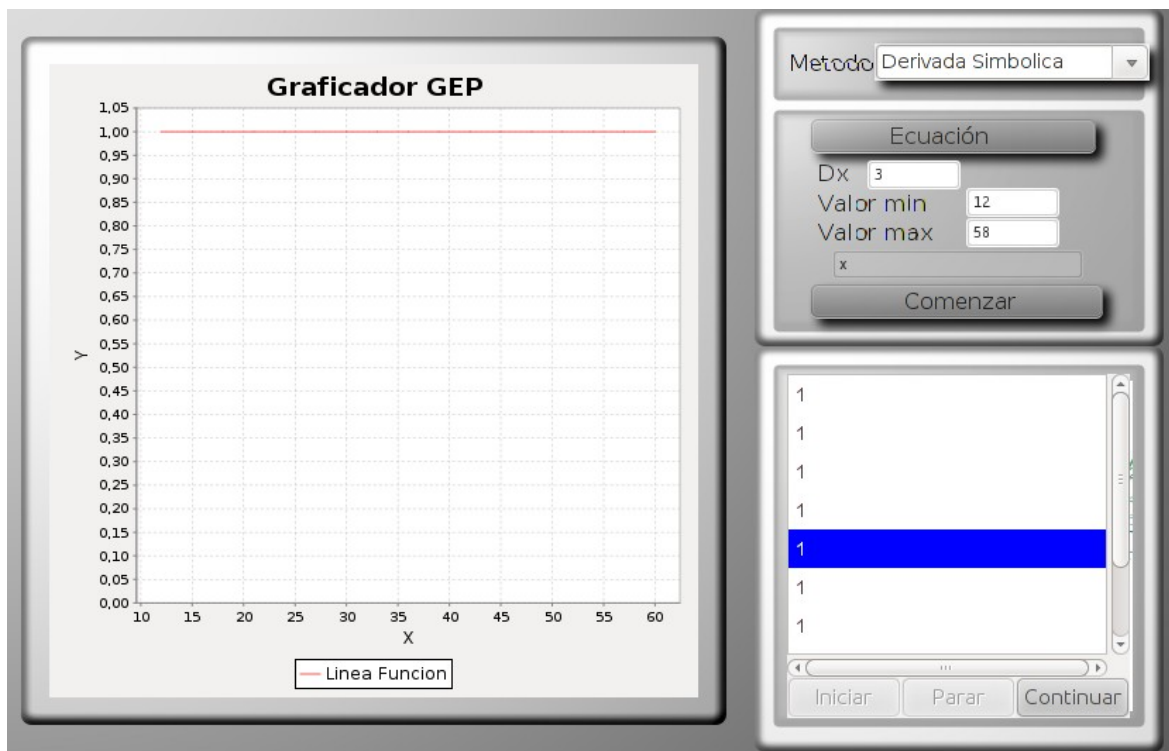
1. Se ingresan los datos:

This screenshot shows the 'Graficador GEP' software interface with the 'Metodo' dropdown set to 'Derivada Simbolica'. The 'Ecuación' section contains input fields for 'Dx' (3), 'Valor min' (12), and 'Valor max' (58), and a text box containing 'x'. A 'Comenzar' button is located below these fields.

2. Se muestra la gráfica de la función inicial:

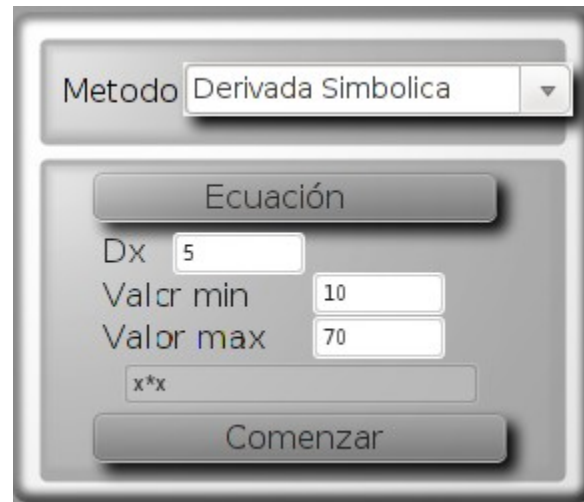


3.El algoritmo encuentra una solución y al dar clic sobre ella se gráfica en pantalla:



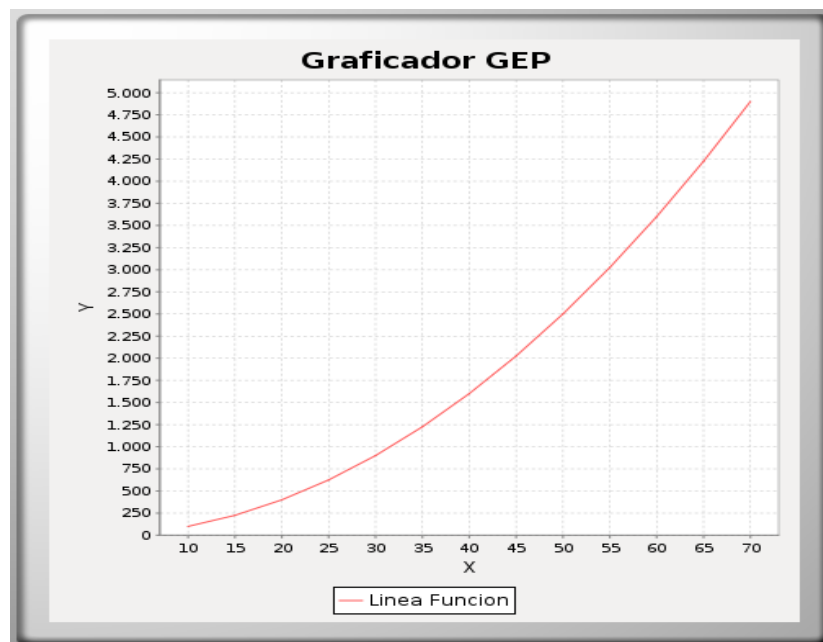
Prueba 4

1. Se ingresan los datos:

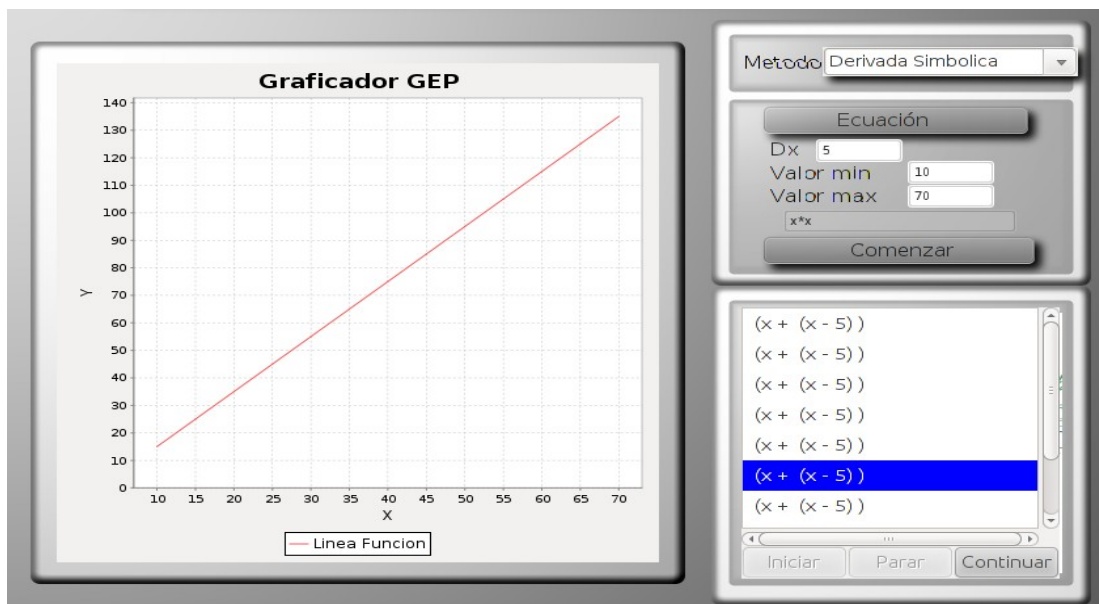


The screenshot shows a software window with a title bar. Inside, there is a dropdown menu labeled 'Metodo' with 'Derivada Simbolica' selected. Below this is a button labeled 'Ecuación'. Under the button, there are three input fields: 'Dx' with the value '5', 'Valor min' with the value '10', and 'Valor max' with the value '70'. Below these fields is a text input field containing 'x*x'. At the bottom is a button labeled 'Comenzar'.

2. Se muestra la gráfica de la función inicial:



3. El algoritmo encuentra una solución y al dar clic sobre ella se gráfica en pantalla:



6. CONCLUSIONES

La realización del proyecto permitió obtener muy valiosas conclusiones tanto en el área de los algoritmos genéticos, como en el desarrollo de objetos virtuales de aprendizaje. Asimismo, surgieron ideas y motivaciones para la creación de mas objetos de aprendizaje y continuar con el estudio de los algoritmos genéticos ya que veo un gran potencial en ellos en especial para resolver problemas de los cuales no sabemos como hallar la solución . A continuación se verán las conclusiones obtenidas y los trabajos futuros propuestos.

6.1 Conclusiones

- El algoritmo para los problemas de interpolación simbólica tarda tiempos cortos en el caso de ecuaciones sencillas, pero para ecuaciones complejas los tiempos para encontrar una solución pueden ser muy largos.
- Los resultados arrojados sin importar el tiempo de demora fueron acertados y también expresaron nuevas formas de ver las ecuaciones iniciales.
- La integración simbólica resuelta por medio de la programación por expresión genética presenta una alternativa muy apropiada ya que es muy difícil representar todas las reglas matemáticas que existen en la integración por medio de los algoritmos tradicionales.

- Los objetos virtuales de aprendizaje son una gran herramienta para masificación del conocimiento, pero no son muy conocidos y debido a estos no se han construido muchos en nuestro país.

- JavaFx es una gran herramienta para el desarrollo de aplicaciones visualmente agradable ya que cuenta con buenas librerías para la creación de aplicaciones para escritorio, aplicaciones para la web y aplicaciones para móviles pero no hay mucha información en la web sus libros se quedan cortos a veces y al parecer no hay muchos desarrolladores interesados en esta tecnología por esta razón no lo recomiendo mucho a la hora de hacer grandes aplicaciones debido a la poca popularidad en el mercado (esto hace que se encuentre pocas soluciones a la hora de enfrentarse a algunos problemas) pero si es una buena alternativa ya que se puede comunicar fácilmente con Java .

- El uso de herramientas libres facilita en desarrollo de los objetos virtuales de aprendizaje ya que se cuenta con la libertad de hacer lo que a mejor se tenga y luego decidir el tipo de licencia para la aplicación final, en esta caso de de licencia libre GNU lo que asegura es es un código abierto para otros desarrolladores puedan utilizar esta aplicación o su código para generar otras aplicaciones.

6.2 Trabajos futuros

- Para una mejor implementación de un objeto virtual se debe contar con un estándar para objetos virtuales de aprendizaje como SCORM, existen herramientas que permiten realizar este empaquetamiento pero no he encontrado ninguno para JavaFx por eso se puede realizar otra versión de la aplicación que sea un Applet de java para que este objeto pueda ser empaquetado.

- Crear un sitio web que contenga gran cantidad de objeto virtuales que sirva para almacenar este objeto virtual y muchos otros para algoritmos genéticos y también para otros temas de estudio.

- Mejorar la aplicación con la ayuda de animaciones y otros ayudas multimediales que hagan mas entretenido el uso de aplicación y mejoren el entendimiento del funcionamiento de la programación por expresión genética.

- Colocar en la aplicación otro tipo de problemas no matemáticos para mostrar el uso de la programación por expresión genética.

7. Referencias

[CFR] Cândida Ferreira, Gene Expression Programming: A New Adaptive Algorithm for Solving Problems , Complex Systems, Vol. 13, issue 2: 87-129, 2001

[WPD109] COLABORADORES DE WIKIPEDIA. Genetic programming. [en línea]. Wikipedia, La enciclopedia libre. 2010 [fecha de consulta: 20 de marzo de 2010]. Disponible en: <http://en.wikipedia.org/w/index.php?title=Genetic_programming&oldid=56260550>.

[LVIAR] Vida Artificial. Programación genética [en línea]. 2010 [fecha de consulta: 25 de agosto de 2010]. Disponible en: <http://vidaartificial.com/index.php?title=Programacion_genetica>

[AGBI] Angel Garcia Baños, COMPUTACIÓN EVOLUTIVA : Introducción 22 de enero de 2010

[AGBAG] Angel Garcia Baños, COMPUTACIÓN EVOLUTIVA : ALGORITMOS GENÉTICOS 22 de enero de 2010 .

[AGBPGE] Angel Garcia Baños, COMPUTACIÓN EVOLUTIVA : PROGRAMACIÓN GENÉTICA EVOLUCIÓN GRAMATICAL PROGRAMACIÓN POR EXPRESIÓN GENÉTICA 22 de enero de 2010

[KPG]Koza, J.R. *Programación genética: Al programar de computadoras por medio de la selección natural*, Prensa del MIT. ISBN 0-262-11170-5 1992

[DANE] DANE: Censo General 2005 [en línea]. 2010 [fecha de consulta: 20 de junio de 2010]. Disponible en: <http://www.dane.gov.co/daneweb_V09/index.php?option=com_content&view=article&id=307&Itemid=124>

[CFDAE] Carlos Fernando Latorre B. DISEÑO DE AMBIENTES EDUCATIVOS BASADOS EN NTIC

[JJDS][en línea]. 2010 [fecha de consulta: 12 de agosto de 2010]. Disponible en: <<http://jose-juan.computer-mind.com/jose-juan/Derivada-Simbolica.php>>

[WKIS] COLABORADORES DE WIKIPEDIA: Integración, Algoritmos simbólicos [en línea]. 2010 [fecha de consulta: 24 de agosto de 2010]. Disponible en: <http://es.wikipedia.org/wiki/Integraci%C3%B3n#Algoritmos_simb.C3.B3licos>

[WKADL] COLABORADORES DE WIKIPEDIA: Advanced Distributed Learning [en línea]. 2010 [fecha de consulta: 2 de octubre de 2010]. Disponible en: <http://es.wikipedia.org/wiki/Advanced_Distributed_Learning>

[WKS] COLABORADORES DE WIKIPEDIA: SCORM [en línea]. 2010 [fecha de consulta: 4 de octubre de 2010]. Disponible en: <<http://es.wikipedia.org/wiki/SCORM>>

[EIMS] Elearning: IMS [en línea]. 2010 [fecha de consulta: 5 de octubre de 2010]. Disponible en: <<http://www.utpl.edu.ec/elearningblog/?cat=5>>

[WKLOM] COLABORADORES DE WIKIPEDIA: Learning Object Metadata [en línea]. 2010 [fecha de consulta: 16 de octubre de 2010]. Disponible en: <http://es.wikipedia.org/wiki/Learning_Object_Metadata>

[OJFX] Oracle: JavaFX [en línea]. 2010 [fecha de consulta: 5 de febrero de 2010]. Disponible en: <<http://download.oracle.com/javafx/index.html>>

[GSGX] Gepsoft: GeneXproTools [en línea]. 2010 [fecha de consulta: 21 de octubre de 2010]. Disponible en: <<http://www.gepsoft.com/whatis.htm>>

[PPG] Python: PyGEP [en línea]. 2010 [fecha de consulta: 22 de octubre de 2010]. Disponible en: <<http://programming.itags.org/python/66131/>>