LABORATORIO VIRTUAL DE INTELIGENCIA DE ENJAMBRES

Ivan Fernando Quintero Rodríguez

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE AUTOMATICA Y ELECTRONICA
PROGRAMA INGENIERÍA ELECTRONICA
SANTIAGO DE CALI
2010

LABORATORIO VIRTUAL DE INTELIGENCIA DE ENJAMBRES

Ivan Fernando Quintero Rodríguez

Proyecto de grado para optar al título de Ingeniero Electrónico

Director
JESÚS ALFONSO LÓPEZ
Ingeniero Electricista
Magíster en Automática
Doctor en Ingeniería

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE AUTOMATICA Y ELECTRONICA
PROGRAMA INGENIERÍA ELECTRONICA
SANTIAGO DE CALI
2010

Nota de aceptación:

Aprobado por el Comité de Grado en cumplimiento de los requisitos exigidos por la Universidad Autónoma de Occidente para optar el título de Ingeniero Electrónico.

Diego Almario

Juan Carlos Mena

Santiago de Cali, Mayo 5 de 2010

Esta tesis está dedicada a mi padre Ivan Quintero Q.E.P.D, a mi madre Gladys Rodríguez y mi querida hermana por proporcionarme el apoyo y los conocimientos pilares para realización de este proyecto de vida.

AGRADECIMIENTOS

Agradezco primero a mi padre Ivan Quintero Q.E.P.D. por los fundamentos de vida y apoyo moral, económico y espiritual, que me brindo en vida, segundo a mi madre Gladys Rodríguez por brindarme proyección de vida desde muy pequeño por el amor brindado y sabiduría que siempre me ha acompañado.

Agradezco a mi hermana Ana Catalina Quintero y a mi sobrino Santiago Quintero, por brindarme todo el amor y ese apoyo que siempre necesite en los momentos más difíciles.

Agradezco a mi novia Isabel Cristina Moreno por estar a mi lado en las buenas y en las malas, por el amor que me brindas día a día que hace que mi vida tenga un mejor sentido.

A mí profesor Jesús Alfonso López, quien también ha sido mi tutor en este proyecto, por su admirable calidad humana, por sus enormes conocimientos que siempre con disposición diligente me entregó para disipar todas mis dudas, por ese tiempo que con paciencia y dedicación me entregó, dejando en mi vida una huella intangible, enseñándome que de nada valen los conocimientos adquiridos si no se complementan con la sencillez.

TABLA DE CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE CUADROS

LISTA DE FIGURAS

RESUMEN

En este trabajo se presenta un laboratorio virtual de inteligencia de enjambres implementado con la herramienta de desarrollo Easy Java Simulations. Las técnicas seleccionadas para ser implementadas en el laboratorio son la Optimización por Enjambre de Partículas (*Particle Swarm Optimization* -PSO-) y la Optimización por Enjambre de Bacterias (*Bacterial Swarm Foraging Optimizatio*n -BSFO-). El laboratorio consta de unas aplicaciones realizadas a manera de demos entre los que tenemos: la Optimización de una función de una sola variable, la optimización de una función de dos variables, y el control del nivel de un tanque por medio de estos dos algoritmos de optimización. Además cuenta con dos módulos genéricos unos para cada Algoritmos de optimización, los cuales son flexibles al cambio del usuario para así generar un aprendizaje más aplicativo de la inteligencia de enjambres.

INTRODUCCIÓN

En la naturaleza todos los seres vivos se enfrentan a problemas que deben resolver con éxito, como conseguir más luz del sol, o cazar una mosca. La Inteligencia Computacional es un área de investigación que tiene por objetivo el desarrollo de técnicas computacionales inspiradas en la observación de los mecanismos exitosos aplicados por la naturaleza para la solución de sus problemas. En la misma naturaleza se observa que los seres biológicos para resolver muchos de los problemas de la supervivencia diaria como el huir o enfrentar depredadores o en la búsqueda de alimento, los pueden afrontar de manera más eficiente si la realizan varios individuos en vez de uno.

Dentro de la inteligencia computacional, en años recientes ha surgido una tendencia inspirada en la maneras como colectividades de animales sea una colmena de abejas, una colonia de hormigas o una bandada de pájaros solucionan problemas claves de supervivencia denominada Inteligencia de enjambres (*SI* – Swarm intelligence). Algunas características básicas de la SI es que no existe un elemento de control central y su auto-organización.

El estudio de la inteligencia de enjambres está proporcionando ideas que pueden ayudar a los humanos a manejar sistemas complejos, desde enrutamiento de camiones hasta robots militares.

En este trabajo se tratara la creación de un laboratorio virtual de inteligencia de enjambres, el cual es un tema relativamente nuevo que actualmente esta siendo muy explotado para diversas aplicaciones. Este laboratorio se dispondrá en la red para su libre uso educacional.

1.0. OBJETIVOS

1.1. OBJETIVO GENERAL

Diseñar e implementar un laboratorio virtual de inteligencia de enjambres, con el fin de optimizar el proceso enseñanza-aprendizaje en el campo de la inteligencia bio-inspirada.

1.2. OBJETIVOS ESPECIFICOS

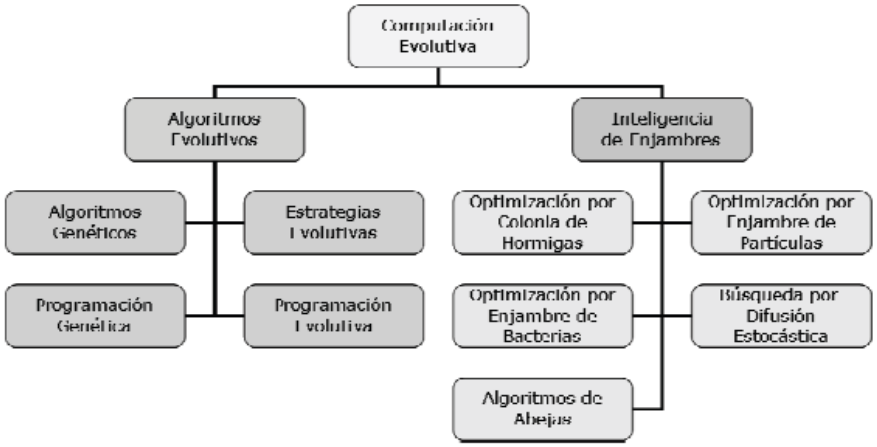
- Estudiar las diferentes técnicas existentes pertenecientes a inteligencia de enjambres.
- Estudiar y valorar la metodología de desarrollo de software que se implementará.
- Valorar o seleccionar la plataforma en que se implementará el laboratorio virtual de Inteligencia de Enjambres.
- Implementar una aplicación basada en Optimización Por Enjambre de Partícula (PSO – Particle Swarm Optimization) para optimizar funciones de una variable dependiente.
- Implementar una aplicación basada en Optimización Por Enjambre de Bacterias (BSFO – Bacteria Swarm Foragging Optimization) para optimizar funciones de una variable dependiente.
- Diseñar demostraciones interactivas donde se demostraría la aplicabilidad del PSO para resolver problemas de ingeniería.
- Diseñar demostraciones interactivas donde se demostraría la aplicabilidad del BSFO para resolver problemas de ingeniería.
- Demostrar las posibilidades de aprendizaje por medio de un laboratorio virtual, de una manera diferente a la educación tradicional que propicie una formación innovadora e integral de la inteligencia de enjambres.

2.0. MARCO TEÓRICO

2.1 INTELIGENCIA DE ENJAMBRES

La inteligencia de enjambres (Swarm Intelligencem, SI) es una rama de la inteligencia computacional (Ver figura 1) bio-inspirada en el comportamiento de animales sociales tales como: colonias de hormigas, colonias de abejas, crecimiento bacterial, bandadas de pájaros, cardúmenes de peces, etc.

Figura 1. Taxonomía de la Computación Evolutiva



Autor: MUÑOZ, Mario A. LOPEZ, Jesús A. CAICEDO, Eduardo F. Inteligencia de enjambres: sociedades para la solución de problemas (una revisión).

Un ejemplo más ilustrativo para explicar la inteligencia de enjambres podría ser: [1]
"*Suponga que tu y un grupo de amigos están en una misión para encontrar un tesoro. Tienes el conocimiento del área aproximada del tesoro, pero no sabes exactamente donde está localizado. Tú quieres ese tesoro, o al menos parte de él. Junto a tus amigos has aceptado en un mecanismo para compartirlo, para que todos los que hayan tomado parte en la búsqueda será recompensado, con la condición que la persona que encuentre el tesoro obtendrá una recompensa mayor a la de todos, y el resto serán recompensados en base a la distancia del tesoro en el momento que el primero encuentre el tesoro. Cada uno en el grupo tiene un detector de metal y puede comunicar la fuerza de la señal y su posición actual a los vecinos más cercanos. Por lo tanto cada persona sabe si su vecino está más cerca del tesoro de lo que él esta. ¿Qué acciones tomarías? Básicamente tienes dos elecciones.*

[1] ENGELBRECTH, Andries P. Computational Intelligence An Introduction. John Wiley & Sons, LTd. Segunda Edicion, 2007. 597p

1. *Ignorar a tus amigos, y buscar el tesoro sin ninguna información que tus amigos podrían proveerte. En este caso, si tú encuentras el tesoro, es todo tuyo. Sin embargo, si tú no lo encuentras primero, no tienes nada.*
2. *Hacer uso fe la información que percibes de tus amigos vecinos, y moverte en la dirección de tu amigo más cercano con la señal más fuerte. Al hacer uso de la información local, y actuando de acuerdo a ella, tu incrementas tus probabilidades de encontrar el tesoro, o al menos maximizar tu ganancia.* "

En el ejemplo anterior podemos observar claramente los beneficios de la cooperación en situaciones donde no tienes un conocimiento global del ambiente. Los individuos dentro del grupo interactúan para resolver el objetivo global mediante el intercambio de información a nivel local, que eventualmente se propaga a través de todo el grupo de tal manera que el problema se resuelve con más eficiencia que lo que una sola persona lo podría hacer.

Un sistema de inteligencia de enjambres típicamente está conformado por una población de agentes simples que interactúan localmente entre ellos y con el ambiente. En general los agentes están regidos por unas reglas simples de comportamiento y, aunque no existe una estructura de control centralizada definiendo el comportamiento de los agentes individuales, las interacciones locales entre dichos agentes conducen a la emergencia de un comportamiento global y complejo.

Los sistemas de SI pueden dividirse en un grupo de algoritmos, algunos grupos son: La Optimización basada en colonia de hormigas (Ant Colony Optimization– *ACO*), La Optimización por Enjambre de Partículas (Particle Swarm Optimization - *PSO*) y en Optimización Por Enjambre de Bacterias (Bacteria Swarm Foraging Optimization - *BSFO*).

2.2 OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS (PSO)

La Optimización por Enjambre de Partículas (*Particle Swarm Optimization,* PSO) es una prometedora metaheurística relativamente reciente introducida por James Kennedy y Russel Eberhat [2,3]. Se trata de un método de evolutivo inspirado en el comportamiento social de individuos dentro de enjambres en la naturaleza, como bandadas de pájaros o bancos de peces, los cuales siguen tres reglas propuestas

[2] KENNEDY, J. y EBERHART, R. Particle Swarm Optimization. En.: Memorias del 1995 IEEE International Conference on Neural Networks. 1995, p. 1942-1948

[3] KENNEDY, J. y EBERHART, R. Particle swarm optimization. Memorias del IEEE International Conference on Neural Networks, 1995, p. 1942–1498.

por Reynolds [4]: Separación donde cada agente trata de moverse lejos de su vecino si es que esta muy cerca, alineamiento donde cada agente sigue hacia la dirección promedio de sus vecinos y cohesión donde cada agente trata de ir hacia la posición promedio de sus vecinos como se muestra en la Fig. 2. Tal comportamiento social se basa en la transmisión del suceso de cada individuo a los demás individuos del grupo, lo cual resulta en un proceso sinergético que permite a los individuos satisfacer de la mejor manera posible sus necesidades más inmediatas, tales como la localización de alimentos o de un lugar de un lugar seguro. Para ello se modela un conjunto de soluciones alternativas o potenciales del problema como miembros del enjambre que se echan a volar en el espacio virtual de las posibles soluciones. En la planificación y optimización logística hay que adoptar tres tipos de decisiones: *estratégicas* o a largo plazo (cada varios meses o años), *tácticas* o a medio plazo (cada pocas semanas o meses) y *operativas* (varias veces en un día o una semana).

Las metaheurísticas son importantes en el apoyo de los tres tipos de decisiones, pero las características que se buscan en ellas son diferentes. Los tres tipos problemas más importante en logística, los problemas de localización, de rutas y de cargas, corresponden preponderantemente a decisiones estratégicas, tácticas operativas respectivamente. Los sistemas evolutivos inteligentes como la PSO son relevantes sobretodo en entornos estratégicos donde es frecuente encontrar elementos nuevos en el problema que hacen inviables procedimientos específicos ajustados a un modelo. En estos entornos, los procedimientos que como la PSO no son muy exigentes con las características del problema son cada vez más necesarios. Sin embargo, las pruebas iníciales de las estrategias de aplicación de la PSO pueden hacerse en instancias de los problemas estándares de localización como es el problema de la *p*-mediana.

Figura 2. Modelos del comportamiento dentro de los enjambre en la naturaleza

Fuente: Realizada por el Autor

[4] PASSINO, K. M.: Biomimicry of Bacterial Foraging for Distributed Optimization and.Control. IEEE. Control System Magazine, 2002, p. 52-67.

“La metaheurística PSO ha mostrado ser muy eficiente para resolver problemas de optimización de un sólo objetivo con rápidas tasas de convergencia haciendo atractiva la idea de su aplicación en la resolución problemas de optimización de múltiples objetivos”. [5]

2.2.1 Optimización Por Enjambre De Partículas Básica Los individuos en un enjambre de partículas siguen un simple comportamiento: Emular los éxitos individuales de sus vecinos y sus propios éxitos. El comportamiento colectivo que surge de este simple comportamiento descubre regiones óptimas en un espacio dimensional alto.

Un algoritmo PSO mantiene un enjambre de partículas, donde cada partícula representa una potencial solución. En analogía con los paradigmas de computación evolutiva, un enjambre es similar a una población, mientras que una partícula es similar a un individuo. En simples términos, las partículas están “volando” a través de un espacio de búsqueda multidimensional, donde la posición de cada partícula es ajustada de acuerdo a su propia experiencia y la de sus vecinos. Dejemos que $x_i(t)$ denote la posición de la partícula i en el espacio de búsqueda en un paso de tiempo t; a menos que se indique lo contrario, t denota pasos de tiempo discreto. La posición de la partícula es cambiada agregando una velocidad $v_i(t)$, a la posición actual,

$$x_i(t+1) = x_i(t) + v_i(t+1) \qquad [2.1]$$

Con $x_i(0) \sim U\ (x_{min}, x_{max})$.
Este es el vector de velocidad que maneja el proceso de optimización, y refleja ambos el conocimiento experimental de la partícula y la información socialmente cambiada de los vecinos de la partícula. El conocimiento experimental de la partícula es generalmente referido como el *componente cognoscitivo*, el cual es proporcional a la distancia de la partícula de su propia mejor posición (Referido a la *mejor posición personal* de la partícula) encontrada desde el primer paso. La información socialmente cambiada es referida como el *componente social* de la ecuación de velocidad.

Originalmente, dos algoritmos de PSO se han desarrollado los cuales difieren en el tamaño de sus vecinos. Estos dos algoritmos, nombrados como *gbest* y l*best* PSO.

2.2.1.1 Mejor PSO Global (gbest PSO) Para el mejor PSO global, los vecinos para cada partícula es enteramente el enjambre. La red social empleada por el

[5] KENNEDY, J. y EBERHART, R. Swarm Intelligence. Morgan Kaufmann Publishers, 2001.

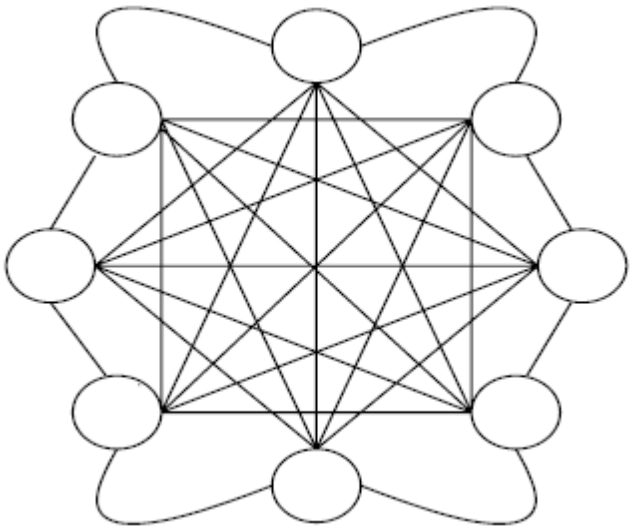
gbest PSO refleja la topología estrella (Ver figura 3). Para la topología vecindad estrella, el componente social de la actualización de velocidad de la partícula refleja la información obtenida de todas las partículas en el enjambre. En este caso, la información social es la mejor posición encontrada por el enjambre, referido como $\hat{y}(t)$.

Para gbest PSO, la velocidad de la partícula i es calculada como:

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_j(t) - x_{ij}(t)\right] \quad [2.2]$$

Donde $v_{ij}(t)$ es la velocidad de la particula en i en la dimensión $j = 1, \ldots, n_x$ en el paso de tiempo $t, x_{ij}(t)$ es la posición de la partícula i en la dimensión j en el paso de tiempo t, c_1 y c_2 son constantes de aceleración positiva usadas para dimensionar la contribución de la cognocion y los componentes sociales respectivamente, y $r_{ij}(t), r_{2j} \sim U(0,1)$ son valores aleatorios en el rango [0,1], muestreado de una distribución uniforme. Estos valores aleatorios introducen un elemento estocástico al algoritmo.

Figura 3. Topología Estrella usada por el Mejor PSO global



Fuente: ENGELBRECTH, Andries P. Computational Intelligence An Introduction.

La mejor posición personal, y_i, asociada con la partícula i es la mejor posición que la partícula ha visitado desde el primer paso de tiempo. Considerando los problemas de minimización, la mejor posición personal en el siguiente paso de tiempo, $t + 1$, es calculada como

$$y_i(t+1) = \begin{cases} y_i(t) & if\ \ f\big(x_i(t+1)\big) \geq f(y_i(t)) \\ x_i(t+1) & if\ \ f\big(x_i(t+1)\big) < f(y_i(t)) \end{cases} \quad [2.3]$$

Donde $f : \mathbb{R}^{n_x} \rightarrow \mathbb{R}$ es la función de optimización. La función de optimización mide que tan cerca esta la solución correspondiente de la óptima, i.e. la función de optimización cuantifica la rendimiento, o calidad, de una partícula (o solución).

La mejor posición global, $\hat{y}(t)$, en un paso de tiempo t, está definido como:

$$\hat{y}(t) \in \{y_0(t), \dots, y_{n_s}(t)\} \mid f\big(\hat{y}(t)\big) = \min\{f\big(y_0(t)\big), \dots, f(y_{n_s}))\} \quad [2.4]$$

Donde n_s es el número total de partículas en el enjambre. Es importante notar que la definición en la ecuación [2.4] declara que $\hat{y}$ es la mejor posición descubierta por cualquier partícula hasta el momento – esta es usualmente calculada como la mejor posición personal. La mejor posición global también puede ser seleccionada de las partículas en el enjambre actual, en este caso. [6]

$$\hat{y}(t) = \min\{f\big(y_0(t)\big), \dots, f(y_{n_s}))\} \quad [2.5]$$

El *gbest* PSO esta resumido en el cuadro 1

Cuadro 1. Algoritmo tipo gbest PSO

Crea e inicia un enjambre n_x-dimension;
MIENTRAS no se cumple la condicion de parada
    PARA cada particula i = 1,…, n_s hacer
        // Fijar la mejor posición personal
        SI $f(x_i) < f(y_i)$ entonces
            $y_i = x_i$;
        FIN SI
        // Fijar la mejor posición global
        SI $f(y_i) < f(\hat{y})$ entonces
            $\hat{y} = y_i$;
        FIN SI
    FIN PARA
    PARA $cada\ particula\ i = 1, \dots, n_s$ Hacer
        Actualizar la velocidad usando la ecuación [2.2];
        Actualizar la posición usando la ecuación [2.1];
    FIN PARA
FIN MIENTRAS

Fuente: Realizada por el Autor

[6] ENGELBRECTH, Andries P. Computational Intelligence An Introduction. John Wiley & Sons, LTd. Segunda Edicion, 2007. 597p.

2.2.1.2 Mejor PSO Local (lbest PSO) El mejor PSO local, o lbest PSO, usa una topología red social de anillo (Ver figura 4), donde los vecinos más pequeños son definidos por cada partícula. El componente social refleja intercambio de información entre el vecindario de la partícula, reflejando conocimiento local del ambiente. Con referencia a la ecuación de la velocidad, la contribución social a la velocidad de la partícula es proporcional a la distancia entre una partícula y la mejor posición encontrada por el vecindario de las partículas. La velocidad es calculada como.

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_{ij}(t) - x_{ij}(t)\right] \quad [2.6]$$

Donde $\hat{y}_{ij}$ es la mejor posición, encontrada por los vecinos de la partícula i en la dimensión j. La mejor posición local de la partícula, $\hat{y}_{ij}$, i.e. la mejor posición encontrada en el vecindario N_i, es definida como.

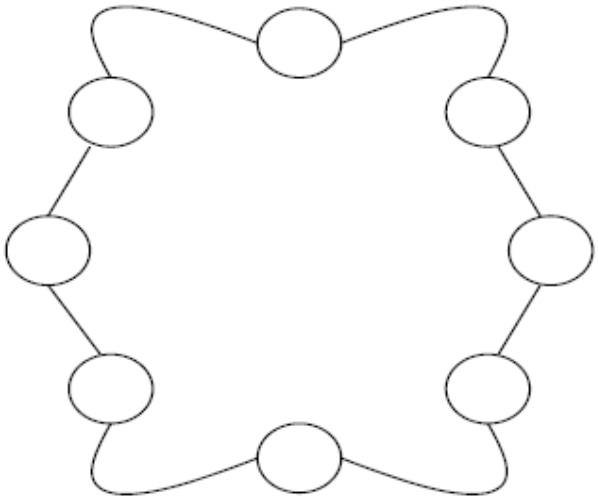
$$\hat{y}_i(t+1) \in \left\{N_i | f\left(\hat{y}_i(t+1)\right) = min\{f(x)\}, \forall x \in N_i\right\} \quad [2.7]$$

Con los vecinos definidos como

$$N_i = \left\{y_{i-nN_i}(t), y_{i-nN_i+1}(t), \dots, y_{i-1}(t), y_i(t), y_{i+1}(t), \dots, y_{i+nN_i}(t)\right\} \quad [2.8]$$

Para un vecindario de tamaño n_{N_i}. La mejor posición local también será referida como la mejor posición del vecindario.

Figura 4. Topología Anillo usada por el Mejor PSO local



Fuente: ENGELBRECTH, Andries P. Computational Intelligence An Introduction.

Es importante notar que para el PSO básico, las partículas dentro de un vecindario no tienen relación entres si. La selección de vecindarios es basándose en los índices de las partículas. Sin embargo, las estrategias se han desarrollado donde las vecindades son formadas basadas en la similitud espacial.

Hay principalmente dos razones del porque las vecindades son basadas en los índices de las partículas son preferidas:

1. Esta es computacionalmente económico, desde que no es requerido el orden espacial de las partículas. Para aproximaciones donde la distancia entre partículas es usada para formar el vecindario, es necesario calcular la distancia Euclidiana entre todos los pares de partículas, el cual es de complejidad $0(n_s^2)$.
2. Ayuda a promover el esparcimiento de la información recordando buenas soluciones a todas las partículas, independientemente de su ubicación actual en su espacio de búsqueda.

También debería ser notado que las vecindades se superponen. Una partícula toma parte como miembro de un número de vecindarios. Esta interconexión de vecindarios también facilita el intercambio de información entre las vecindades, y asegura que el enjambre converja a un solo punto, nombrado como la mejor partícula global. El gbest PSO es un caso especial de el lbest PSO con $n_{N_i} = n_s$.

El *gbest* PSO esta resumido en el cuadro 2

Cuadro 2. Algoritmo tipo lbest PSO

Crea e inicia un enjambre n_x-dimension;
MIENTRAS no se cumple la condición de parada
    PARA cada partícula i = 1,…, n_s hacer
        // Fijar la mejor posición personal
        SI $f(x_i) < f(y_i)$ entonces
            $y_i = x_i$;
        FIN SI
        // Fijar la mejor posición global
        SI $f(y_i) < f(\hat{y}_i)$ entonces
            $\hat{y} = y_i$;
        FIN SI
    FIN PARA
    PARA $cada\ particula\ i = 1, \dots, n_s$ Hacer
        Actualizar la velocidad usando la ecuación [2.6];
        Actualizar la posición usando la ecuación [2.1];
    FIN PARA
FIN MIENTRAS

Fuente: Realizada por el Autor

***2.2.1.3 Gbest* Contra *Lbest* PSO [4]** Las dos versiones de PSO discutidas arriba son similares en el sentido que el componente social de la actualización de la velocidad hace ambos moverse hacia la mejor posición global de la partícula. Esto es posible para el lbest PSO debido a la superposición de vecindarios.

Hay dos principales diferencias entre las dos aproximaciones con respecto a sus características de convergencia;

- Debido a las partículas de mayor interconectividad del *gbest* PSO, este converge más rápido que el lbest PSO. Sin embargo, esta convergencia rápida viene con el costo de menor diversidad que el *lbest* PSO.
- Como consecuencia de su mayor diversidad (El cual resulta en mayores partes de búsqueda en el espacio cubierto), el *lbest* PSO es menos susceptible a ser atrapado en mínimos locales. En general (dependiendo de el problema), la estructura del vecindario tal como la topología de anillo usada en *lbest* PSO mejora el rendimiento.

2.3 OPTIMIZACIÓN POR ENJAMBRE DE BACTERIAS (BSFO)

La optimización por enjambre de bacterias representa una aproximación diferente a la búsqueda de valores óptimos en funciones no lineales desarrollado por Passino [7], basado en el comportamiento quimiotáctico de la E. Coli. Este es tal vez el microorganismo mas comprendido, ya que su comportamiento y estructura genética están bien estudiados. Esta consta de una capsula que incluye sus órganos, y flagelos que utiliza para su locomoción; posee capacidad de reproducirse por división y también es capaz de intercambiar información genética con sus congéneres. Además, puede detectar alimento y evitar sustancias nocivas, efectuando un tipo de búsqueda aleatoria, basado en dos estados de locomoción, el desplazamiento y el giro. En la posición inicial ellas miden la concentración de comida y entonces giran para tomar una dirección al azar y nada hacia una distancia fija y miden la concentración ahí como se muestra en la Fig. 5. La decisión de permanecer en uno de estos dos estados se debe a la concentración de nutrientes o sustancias nocivas en el medio. Este comportamiento se denomina quimiotaxis. Si la concentración es mayor, luego dan un paso en ese sentido. Cuando la concentración en la siguiente posición es menor que la posición anterior ellas giran para encontrar una nueva posición y nadar hacia esta nueva posición. Este proceso lleva a cabo un cierto número de pasos, los cuales están limitados por el tiempo de vida de la bacteria. Al final de su vida las bacterias que han reunido una buena salud que se encuentran en una mayor concentración estas se dividen en dos células. Así en el próximo paso

[7] PASSINO, Kevin. Distributed Optimization and Control Using Only a Germ of Intelligence. Memorias del 2000 IEEE International Symposium on Intelligent Control, 2000, p. 5-13.

reproductivo la siguiente generación de bacterias empiezan desde una posición sana. La mejor mitad reproduce la siguiente generación mientras que la peor mitad muere. Este paso de reproducción también se lleva a cabo un número fijo de veces.

2.3.1. Forrajeo Bacterial: E. Coli La bacteria E. Coli posee una membrana de plasma, pared celular, y una capsula que contiene el citoplasma y el núcleoide. Los pilus son usados para un tipo de transferencia de genes hacia otra bacteria E. coli, y un flagelo usado para la locomoción. La célula tiene acerca de 1µm de diámetro y 2 µm en longitud. La célula E. coli pesa acerca de 1 picogramo y tiene alrededor de 70% de agua.

La bacteria E. Coli es probablemente el microorganismo mejor entendido. Su genoma entero ha sido secuenciado; contiene 4,639,221 de "letras" A, C, G y T – Adedesina, Citosina, Guanina y Timina- organizados en un total de 4,288 genes.

Cuando la E. coli crece, se hace más largo, entonces se divide en el medio en dos "hijas." Recibe suficiente alimento y se mantiene a la temperatura del intestino humano de 37ºC, E. coli puede sintetizar y replicar todo lo que necesita para hacer una copia de sí misma en acerca de 20 minutos; por lo tanto el crecimiento de una población de bacterias es exponencial con un tiempo relativamente pequeño.

La bacteria E. coli tiene una sistema de control que permite buscar por comida y tratar de evitar sustancias nocivas.

2.3.2. Nadando y Rotando por medio del flagelo La locomoción se logra a través de una serie de flagelos relativamente rígidos que permiten a la bacteria a nadar a través de cada uno de ellos que giran en la misma dirección en alrededor de 100 a 200 revoluciones por segundo. Cada flagelo es una es una hélice zurda configurada de manera que a medida que la base del flagelo gira en sentido anti-horario, visto desde el extremo libre del flagelo mirando hacia la célula, se produce una fuerza en contra de la bacteria por lo que empuja a la célula.

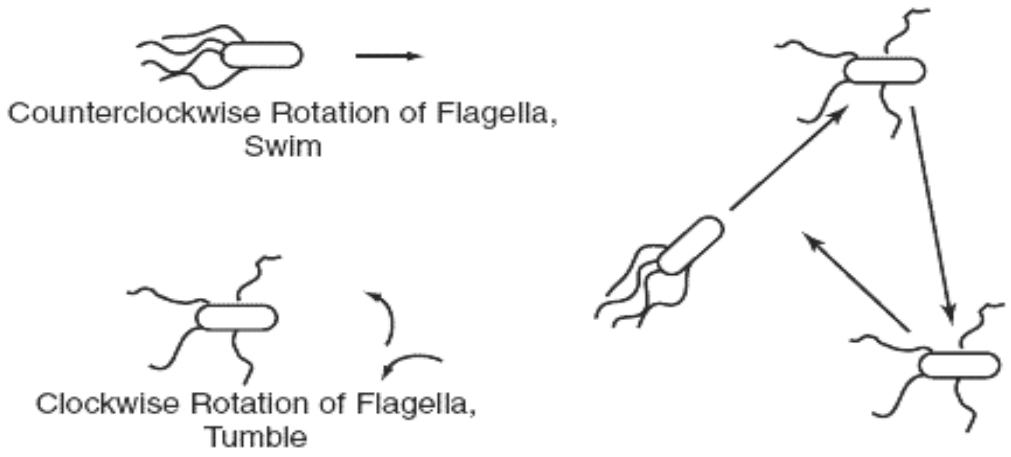
Una bacteria E.coli puede moverse en dos diferentes direcciones; puede correr (Nadar por un periodo de tiempo) o puede rotar, y alternar entre estos dos modos de operación toda su vida.

Si el flagelo rota en sentido del reloj, cada flagelo empuja en la célula, y el efecto total es que cada flagelo opera relativamente independientemente de los otros, y entonces la bacteria "rota". Ver figura 5

Estos patrones de movimiento (Llamados "taxis") que la bacteria genera en la presencia de químicos atrayentes y repulsivos son llamados *quimiotaxis.*

Generalmente, como grupo, las bacterias trataran de encontrar comida y evitar fenómenos dañinos, y cuando son vistas bajo un microscopio, se tendrá la sensación de que un tipo de comportamiento inteligente ha emergido, debido a que ellas parecen moverse intencionalmente en grupo.

Figura. 5 Desplazamiento, Giro, y comportamiento quimiostatico de la bacteria E.Coli.



Fuente: PASSINO, K. M.: Biomimicry of Bacterial Foraging for Distributed Optimization and.Control.

2.3.3 Eventos de Eliminación y Dispersión Es posible que el ambiente local donde una población de bacterias vive cambie paulatinamente o de repente debido a alguna otra influencia. Estos eventos pueden ocurrir de manera que las bacterias en una región son asesinadas o grupo se dispersa en un nuevo medio ambiente.

Por ejemplo, significantes aumentos de la temperatura local puede matar una población de bacterias que están actualmente en una región con alta concentración de nutrientes. O puede ser que el agua o algún otro animal muevan las poblaciones de bacterias de un lugar a otro en el ambiente.

Tras largos periodos de tiempo, dichos eventos han esparcido varios tipos de bacteria virtualmente dentro de cada parte de nuestro ambiente.

¿Cuál es el efecto de los eventos de eliminación y dispersión en la quimiotaxis? Ellas tienen el efecto de la posible destrucción de los progresos quimiotacticos, pero ellas también tienen el efecto de asistencia al progreso de la quimitaxis, desde que la dispersión puede haber ubicado la bacteria cerca de buenas fuentes

de comida. Desde una perspectiva general, eliminación y dispersión son partes del comportamiento móvil.

2.3.4 Optimización por enjambre Forrajea de la Bacteria E.Coli Supongan que queremos encontrar el minimo de $J(\theta)$, $\theta \in \mathbb{R}^p$, donde nosotros no tenemos medidas o una descripción analítica de el gradiente $\nabla J(\theta)$. Aquí, nosotros usamos ideas del forrajeo de la bacteria para resolver este problema de optimización no gradiente.
Primero, suponga que θ es la posición de la bacteria y $J(\theta)$ representa los efectos combinados de atracción y repulsión del ambiente, con, por ejemplo, $J(\theta) < 0$, $J(\theta) = 0$ y $J(\theta) > 0$ representando que la bacteria en el lugar θ es rico en nutrientes, neutral, y ambiente nocivo, respectivamente. Básicamente, quimiotaxis es un comportamiento forrajeo que implementa un tipo de optimización donde la bacteria trata de trepar a una concentración mayor de nutrientes, evita las sustancias nocivas, y busca por vías de un medio neutro. Esto implementa un tipo de una caminata parcialmente al azar. [8]

2.3.5 Quimiotaxis, Enjambre, Reproducción, Eliminación y Dispersión Definir un paso quimiotactico para ser un giro seguido por un giro o un giro seguido por un nado. Dejamos a j ser el índice para los pasos quimiotácticos. Dejamos a k ser el índice para los pasos de reproducción. Dejamos a l ser el índice para el evento de eliminación-dispersión. Dejamos representar la posición de cada miembro en la población de la bacteria S al $j - esimo$ paso quimiotáctico. Aquí, dejamos $J(i,j,k,l)$ denotar el costo en el lugar de la $i - esima$ bacteria $\theta^i(j,k,l) \in \mathbb{R}^p$. Tenga en cuenta que nos dirigiremos indistintamente a J como un "costo" y como una superficie de nutrientes. Para las poblaciones actuales de bacterias, S pude ser muy largo, pero $p = 3$.

$$P(j,k,l) = \{\theta^i(j,k,l) \mid i = 1,2,\dots,S\} \qquad [2.9]$$

Sea $N_c$ser la longitud de el tiempo de vida de la bacteria según lo medido por el número de pasos quimiotácticos toman durante la vida. Sea $C(i) > 0, i = 1,2,\dots,S$, denota un tamaño básico de un paso quimiotáctico que usaremos para definir la longitud de pasos durante la simulación. Para representar un giro, una unidad de longitud con dirección al azar, dice $\emptyset(j)$, es generado: Esto será usado para definir la dirección de movimiento después de un giro. En particular, sea

$$\emptyset^i(j+1,k,l) = \emptyset^i(j,k,l) + C(i)\emptyset(j) \qquad [2.10]$$

[8] PASSINO, K. M.: Biomimicry of Bacterial Foraging for Distributed Optimization and.Control. IEEE. Control System Magazine, 2002, p. 52-67.

Asi que $C(i)$ es el tamaño de los pasos tomados en la dirección al azar especificada por el giro. IS en $\emptyset^i(j+1,k,l)$ el costo $J(i,j+1,k,l)$ es mejor (mas bajo) que en $\emptyset^i(j,k,l)$, entonces otro paso de tamaño $C(i)$ en esta misma dirección será tomada, y de nuevo si este paso resulta en una posición con un mejor costo que el paso previo, entonces otro paso es tomado. Este nado es seguido tanto como continúe reduciendo el cotos, pero solo hasta un máximo número de pasos, N_s. Este representa que la célula tendera a seguirse moviendo si está dirigida a entornos cada vez más favorables.[13]

2.3.6 Algoritmo de Optimización por Forrajeo de Bacterias Para la inicialización, debes elegir $p, S, N_c, N_s, N_{re}, N_{ed}, P_{ed,}$ y el $C(i), i = 1,2,\dots,S$. Los valores iníciales para la $\theta^i, i = 1,2,\dots,S$. Deben ser elegidos.

Escogiendo estos para estar en aéreas donde un valor optimo tiene buenas probabilidades de ser elegido.

El algoritmo que modela la población bacterial quimitaxis, enjambre, reproducción, eliminación, y dispersión esta dado (inicialmente, $j = k = l = 0$). Para el algoritmo, cabe anotar que se actualiza a la θ^i automáticamente resultando en la actualización de P.

1) Ciclo Eliminación – Dispersión: $l = l + 1$
2) Ciclo Reproducción: $k = k + 1$
3) Ciclo Quimiotactico: $j = j + 1$
   a) Para $i = 1,2,\dots,S$, tomar una paso quimiotáctico para la bacteria i.
   b) Calcular $J(i,j,k,l)$. Sea $J(i,j,k,l) = J(i,j,k,l) + J_{cc}\left(\theta^i(j,k,l), P(j,k,l)\right)$.
   c) Sea $J_{last} = J(i,j,k,l)$ gravar este valor ya que podemos encontrar un mejor valor en el camino.
   d) Giro: Genera un vector al azar $\Delta(i) \in \mathbb{R}^p$ con cada elemento $\Delta_m(i), m = 1,2,\dots,p$, un valor al azar entre [-1,1].
   e) Mover: Sea

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}.$$

   Este resulta en un paso de tamaño $C(i)$ en la dirección de el giro de la bacteria i.
   f) Calcular $J(i,j+1,k,l)$, y entonces sea $J(i,j+1,k,l) = J(i,j+1,k,l) + J_{cc}\left(\theta^i(j+1,k,l), P(j+1,k,l)\right)$.
   g) Nada
   I - Sea $m = 0$.
   II – Mientras $m < N_s$.

- Sea $m = m + 1$.
- Si $J(i, j+1, , k, l) < J_{last}$ (si lo está haciendo mejor), Sea $J_{last} = J(i, j+1, , k, l)$ y sea

$$\theta^i(j+1, k, l) = \theta^i(j+1, k, l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$$

Y usa este $\theta^i(j+1, k, l)$ para calcular el *nuevo* $J(i, j+1, k, l)$

- Si no, sea $m = N_s$. Final del Mientras

h) Ve a la siguiente Bacteria $(i+1)$ si $i \neq S$ (i.e. ve al punto b a procesar la siguiente bacteria)

4) Si $j < N_c$, ve al paso 3.
5) Reproducción
   a) Para k dado y l, y por cada $i = 1,2, \ldots, S$, Sea.

$$J^i_{healt\,h} = \sum_{j=1}^{N_c+1} J(i, j, k, l)$$

Sea la salud de la bacteria i (una medida de cuantos nutrientes tuvo en toda su vida y como el triunfo fue evitando sustancias nocivas). Ordenar la bacteria y los parámetros quimiotácticos $C(i)$ en orden ascendientes del costo de $J_{healt\,h}$.

   b) La bacteria S_r con el valor más grande de $J_{healt\,h}$ muere y la otra bacteria S_r con el mejor valor se divide (y las copias que son hechas son colocadas en la misma posición que sus padres)

6) Si $k < N_{re}$, ve al paso 2.
7) Eliminación –Dispersión: Para $i = 1,2, \ldots, S$, con probablemente p_{ed}, eliminar y dispersar cada bacteria (esto mantiene la población de bacterias constante).
8) Si $l < N_{ed}$, entonces ve al paso 1, de otra forma terminar

En la técnica de optimización podemos tomar la variable que queremos optimizar como la posición de la bacteria en el plano de búsqueda (el plano donde la bacteria puede moverse). Las especificaciones tales como el número de pasos reproductivos, el número de pasos quimiotáticos los cuales consisten en el desplazamiento (nado) y el giro, la distancia de desplazamiento, el máximo desplazamiento permitido en una determinada dirección son dados para un problema en particular entonces la variable puede ser optimizada utilizando este método (BSFO) [9].En el Cuadro 3 podemos apreciar el algoritmo general de BSFO.

9 PASSINO, K. M.: Biomimicry of Bacterial Foraging for Distributed Optimization and.Control. IEEE. Control System Magazine, 2002, p. 52-67.

Cuadro 3. Optimización por enjambre de bacterias

```
Inicializar la población aleatoriamente
Definir para cada individuo un C(i)
PARA l=1 HASTA Ned HACER
        PARA k=1 HASTA Nre HACER
                PARA j=1 HASTA Nc HACER
                        Calcular el Costo(i,j,k,l) Incluyendo Ccc
                        Cultimo = Costo(i,j,k,l)
                        Realizar Desplazamiento
                        Calcular Nuevo  Costo(i,j,k,l)
                        m=0
                        MIENTRAS m<Ns
                        m=m+1
                        SI Costo(i,j,k,l) < Cultimo
                        Cultimo= Costo(i,j,k,l)
                        Realizar Desplazamiento
                        Calcular Nuevo  Costo(i,j,k,l)
                        SINO
                                M=Ns
                        FIN SI
                        FIN MIENTRAS
                FIN PARA
        Calcular la salud de cada bacteria
        Eliminar las bacterias inadecuadas
        FIN PARA
        Dispersar/Eliminar con probabilidad Ped
FIN PARA
```

Fuente: Realizada por el Autor.

2.4. OPTIMIZACION POR COLONIA DE HORMIGAS

La optimización basada en colonias de hormigas [10] representa en forma artificial el comportamiento de las colonias de hormigas en la naturaleza. Las hormigas son insectos sociales que viven en colonias y que debido a su colaboración mutua son capaces de mostrar comportamientos complejos y realizar tareas difíciles desde el punto de vista de una hormiga individual.

Un aspecto interesante del comportamiento de las hormigas es su habilidad para encontrar los caminos más cortos entre su hormiguero y las fuentes de alimento;

[10] DORIGO, M. MANIEZZO, V. COLORNI, A. “Ant System: Optimization by a Colony of Cooperative Agents”. En: IEEE Transactions on System, Man and Cybernetics- Part B Vol. 26 Nº 1, p. 29-41. 1996.

cuando estas se mueven entre el hormiguero y la fuente de alimento van depositando una sustancia química llamada feromona.

Si las hormigas en su recorrido no encuentran ningún rastro de feromona moverán de manera aleatoria, pero cuando existe feromona depositada en la ruta, esta tendrá una mayor tendencia a ser recorrida. Este mecanismo permite a las hormigas encontrar el camino más corto entre el nido y la fuente [11].

Según transcurre el tiempo y mientras que las hormigas recorren los caminos más transitados en su búsqueda de comida, cada uno de los caminos seleccionados va recibiendo una cantidad superior de feromona. Finalmente en los caminos más cortos habrá un rastro de feromona ligeramente superior y, por lo tanto, las decisiones de las siguientes hormigas estarán dirigidas en mayor medida a dichos caminos.

Este proceso finaliza haciendo que la probabilidad de que una hormiga escoja el camino más corto aumente progresivamente y que al final el recorrido de la colonia converja al más corto de todos los caminos posibles. Esta convergencia se complementa con la acción del entorno natural que provoca que la feromona se evapore transcurrido un cierto tiempo, por lo que los caminos menos prometedores, al perder progresivamente la feromona, son visitados cada vez por menos hormigas.

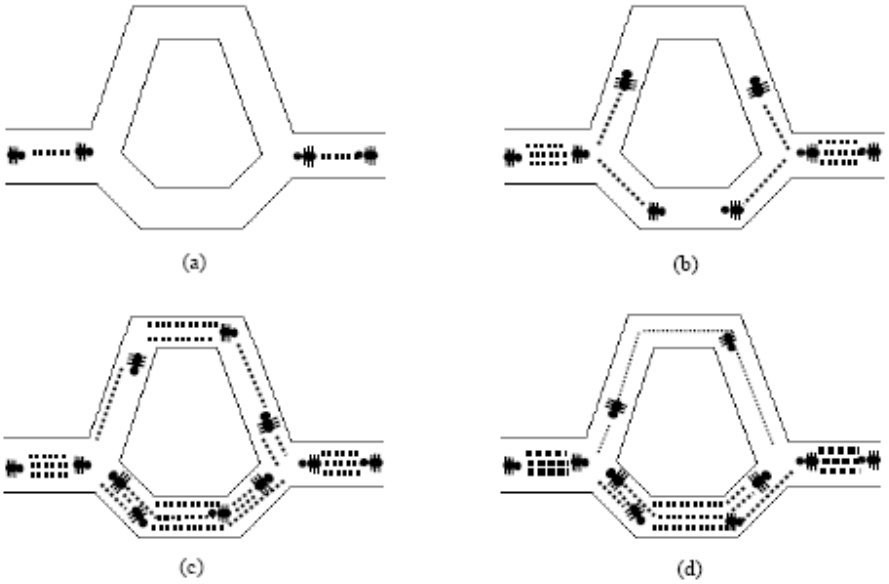
Así, los algoritmos basados en colonias de hormigas son esencialmente algoritmos constructivos, en cada ciclo del algoritmo, cada hormiga construye una solución al problema recorriendo los arcos de un grafo [12, 13, 14].

En la figura 6. el cual es una grafica de un experimento con hormigas reales. Donde en la figura 6a podemos apreciar como las hormigas llegan a un punto donde tienen que decidir por uno de los caminos que se les presenta, lo que resuelven de manera aleatoria. En consecuencia, la mitad de las hormigas se dirigirán hacia un extremo y la otra mitad hacia el otro extremo, como ilustra la figura 6b. Como las hormigas se mueven aproximadamente a una velocidad constante, las que eligieron el camino mas corto alcanzaran el otro extremo mas rápido que las que tomaron el camino más largo, quedando depositado mayor cantidad de feromona por unidad de longitud, como ilustra la figura 6c. La mayor

[11] DORIGO, M. DI CARO, G. GAMBARDELLA, L. “Ant Algorithms for Discrete Optimization”. En: Artificial Life. Vol. 5 Nº 2, p. 137-172. 1999.
[12] DORIGO, M. DI CARO, G. GAMBARDELLA, L. “Ant Algorithms for Discrete Optimization”. En: Artificial Life. Vol. 5 Nº 2, p. 137-172. 1999.
[13] DORIGO, M. MANIEZZO, V. COLORNI, A. “Ant System: Optimization by a Colony of Cooperative Agents”. En: IEEE Transactions on System, Man and Cybernetics- Part B Vol. 26 Nº 1, p. 29-41. 1996.
[14] DORIGO, M. MANIEZZO, V. COLORNI, A. “Possitive feedback as a search strategy”. Technical Report Nº 91-016. Dipartimento di Elettronica Politecnico di Milano. Italy. 1991.

densidad de feromonas depositadas en el trayecto mas corto hace que este sea más deseable para las siguientes hormigas y por lo tanto la mayoría elige transitar por él. Considerando que la evaporación de la sustancia química hace que los caminos menos transitados sean cada vez menos deseables y la realimentación positiva en el camino con más feromonas, resulta claro que al cabo de un tiempo casi todas las hormigas transiten por el camino mas corto.

Figura 6. Experimento con hormigas reales al buscar el camino más corto



Fuente: DORIGO, M. DI CARO, G. GAMBARDELLA, L. Ant Colonies for the Traveling Salesman Problem.

2.4.1 Algoritmo ACO En un experimento hecho por Goss [15] en donde dispuso de dos puentes, uno significativamente más largo que el otro. Al inicio del experimento las hormigas elegían uno de los 2 puentes sin ninguna influencia, pero las que elegían el puente mas corto regresaban mas rápido y es así como ese camino obtenía mas feromona que el otro. A este fenómeno se le llama autocatalisis y es como explotan la retroalimentación positiva para encontrar la ruta mas corta entre la fuente de alimento y su nido. Este es el modelo de comunicación de las hormigas, también conocido como modelo de comunicación por el medio ambiente o “stigmergic” en ingles.

A partir de los resultados obtenidos con el experimento se desarrollo un modelo para explicar el comportamiento observado en el experimento del puente binario.

[15] DORIGO, M. GAMBARDELLA, L.. “Ant Colonies for the Traveling Salesman Problem”. En: BioSystems, Vol. 43, p. 73-81. 1997.

Asumiendo que después de t unidades de tiempo del inicio del experimento, y que m_1 hormigas usaron el primer puente y que m_2 hormigas el segundo, la probabilidad p_1 de que la $(m+1)$ hormiga elija el primer puente está dada por:

$$p_{1(m+1)} = \frac{(m_1+k)^h}{(m_1+k)^h + (m_2+k)^h} \quad [2.11]$$

$$P_{2(m+1)} = 1 - P_{1(m+1)} \quad [2.12]$$

Donde k y h son parámetro del modelo para ajustarlo a los datos experimentales.

El modelo anterior se utilizo como inspiración para crea algoritmos con hormigas artificiales que simulan el concepto de feromona modificando variables-feromona asociadas a estados del problema que vistan al construir una solución al problema de optimización.

Las características de la comunicación por medio del ambiente (stigmergic) se llevan a agentes artificiales como:

- Asociando variables de estado a diferentes estados del problema
- Dando a los agentes solo el acceso local a estas variables

Otro Aspecto del comportamiento de las hormigas que se puede llevar a las hormigas artificiales es el acoplamiento entre el mecanismo auto-catalítico y la evaluación implícita de soluciones (las rutas más cortas, que corresponden a soluciones con menor costo en caso de las hormigas artificiales), se encuentran más rápido que las largas y por eso reciben su refuerzo de feromona más rápido.

La comunicación por medio del ambiente (stigmergy), la evaluación de solución implícita y el comportamiento autocatalítico dieron lugar a ACO.

Cosas en común entre hormigas reales y artificiales:

- Ambas colonias de hormigas se componen de una población de individuos que trabajan juntos para alcanzar una cierta meta.
- Una colonia es una población de simples e independientes agentes asíncronos que cooperan para encontrar una buena solución a el problema
- En caso de hormigas reales la meta es encontrar comida, en caso de hormigas artificiales, la meta es encontrar la solución a un problema dado de optimización
- Una sola hormiga es capaz de encontrar solución a un problema pero solo la cooperación entre muchos individuos a través de la comunicación por medio del ambiente (stigmergy) les permite encontrar buenas soluciones

Las diferencias entre las hormigas reales y las artificiales se dan por que las hormigas reales depositan la sustancia química llamada feromona y la artificiales utilizan variables con valores numéricos que simulan la feromona. Los caminos de feromona de las hormigas reales se simulan con caminos de feromona artificiales que son una secuencia de valores de feromona asociados con estados del problema. En la vida real la feromona física se evapora y de esta manera las hormigas olvidan el pasado y se enfocan en nuevas direcciones prometedoras.

Las hormigas artificiales crean soluciones secuenciales al ir de un estado a otro, van hacia los estados disponibles tomando una decisión a la vez.
Diferencias entre hormigas reales y artificiales

- Las hormigas artificiales viven en un mundo discreto, se mueven secuencialmente a través de un conjunto finito de estados del problema
- Las actualizaciones de feromona (depósitos y evaporación de feromona) no se realiza de la misma manera. Algunas veces la actualización de feromona se hace solo para algunas de las hormigas artificiales y con frecuencia solo después que se construyo una solución.
- Algunas implementaciones de hormigas artificiales usan mecanismos adicionales que no existen en hormigas reales: mirar hacia adelante, búsqueda local, backtracking, etc.

2.4.2 La Meta-heurística de la Optimización de Colonias de hormigas ACO se ve como una meta-heurística de optimización combinatoria (Combinatorial Optimization Problem COP). Para encontrar la solución al problema se define el modelo de la feromona como un COP.

MODELO

Un modelo $P = (S, \omega, f)$ de un COP consta de:

- Un Espacio de búsqueda S definido sobre un conjunto finito de variables discretas de decisión y un conjunto finito de variables discretas de decisión y un conjunto ω de restricciones entre las variables.
- Una función objetivo $f: S \rightarrow \mathbb{R}^+$ a ser minimizada

El espacio de búsqueda se define de la siguiente manera:

- Variables $X_i, i = 1, \dots, n$ con valores $v_i^j \epsilon\, D_i = \{v_i^1, \dots, v_i^{|D|}\}$
- Instalación de variables: asignar un valor v_i^j a una variable X_i denotado como $x_i \leftarrow v_i^j$
- Una solución $s^* \in S$ se llama optimo global si y solo si $F(s^*) \leq f(s) \forall s \in S$
- $S^* \subseteq S$ es el conjunto de todas las soluciones optimas globales

- Resolver un COP es encontrar al menos una $s^* \in S$

El modelo de un COP se usa para derivar el modelo de feromona de los ACO

- Un componente de solución c_{ij} es la instancia de una variable de decisión $X_i = v_i^j$
- El conjunto de todos los posibles componentes de solución se denota con C.
- A cada c_{ij} se asocia un parámetro de rastro de feromona T_{ij}
- El conjunto de todos los parámetros de rastro de feromona se denota con T
- El valor de un parámetro de rastro de feromona es τ_{ij}

En un ACO se utiliza una representación de conocimiento basada en grafos para encontrar la solución al problema de optimización combinatoria. Para esto se hace un recorrido al grafo de construcción $G_c = (V, E)$. G_c es un grafo totalmente conectado y el conjunto de componentes C se asocian ya sea a los vértices o a los arcos de G_c. En este modelo la solución se construye de la siguiente manera:

- Las hormigas se mueven de un vértice a otro a través de los arcos del grafo de construcción para construir una solución de forma incremental.
- Las hormigas depositan una cantidad de feromona en los componentes
- La cantidad de feromona depositada $\delta\tau$ depende de la calidad de la solución encontrada
- Las siguientes hormigas utilizan la información de la feromona depositada como una guía en la búsqueda de soluciones dentro del espacio de búsqueda

En el cuadro 4 se muestra el algoritmo de la Meta-Heurística ACO

Cuadro 4. Optimización por colonia de hormigas

```
PARA cada arco(i,j)
        Inicializar feromona
FIN PARA
Ubicar las hormigas en nodos aleatorios
Obtener una solución inicial  Smin con costo Cmin

PARA t=1 HASTA tmax HACER
        PARA k= HASTA m HACER
                PARA j=1 HASTA n HACER
                        Aplicar las reglas de construcción/modificación
                        En función de la feromona y de una medida de distancia entre
arcos
                FIN PARA
```

```
            Calcular costo Ck de la solución Sk
            SI Ck < Cini
                  Cmin=Ck y Smin= Sk
            FIN SI
      FIN PARA
      PARA cada arco(i,j)
            Actualizar los rastros de feromona de acuerdo a la regla de
actualización
      FIN PARA
FIN PARA
```

Fuente: Realizado por el autor

2.4.3 Ejemplo de ACO: Problema del Agente Viajero

2.4.3.1 Presentación del Problema La primera versión de ACO se utilizo para resolver el agente viajero.

El grafo de construcción se crea asociando una ciudad a cada vértice y el paso de una ciudad a otra corresponde a los componentes de la solución. El movimiento de la ciudad i a la ciudad j es el componente de la solución C_{ij}. El peso de los arcos indica la distancia entre ciudades y el nivel de feromona se asocia a los arcos.

2.4.3.2 Construcción de la solución

- Cada hormiga inicia desde una ubicación aleatoria o vértice del grafo
- En cada paso de la construcción, la hormiga se mueve a través de los arcos del grafo
- Cada hormiga mantiene memoria de la ruta que ha seguido y al construir su solución no regresa a un vértice ya visitado
- LA hormiga termina de construir su solución cuando ya visito todos los vértices del grafo
- En cada paso de construcción la hormiga elige el arco a seguir (de los disponibles) de manera probabilística dependiendo de la implementación
- Cuando todas las hormigas terminaron su recorrido se actualiza la feromona de los arcos de acuerdo a alguna de las reglas vistas previamente

Al construir soluciones, una hormiga k decide irse de la ciudad i a la ciudad j con la siguiente probabilidad:

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [n_{ij}]^{\beta}}{\sum l \in N_i^k [\tau_{ij}(t)]^{\alpha} \cdot [n_{ij}]^{\beta}} \text{ si } j \in N_i^k \qquad [2.13]$$

Donde $n_{ij} = {}^{1}/_{d_{ij}}$ es información disponible, α y β son parámetros que se tienen que definir, y K_i^k es la vecindad factible de la hormiga k
(el conjunto de ciudades que k no ha visitado)

Si $\alpha = 0$ se visita la ciudad mas cercana. Si $\beta = 0$ se basa solo en las trazas de feromona y tiende a converger rápidamente a un punto de no mejora subóptimo.

Cuando todas las hormigas completan un circuito, se actualizan las feromonas, Primero se reducen todos los caminos por un factor constante (evaporación) y después cada hormiga deposita la siguiente cantidad de feromona en los nodos de su circuito:

$$\forall(i,j)\tau_{ij}(t+1) = (1-\rho)\cdot\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \qquad [2.14]$$

Donde $0 \leq \rho \leq 1$ es la razón de evaporación de feromona y m es el numero de hormigas.

$\Delta\tau_{ij}^{k}(t)$ es la cantidad que deposita cada hormiga en cada nodo, definida por:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} {}^{1}/_{L^k(t)}, & Si\ la\ liga\ (i,j) es\ usada\ por\ la\ hormiga\ k \\ 0, & De\ otra\ forma \end{cases}$$

Donde $L^k(t)$ es la longitud del circuito de la hormiga k.

2.5. LABORATORIOS VIRTUALES

2.5.1. Definición Un laboratorio virtual es un entorno distribuido de herramientas de simulación y animación, cuyo propósito es realizar la simulación interactiva de un modelo matemático. Los laboratorios virtuales proporcionan un método flexible y amigable para definir los experimentos que se llevan a cabo sobre el modelo siendo por ello herramientas útiles para la enseñanza. Típicamente, la definición de un laboratorio virtual incluye las dos partes siguientes: el modelo y la vista. La vista es la interfaz entre el usuario y el modelo, su objetivo es proporcionar una representación visual del comportamiento dinámico del modelo y facilitar las acciones interactivas del usuario sobre aquel. Las propiedades gráficas de los

elementos de la vista se enlazan a las variables del modelo, produciendo un flujo bidireccional de información entre la vista y el modelo. [16]

2.5.2 Laboratorios virtuales en la enseñanza Actualmente, la aplicación del concepto nuevas tecnologías en la enseñanza al ámbito de la realización de prácticas ha dado lugar a la aparición de diferentes modalidades de entornos de experimentación. Desde el punto de vista del estudiante/usuario, los criterios que permiten establecer una clasificación muy clara de estos nuevos entornos son dos: la forma de acceder a los recursos sobre los que se experimenta y la naturaleza del sistema sobre el que se opera. Atendiendo al primer criterio, se puede discernir entre acceso remoto a través de una red y acceso local, es decir, que no implica la utilización de una conexión a Internet para poder operar con los componentes. En lo referente a la naturaleza del recurso, hay que distinguir entre recurrir a modelos simulados o trabajar con plantas reales. De la combinación de estos dos criterios se obtienen cuatro clases de entornos muy diferentes (Ver Tabla 1), pero que abarcan todas las formas de experimentación posibles [17]:

Tabla 1. Taxonomía de los laboratorios

		Tipo de Recurso	
		Real	**Simulado**
Acceso	**Local**	Laboratorio Tradicional	Laboratorio Virtual Mono Usuario
	Remoto	Laboratorio Remoto	Laboratorio Virtual Multi-Usuario

Fuente: Laboratorio virtual [en línea]. Madrid: Departamento de informática y automática UNED

Acceso local-recurso real. Representa el laboratorio de prácticas tal y como lo conocemos, en el que el alumno se sitúa frente a un ordenador conectado a un sistema real para proceder a la realización de la práctica correspondiente.

Acceso local-recurso simulado. Todo el entorno de trabajo es software y la interfaz de experimentación opera sobre un sistema simulado, virtual e inexistente físicamente que reside en el mismo ordenador que la interfaz.

[16] MARTIN, Carla, DORMIDO, Sebastián, URQUIA, Alfonso. Modelado orientado a objetos de laboratorios virtuales con aplicación a la enseñanza del control de procesos químicos.
[17] Laboratorio virtual [en línea]. Madrid: Departamento de informática y automática

Acceso remoto-recurso real. Constituye el acceso al equipamiento de un laboratorio real a través de una red. El usuario opera y controla de forma remota sistemas reales mediante una interfaz de experimentación que se ejecuta en un ordenador conectado a una red (Internet). Este enfoque es lo que se denomina telelaboratorio, laboratorio remoto o teleoperación a través del web.

Acceso remoto-recurso simulado. Esta forma de experimentación es similar a la anterior en cuanto al acceso pero el sistema real se sustituye por un modelo, por lo que el estudiante trabaja con su interfaz de experimentación sobre un sistema virtual accesible a través de Internet. Presenta como diferencia que pueden trabajar múltiples usuarios simultáneamente sobre el mismo sistema virtual ya que al estar simulado se puede instanciar para atender a todo aquel que lo solicite. Estamos pues ante la figura del laboratorio virtual multiusuario o simulación basada en el web.

2.5.3 Criterios de clasificación Pese a que la definición de laboratorio virtual puede parecer bastante intuitiva, es necesario profundizar en ella realizando una clasificación de las diferentes formas de diseñar un laboratorio virtual atendiendo a cuatro criterios [18]:

• La ubicación del motor matemático de cálculo.

• La naturaleza del núcleo de simulación.

• Las capacidades de diseño que tiene disponibles el usuario.

• El grado de interactividad con la simulación.

Acorde con el primer criterio, la ubicación del motor de cálculo, se puede distinguir entre local o remota. Dentro del ámbito en que nos movemos, las simulaciones locales se caracterizan porque el motor de cálculo se encuentra en el computador en el que está trabajando el usuario, de forma que la interfaz gráfica y el núcleo numérico forman un todo que convive en el mismo entorno, es decir, dentro del navegador. Por el contrario, en las simulaciones remotas el núcleo numérico se ejecuta en un ordenador remoto, la interfaz gráfica y la simulación se ejecutan en ordenadores diferentes.

El segundo criterio de clasificación es la naturaleza del núcleo de simulación con independencia de que su ubicación sea local o remota. Este criterio considera si la simulación en sí misma ha sido construida por medio de una herramienta específica orientada al modelado y la simulación (Matlab, Simulink, ACSL,

[18] SÁNCHEZ, José, MORILLA, Fernando, DORMIDO, Sebastián. Laboratorios virtuales y remotos para la práctica a distancia de la automática

Dymola, Ecosim, etc.), o se ha recurrido a lenguajes de alto nivel de propósito general (C, C++, Fortran, Java) mediante el empleo de librerías específicas orientadas a la simulación.

Las capacidades de diseño consideran que el cliente pueda cambiar no sólo los parámetros numéricos del modelo a simular, sino que tenga la posibilidad de modificar su arquitectura. De esta forma, el cliente no se limita únicamente a la introducción de parámetros para configurar el comportamiento del modelo sino que toma parte activa en la construcción del propio modelo.

La cuarta y última característica es el grado de interactividad con la simulación. Podemos distinguir dos casos: pseudo-batch y on-line. En la simulación pseudo-batch no hay inmediatez desde que se inicia el proceso de simulación hasta que se obtiene la respuesta en forma de datos numéricos o gráficos. La simulación on-line representa el polo opuesto pero, a su vez, es el más sugerente. En este caso, el proceso de simulación avanza de forma continua y dinámica, obteniendo el usuario en cada periodo de muestreo de tiempo simulado los resultados bajo la forma de un flujo continuo de valores numéricos o de gráficos evolucionando de forma sostenida. Otra gran diferencia con respecto al supuesto previo es que, según se modifica un parámetro en la interfaz, la respuesta del sistema es inmediata. Dadas las características de este tipo de simulaciones, la herramienta por excelencia es el lenguaje Java.

Resumiendo, con independencia de las posibilidades de diseño y del grado de interactividad, los laboratorios virtuales más frecuentes son tres:

• La interfaz gráfica y el motor de simulación constituyen una aplicación monolítica que se ejecuta dentro del navegador WWW y reside en el ordenador del cliente (Figura 7.a).

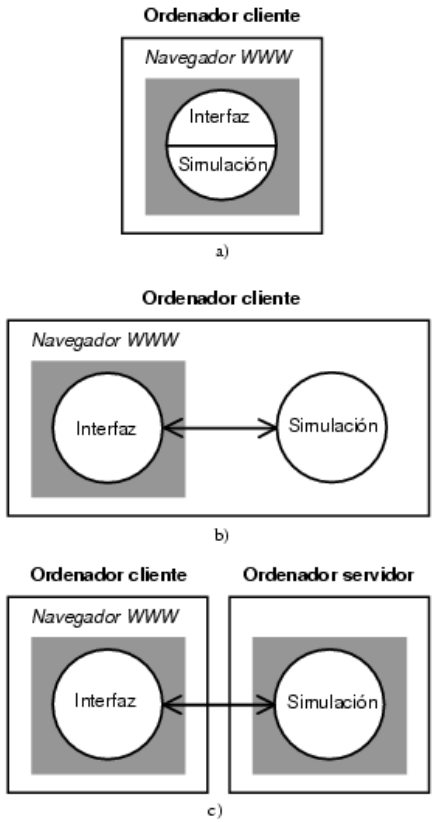
• La interfaz gráfica y el motor son aplicaciones independientes que residen ambas en el mismo computador, es decir, en el del cliente (Figura 7.b). La interfaz se localiza dentro del navegador WWW, mientras que el motor habitualmente es un entorno de simulación del tipo Matlab.

• La interfaz gráfica y el motor son aplicaciones independientes y están físicamente separadas (Figura 7.c). Como en el caso previo, la interfaz se localiza en el navegador WWW del cliente pero la simulación reside en el servidor remoto.

2.5.4 Ventajas Los laboratorios virtuales ofrecen un gran número de ventajas, debido a que se puede simular cualquier cosa, desde un laboratorio de química y física, hasta robótica remota o robótica simulada. Las ventajas son las siguientes

- Practicas remotas, en las cuales los estudiantes pueden tomar parte sin tener que estar presentes en el laboratorio.
- Aprender en un ambiente libre y flexible en contraste a unas clases fijas y regulares que están cronometradas.
- Un fácil proceso de autoevaluación, con su propia corrección automática en línea y muestra de resultados inmediata.
- Objetos de gran costo pueden ser simulados, con la mayor realidad posible para evitar el uso del mismo y por ende su deterioro de alguna forma.

Figura 7. Configuraciones más habituales de simulaciones basadas en Web (SBW)



Fuente: SÁNCHEZ, José, MORILLA, Fernando, DORMIDO, Sebastián. Laboratorios virtuales y remotos para la práctica a distancia de la automática

2.5.5 Requisitos de un buen laboratorio Virtual Los requisitos que debe cumplir un buen laboratorio virtual según Kappelman son:

1. Ser auto contenido.
2. Ser interactivo.

3. Combinar imágenes bidimensionales y tridimensionales.
4. Tener animación tridimensional, video y sonido.
5. Incluir ejercicios (cuya calificación puede ser enviada automáticamente al docente).
6. Instalación Automática.
7. Que la navegación no sea necesariamente lineal.
8. Posibilidad de guardar notas sin necesidad de procesador de textos externo.
9. Contar con un buscador.

2.5.6 Clasificación de la complejidad La complejidad de los laboratorios virtuales es variable y puedes ser clasificada de la siguiente manera:

- *Laboratorio de pruebas:* Estos son laboratorios para prácticas en los cuales un sistema físico no es requerido. Algunos ejemplos son los laboratorios de física, matemática y química que solo requieren algunas herramientas de calculo.
- *Laboratorio de simulación:* Una simulación de un ambiente real puede ser hecho a través de realidad virtual en un laboratorio virtual. Para lograr un sistema mas realista, la realidad virtual es definida como una interfase humano-computador de alto nivel a través de la cual el usuario es capas de interactuar con el ambiente simulado en tiempo real y a través de múltiples canales sensoriales. Una desventaja es que los estudiantes no obtienen un dato real, solo los datos simulados. Algunos de los ejemplos de estos laboratorios son Simulink, PSpice, Labview.
- *Laboratorios de ejecución remota y simulación:* En adición a la simulación de realidad virtual, el equipo real es remotamente controlado a través de una sesión de Internet previa pero no en tiempo real. Las mejores características de esta opción son: Gran motivación del estudiante, ya que es capaz de comparar el resultado real con el previamente simulado y la opción de ver resultados en equipos caros sin la necesidad de tocar el sistema real.
- *Laboratorios de tele ejecución en tiempo real:* Este es el laboratorio virtual mas realista de todos, permite la tele-operación de los equipos en tiempo real. El mayor inconveniente es que necesita una retroalimentación usando tiempo real, que puede estar limitado por el ancho de banda del medio que comunica al estudiante con el laboratorio real. Sin embargo, el sistema introduce la opción de mejorar una retroalimentación de fuerza, de esfuerzo y una retroalimentación visual.

3.0. DESARROLLO DEL LABORATORIO VIRTUAL DE INTELIGENCIA DE ENJAMBRES

3.1 PLANTEAMIENTO DEL PROBLEMA

Actualmente el departamento de automática de la Universidad Autónoma de Occidente no cuenta con un Laboratorio Virtual de Inteligencia de Enjambres que permita la práctica y el aprendizaje de los conceptos de optimización por enjambre de partícula y enjambre de bacterias.

Debido a que existen pocas herramientas para la simulación de la inteligencia de enjambres, y estas herramientas existentes son enfocadas a una aplicación específica. Una herramienta donde se encuentran aplicaciones de la inteligencia de enjambres es MATLAB-SIMULINK sin embargo, se tiene el inconveniente de ser una herramienta comercial de alto costo generalmente no asequible para personas que estén por fuera del ámbito académico. Se ha decidido diseñar e implementar dos módulos de Inteligencia de Enjambres como lo son Optimización por Enjambres de Partículas y Optimización por Enjambre de Bacterias con el fin de ayudar a la docencia para la facilidad de generar nuevos cursos virtuales.

3.1.2 Metodología Desarrollada Antes de diseñar y por consiguiente desarrollar un laboratorio virtual, es indispensable llevar a cabo un análisis extensivo del contexto en el cual vamos a implementar el sistema, atendiendo a las diferentes necesidades educativas, funcionales y de información de los usuarios.

3.2 DESARROLLO CONCEPTUAL

Para el desarrollo del Laboratorio Virtual de Inteligencia de Enjambres se prevé que las aplicaciones otorguen al usuario una amplia visión, aplicando en el todos sus conocimientos acerca de la Inteligencia de Enjambres.

3.2.1 Identificación de necesidades

3.2.1.1 Planteamiento del cliente

- Acceso al laboratorio virtual desde cualquier lugar de la universidad a través de Internet.
- Poseer la teoría necesaria en la página WEB

- Implementación Demos de PSO y BSFO
- Implementación de algunas aplicaciones para PSO y BSFO
- Estabilidad en el programa
- La interfaz grafica sea muy fácil de manejar y entender
- El laboratorio genere nuevos conocimientos

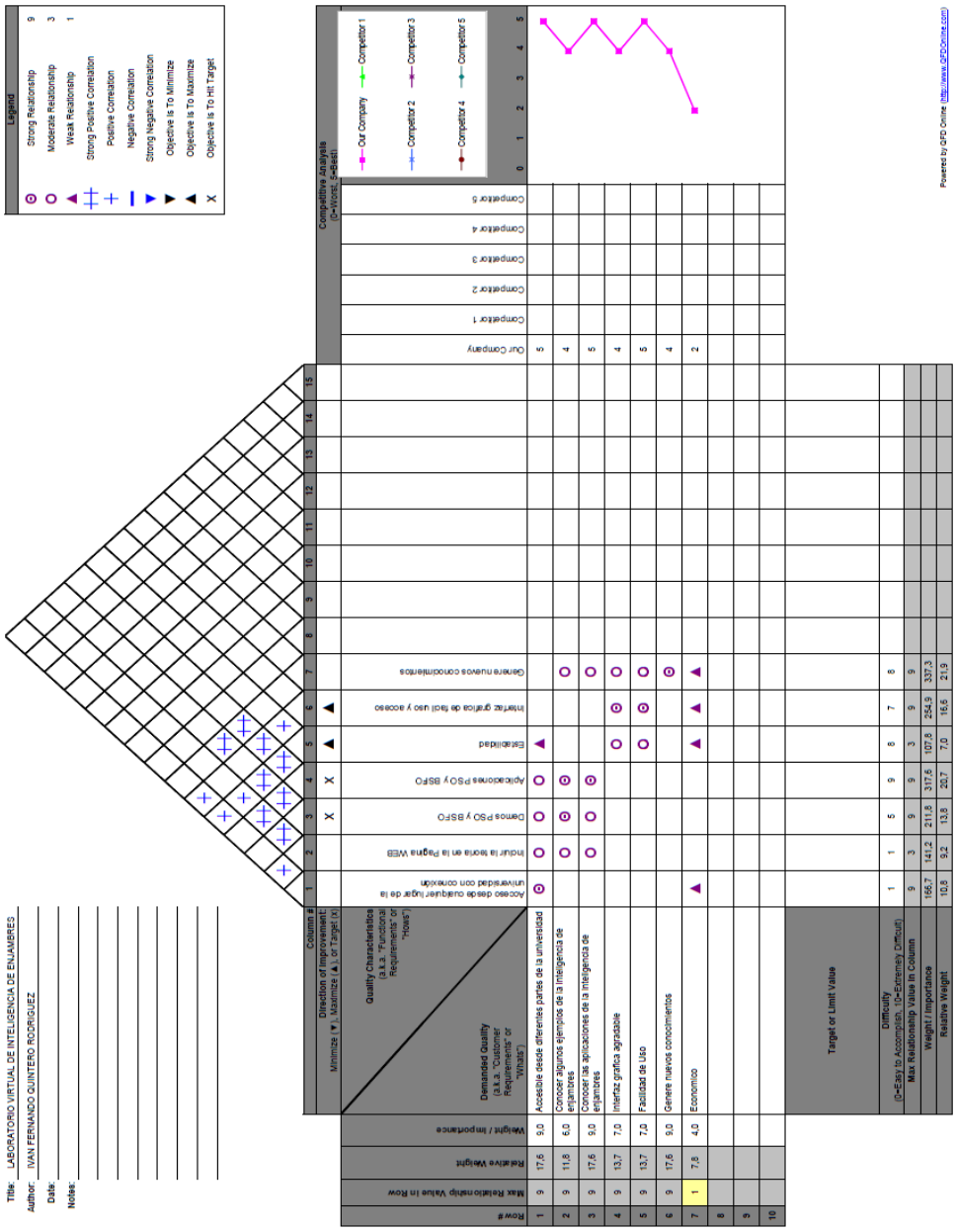
3.2.1.2 Necesidades del cliente

- Accesible desde diferentes partes de la universidad
- Conocer algunos ejemplos de la inteligencia de enjambres
- Conocer las aplicaciones de la inteligencia de enjambres
- Interfaz grafica agradable
- Facilidad de uso
- Genere nuevos conocimientos
- Económico

3.2.2 Especificaciones preliminares

3.2.2.1 Quality Function Deployment (QFD)

Figura 8. Quality Function Deployment (QFD)

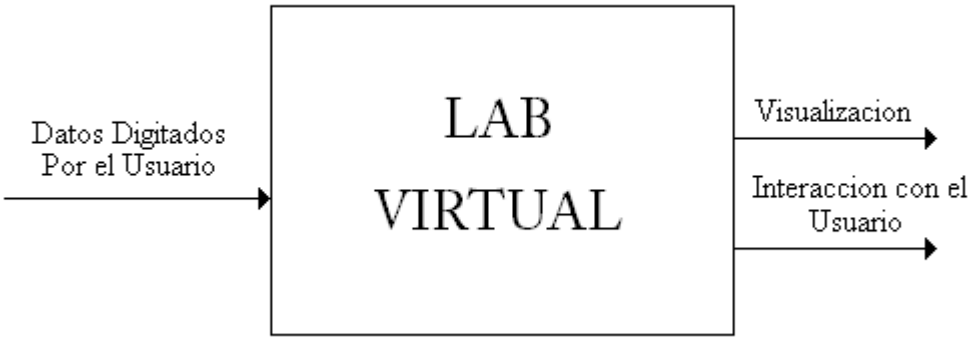


Fuente: Realizada por el autor.

3.2.3 Generación de conceptos

3.2.3.1 Descomposición funcional Antes que nada, es necesario definir claramente el problema a solucionar, para ello usaremos una caja negra para simplificar el problema, la cual operara por los datos digitados por el usuario.

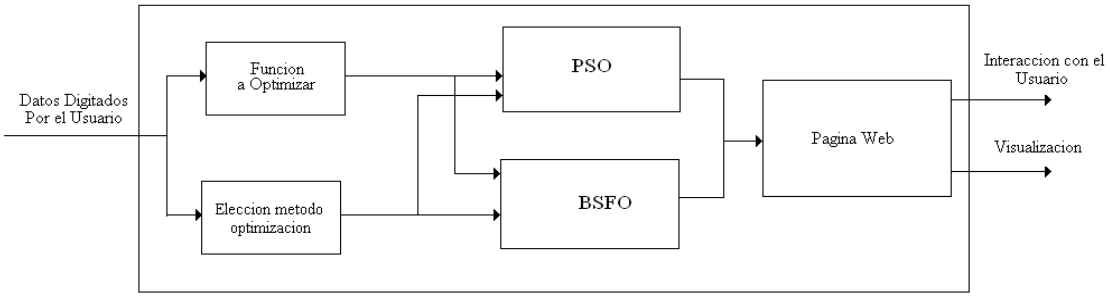
Figura 9. Caja Negra



Fuente: Realizada por el autor.

Por medio de la caja negra se ha definido y clarificado e problema, a continuación como se divide la caja negra en varias subfunciones para la generación de una solución optima al problema planteado, subfunciones las cuales describan específicamente que debe hacer cada elemento inmerso en el producto para implementar la función principal.

Figura 10. Descomposición funcional



Fuente: Realizada por el autor

3.2.3.2 Exploración sistematizada

Selección Aplicación cliente. La aplicación Cliente fue desarrollada mediante **Easy Java Simulations (EJS),** el cual es un software de distribución gratuita (Open source) y que a su vez facilita la creación de simulaciones interactivas en lenguaje JAVA definidas en función del paradigma modelo - control – vista, los cuales se actualizan continuamente en función de la dinámica del sistema.

La elección de Java como lenguaje de desarrollo se justifica por su amplia aceptación por parte de la comunidad internacional de Internet y el hecho de que está soportada en diferentes plataformas de software, esto significa que las simulaciones creadas usando EJS pueden ser utilizadas como programas independientes bajo diferentes sistemas operativos o se pueden distribuir de manera inmediata vía Internet y ejecutadas como applets dentro de páginas HTML utilizando algunos de los navegadores de Web más populares [19].

Selección Visualización. Para visualizar la aplicación cliente desarrollada en EJS se desarrollo una **Página Web** en la cual se tendrá acceso a los applets, así como algunos conceptos teóricos, bibliografía y un conjunto de instrucciones que indiquen al usuario (estudiante) la forma correcta de como manipular la aplicación sin inconvenientes.

3.3 DISEÑO A NIVEL DEL SISTEMA

Tomando en cuenta la identificación de necesidades se procede a diseñar el ambiente virtual. Teniendo en cuenta los requerimientos que han sido extraídos del análisis inicial, se procede a llevar a cabo el diseño del ambiente virtual de aprendizaje. Dividiéndolo en tres niveles diferentes: Diseño educativo, Diseño de la presentación y diseño de la interfaz.

3.3.1. Diseño Educativo El laboratorio virtual tiene como principio generar nuevos conocimiento a los usuarios potenciales (estudiantes).

Para el diseño Educativo de nuestro Laboratorio Virtual consideramos los siguientes principios:

- Interactividad persona-computador, asegurando que los medios interactivos estén directamente relacionados con las dimensiones pedagógicas señaladas.

[19] SÁNCHEZ, José, MORILLA, Fernando, DORMIDO, Sebastián. Laboratorios virtuales y remotos para la práctica a distancia de la automática

- Aprendizaje centrado en procesos más que en contenidos, localizando la actividad del usuario más en los procesos para generar y utilizar información, que en el cuerpo de la información misma.

- Globalización, considerando que el acceso al laboratorio virtual puede realizarse desde cualquier parte del mundo, y que está disponible a quien decida buscarla por la red, habilitando al usuario para que los contenidos educativos puedan identificarse; transferirse localmente, valorarse y aprovecharse según sus necesidades.

3.3.1.1. Qué aprenderá el usuario En general, con el uso del Laboratorio Virtual se espera que el usuario aprenda los conceptos básicos de la Inteligencia de Enjambres.

Para cumplir con esta premisa, el laboratorio virtual debe presentar a los ejercicios de Inteligencia de enjambres en términos de un conjunto de etapas seguidas desde el punto de vista teórico y práctico.

3.3.1.2. Cómo Aprenderá el Usuario El usuario aprenderá según las dimensiones pedagógicas establecidas para orientar el diseño Educativo. Básicamente, la esencia de la experiencia de aprendizaje, se centra en los elementos interactivos de la interfaz, en particular en las simulaciones, pues es a través de ellas que el usuario recibe la retroalimentación de los efectos de sus acciones sobre el comportamiento del sistema en estudio.

3.3.2. Diseño de la Presentación En nuestro caso para el diseño de la presentación, basta abordar aquellos aspectos para construir un sistema que facilite al máximo los procesos de comprensión y asimilación de la información presentada a los usuarios en el contexto de las simulaciones.

El primer paso consiste en darle una estructura general a los contenidos que maneja el Laboratorio Virtual.

Claramente, la elaboración del diseño de presentación del Laboratorio Virtual requiere de varios tipos de pantalla, que podemos clasificar según la funcionalidad del contenido que albergan.

- **Pantallas Informativas**: Están formadas por aquellos contenidos cuya función es informar al usuario sobre el Laboratorio Virtual y sus recursos locales. En esta categoría se incluyen también a las ventanas emergentes cuyo propósito es aclarar conceptos específicos o proporcionar mensajes de ayuda.

- **Pantallas expositivas:** Consta de todos los contenidos que exponen la base teórica de la inteligencia de enjambres.
- **Pantallas de experimentación:** Son las que contiene al entorno de simulación propiamente dicho.

Las pantallas informativas y expositivas están formadas por elementos de texto e imágenes en formatos ligeros, mientras que las pantallas de experimentación, además de texto e imágenes, incluyen a los componentes Web donde se ejecuta la interfaz de las simulaciones y se despliegan los resultados.

En general, para todas las pantallas la apariencia (fuentes, colores, fondos, enlaces, etc.) está definida favoreciendo una presentación homogénea y más estética al usuario. El conjunto de todos estos aspectos determina la usabilidad del sistema.

3.3.3. Diseño de la Interfaz La interfaz grafica de usuario (GUI) del Laboratorio Virtual debe disponer de las características necesarias para su integración con las tareas que apoya. Tales características incluyen aspectos como:

- **Facilidad de uso:** La tecnología utilizada para implementar el Laboratorio Virtual debe aportar aplicaciones fáciles de utilizar, mantener o modificar. La interfaz con el usuario está basada en páginas Web cuya utilización está ampliamente difundida y aceptada. Por tal razón el diseño, mantenimiento y modificación del Laboratorio Virtual no necesita de un alto grado de especialización.

- **Sistema abierto:** Se refiere a un sistema cuyo funcionamiento sea independientes de la plataforma. El entorno de simulación Web no debería depender de los sistemas operativos con los que se trabaje. Cada nodo del sistema (cliente y servidor) deberá contar con conexión a Internet y cada sistema operativo deberá incluir la pila de protocolos de comunicaciones necesaria.
- **Acceso transparente:** Garantiza el acceso a la información y los servicios desde cualquier computador conectado a la red.

- **Estándar libre:** El diseño y desarrollo del Laboratorio Virtual debe estar preferiblemente basados en estándares y tecnologías ampliamente extendidas y de acceso libre.

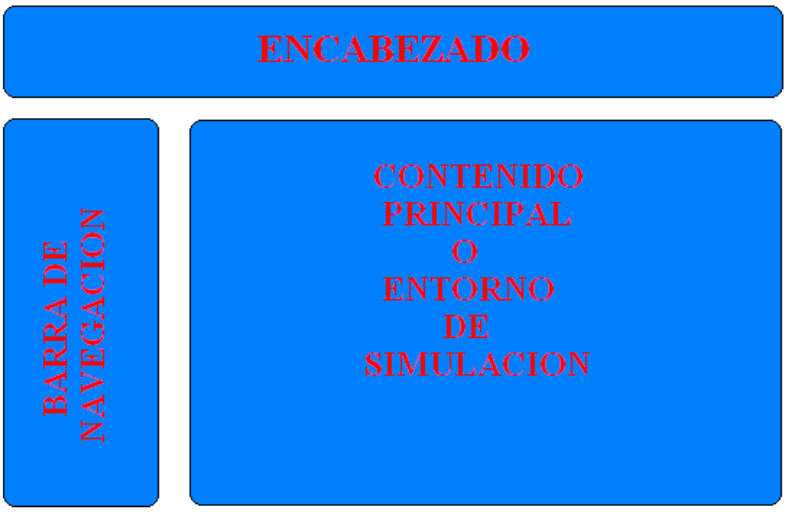
Las paginas en HTML puede crearse con cualquier editor de texto disponible y los componentes dinámicos pueden implementarse en distintos lenguajes de programación establecidos (C, Java, etc.). De esta forma también aseguramos un costo económico bajo.

En base a esto, y a partir de la información obtenida en las fases de diseño educativo y comunicacional, podemos establecer en esta etapa un diseño más refinado del rostro del Laboratorio Virtual.

Cuando hablamos de la GUI del Laboratorio Virtual, nos estamos refiriendo en realidad a dos tipos de GUI: la de las páginas Web que conforman el sitio Web donde colocamos los diferentes contenidos del Laboratorio Virtual SI, y la GUI del entorno de simulación.

La figura 11 muestra un bosquejo de la forma en la cual se pueden organizar los elementos GUI dentro de las páginas Web, siendo esta referencia para organizar los elementos de la pantalla informativa, formativa y la pantalla de experimentación.

Figura 11. Bosquejo para la Página Web



Fuente: Realizada por el autor

Para las distintas aéreas mostradas en los bosquejos, definimos:

- **Encabezado:** Es el área destinada para desplegar el titulo de la página. Contiene elementos de texto y la imagen del logotipo de la sección del sitio Web que alberga el laboratorio Virtual.

- **Barra de Navegación:** Como su nombre lo indica, es aquí donde se coloca el control principal de navegación definido. Esta barra está implementada con elementos de imagen dinámicos.

- **Contenido Principal (General):** Es el área donde colocamos los elementos de texto e imágenes correspondientes a los contenidos informativos y formativos.

- **Contenido Principal (Entorno de Simulación):** Está área aparece exclusivamente en las pantallas de experimentación, y está destinada para colocar el entorno de simulación compuesto por un componente Web interactivo con una GUI propia.

3.3.3.1. La GUI en Entorno de Simulación Con el fin de reforzar el dinamismo y los aspectos cualitativos de la GUI del Laboratorio Virtual, el entorno de simulación debe estar dotado de elementos interactivos, y debe favorece la visualización dinámica y la animación de los elementos (cuando el sistema lo permite).

La GUI del entorno de simulación consta de los siguientes elementos: el diagrama del proceso, los paneles de control y los registros gráficos.

A continuación se describen cada uno de ellos.

Diagrama de Proceso: está compuesto por un esquema grafico del proceso (algunos de los objetos presentes pueden tener animación) con visualización alfanumérica de las señales y unidades más importantes.

Paneles de Control: tienen tres tipos de elementos (botones, deslizaderas y campos).

Panel de Simulación: consiste en botones que permiten inicializar, arrancar, parar y reiniciar la simulación del proceso. Incluye también las cajas de chequeo que activan el despliegue del registro grafico y otros elementos opcionales. Se localiza en la parte superior de la GUI.

Panel de Variables Interactivas: consiste en deslizaderos y/o campos alfanuméricos para modificar el valor de los parámetros del modelo.

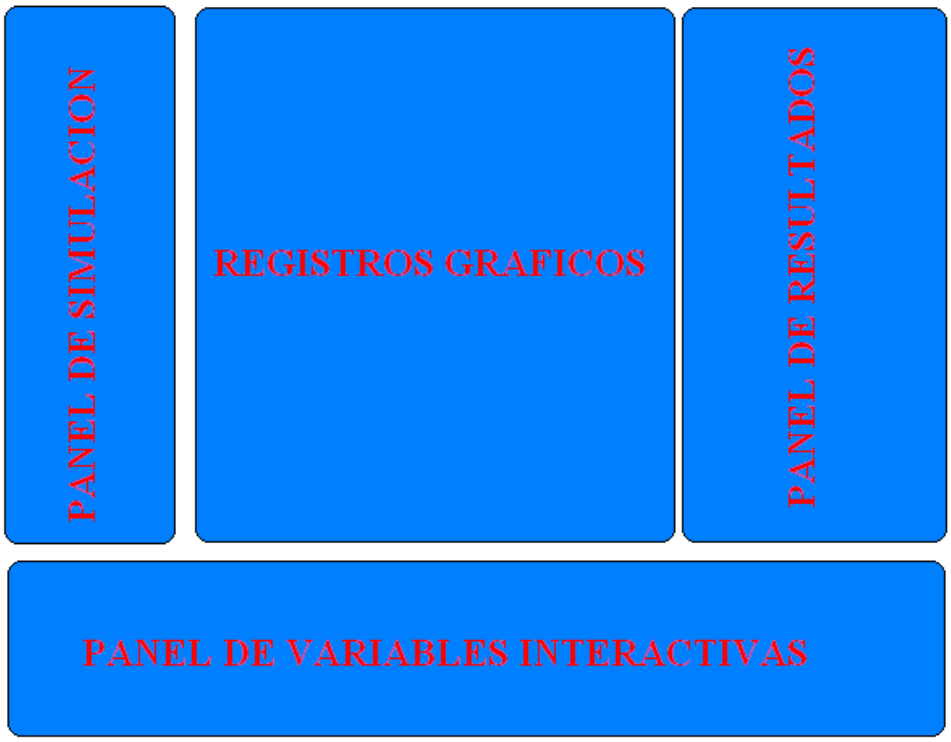
Panel de Resultados: este panel incluye el despliegue de los resultados numéricos y textuales de la simulación, es decir muestra información de salida.

Registros gráficos: proporcionan la visualización gráfica de las variables principales del sistema. Al mismo tiempo cuando la simulación progresa, estos registros reflejan de forma dinámica y continua cualquier cambio en las variables que toman parte del proceso. Los cambios en las señales del sistema se pueden deber a las acciones del usuario (movimiento de una deslizadera en el diagrama del proceso). Este panel puede o no ser visible, de acuerdo con la preferencia del

usuario, es por ello que no forman parte de la ventana que contiene a los paneles principales

El bosquejo de la GUI del entorno de simulación con la ubicación de todos sus elementos se muestra en la figuras 12.

Figura 12. Bosquejo de la GUI del Entorno de Simulación



Fuente: Realizada por el autor

Es importante tener en cuenta, que los entornos de simulación deben estar definido en el contexto de una pantalla de experimentación, en donde la simulación propiamente dicha, está antecedida por una descripción de los objetivos, conocimientos previos requeridos e información sobre el funcionamiento de la simulación.

3.4 DESARROLLO DE LA SOLUCIÓN

Conforme al diseño que se ha efectuado y observando los lineamientos planteados en la etapa de análisis, se continúa con la etapa de desarrollo donde se lleva a cabo la elaboración del laboratorio virtual.

Es importante tener en cuenta que el desarrollo de este proyecto va dirigido hacia dos vertientes: El entorno de Simulación (vertiente principal) y el sitio Web donde se incluye el entorno de simulación. Es al conjunto de las dos vertientes lo que llamamos Laboratorio Virtual SI.

Para la construcción del sitio Web, utilizamos tecnologías Web básicas, como lo son:

HTML, es un bloque básico para construir páginas Web, para componentes expositivos, además HTML es un elemento indispensable para la creación de pagina web.

Flash, es un programa de animación vectorial, que incorporan animaciones, sonido y efectos realmente interesantes. En la páginas web

JavaScript, es un lenguaje pequeño y ligero, diseñado para una fácil incrustación en otros productos y aplicaciones, tales como los navegadores Web.

Para poder darle al Laboratorio virtual la dimensión pedagógica necesaria, es importante el diseño y construcción del sitio Web; el núcleo de éste desarrollo es el entorno de simulación, debido a que constituye el elemento de experimentación en la cual se sustenta el Laboratorio Virtual SI.

3.4.1. El Respaldo Matemático de las Simulaciones Existen diferentes alternativas que podrían seleccionarse como motor matemático del entrenamiento, entre las cuales MATLAB® y Java que son los de más amplia utilización para propósitos educativos.

Realizando un análisis sobre la base de MATLAB® y Java, las formas de desarrollar las funcionalidades de MATLAB® para implementar un ambiente de simulación con las capacidades demandadas, es necesario manejar aspectos computacionales complejos. Además, es importante destacar que al utilizar MATLAB® traería consigo limitaciones de acceso, pues el uso del ambiente de simulación estaría limitado solo a personas con autorización a utilizar la licencia.

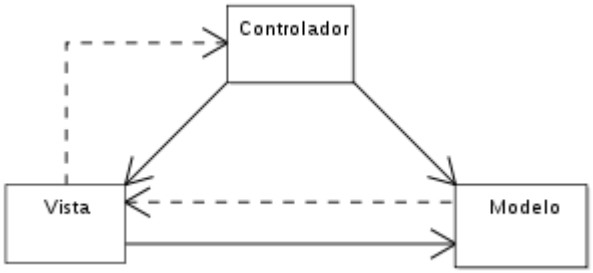
continuando el análisis en el contexto de las herramientas expuestas, observamos que a través del uso de Easy Java Simulations (Ejs), Java se convierte en una de

las alternativas más adecuadas para implementar ambientes de simulación basados en Web para el contexto académico, disminuyendo el nivel de complejidad de las simulaciones a los aspectos de modelado matemático del sistema en estudio, al manejo adecuado de los conceptos de teoría de sistemas inteligentes y al conocimiento básico del lenguaje de programación Java.

3.4.2. Construcción de las Simulaciones en Java con Easy Java Simulations

Una de las mayores ventajas de *Ejs* consiste en que está estructurado para implementar las simulaciones de forma directa en términos del paradigma MVC (Modelo-Vista-Control) como se observa en la figura 13. Para simplificar la construcción de las simulaciones, *Ejs* suprime la parte de control, colocándola repartida, una mitad en la vista y la otra mitad en el modelo. Esto es posible porque *Ejs* define el control en base a las relaciones de los objetos del modelo y las vistas.

Figura 13. Flujo entre las Capas de una Arquitectura MVC



Fuente: Realizado por el autor

En la interfaz de Ejs se distinguen tres partes principales: la introducción, el modelo y la vista.

La introducción es una sección alternativa para la creación de una estructura de Páginas Web para colocar las simulaciones generadas en forma de Applets de Java. Como esta estructura ya se creo, con este propósito solo se utilizará las secciones correspondientes al Modelo y a la Vista.

De esta forma, la construcción del entorno de simulación en Ejs consiste en un desarrollo en dos pasos:

- Definición del modelo de la simulación (Modelo en Ejs).
- Diseño de la vista de la simulación (Vista en Ejs).

El punto más crítico de la programación con Ejs está en establecer las conexiones requeridas entre la Vista y el Control para lograr la visualización correcta de los

estados del sistema que se simula y una interacción apropiada del usuario con la GUI de la simulación.

3.4.3. Construcción del Laboratorio Virtual SI Par finalizar el desarrollo de la solución, basta con construir la pagina Web con los contenidos informativos y formativos del Laboratorio Virtual SI (incluyendo la pagina HTML con los Applets de las simulaciones), siguiendo las pautas trazadas en las fases de análisis y diseño.

Como es un sitio Web bastante sencillo y esta fuera del propósito de este trabajo, no se va a mostrar los detalles de la construcción de la página Web (tomando en cuenta que lo más importante en la relación a este tema ya fue expuesto con suficiente detalle en la fase de diseño).

3.5 ADMINISTRACIÓN

La administración del Laboratorio virtual SI incluye todo lo necesario para asegurar un funcionamiento correcto del sistema con el mínimo de problemas y una satisfacción por parte de los usuarios.

3.5.1 Instalación y Configuración Para introducir al Laboratorio Virtual SI en su etapa funcional, definimos finalmente cuál es el soporte del sistema y la configuración requerida. Refiérase a la tabla 2.

Las Anteriores especificaciones corresponden al servidor que se utilizo para instalar, configurar y probar el entorno de simulación del Laboratorio Virtual SI. Sin embargo el sistema desarrollado presenta características que le permiten ser instalado, configurado y habilitado en un servidor en cualquier plataforma que soporte el entorno de Java 2.

Tabla 2. Especificaciones de Instalación y Configuración del Servidor

Especificaciones	Descripción
Sistema Operativo	Microsoft® Windows XP
Hardware	Procesador AMD Athlon 2.3GHz y Memoria RAM 512 MB
Servidor Web (HTTP)	Apache HTTP Server 2.0.55

Otros	Entorno Java: Java 2 Standard Edition (J2SE) versión 5.0 con JDK 1.5

Fuente: Realizada por el autor

Así mismo, para que el sistema funcione adecuadamente, puede utilizarse cualquier navegador Web que soporte gráficos (Microsoft Internet Explorer, Mozilla Firefox o Netscape Navigator, por ejemplo) y que tenga habilitada la maquina virtual de Java (JVM por las siglas en ingles Java Virtual Machine).

3.5.2. Administración previa al uso del Laboratorio Virtual SI Se sugiere a los docentes de las asignaturas relacionadas con la temática de sistemas inteligentes, si se decide a incorporar este recurso académico como apoyo a la formación práctica dentro de sus asignaturas, implementar un conjunto de condiciones de uso e interacción con los estudiantes y hacer lugar practicas de prueba que faciliten el proceso de adaptación y evidencien el grado de aceptación de los usuarios.

3.6. EVALUACIÓN

Con la evaluación se quiere comprobar si existen errores a nivel de análisis, diseño y desarrollo. Esto se logra básicamente con dos tipos de pruebas:

Prueba de Interfaz y de Funcionamiento: En esta prueba se puede detectar errores de funcionamiento y fallas en el ambiente desarrollado.

Primero que todo se requiere de la evaluación del especialista, quien se encarga de revisar el contenido que se desea transmitir a través del sistema. Luego, se realiza en una situación real de aprendizaje, utilizándolo como apoyo pedagógico en el desarrollo de un curso normal. Con esta prueba además se determina la aceptación del software por parte de los usuarios.

Específicamente, para validar el funcionamiento del entorno de simulación lo más importante es asegurar que los resultados que se están obteniendo ante los valores de entrada introducidos sean correctos.

4.0. PRESENTACION DE LA SOLUCION

El siguiente capítulo describe en detalle el Laboratorio Virtual de Inteligencia de Enjambre o laboratorio Virtual SI; en primera medida se ilustra el prototipo por medio de imágenes correspondientes a la interfaz grafica desarrollada para los módulos genéricos de PSO y BSFO, respectivamente. Finalmente, se presentaran tres módulos “Demos” para cada uno de los algoritmos de optimización.

4.1. PRIMERA ETAPA: MODULOS DEMOS

Esta primera etapa pretende familiarizar al usuario con los conceptos básicos de inteligencia de Enjambres y visualizar algunas aplicaciones de la misma.

Los módulos demos fueron divididos en los dos algoritmos de optimización por enjambres: Los demos PSO y los demos BSFO.

4.1.1. Modulo Demos PSO Siguiendo el esquema general de interfaz gráfica propuesta en la sección **3.3.3.1 Figura 12**, se crea una ventana con los cuatro paneles principales. Al abrir el Modulo Demo de PSO, el usuario se encuentra con una pantalla de inicio, la cual le proporciona la posibilidad de elegir alguno de los dos demos: Demo 2D, Demo 3D o el Demo Control Nivel de un Tanque. En la figura 14 se puede apreciar esta pantalla de inicio.

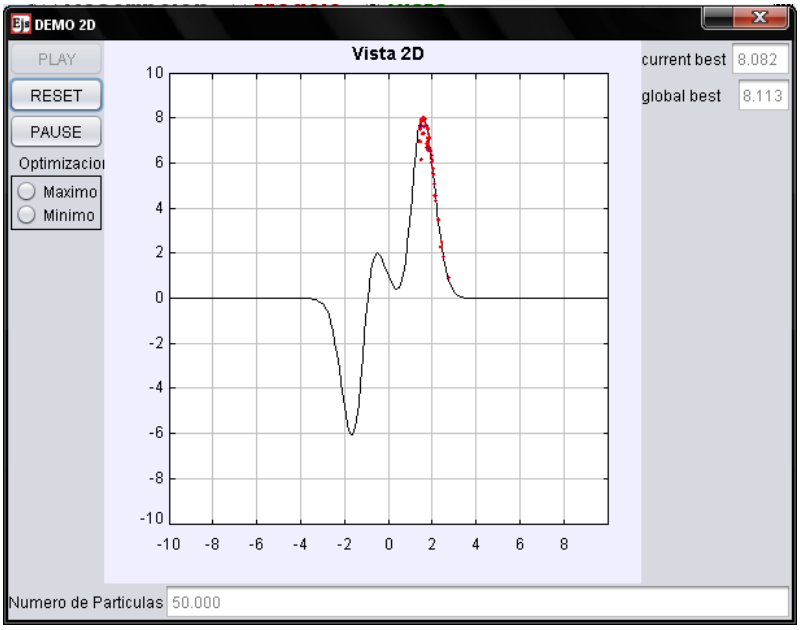
4.1.1.1. Modulo Demo 2D PSO Este modulo muestra una optimización de una función de de una sola variable dependiente. La función a optimizar escogida fue la función “Peaks”, escogida por sus picos positivos y negativos de la misma amplitud, pudiendo así mostrar al usuario una optimización de dicha función en su máximo o mínimo punto. En la figura 15 se puede apreciar el modulo.

El Usuario podrá escoger si la optimización será de un mínimo o será de un máximo, además de la cantidad de partículas que conformaran el enjambre.

Figura 14. Pantalla de inicio modulo Demo PSO

Fuente: Realizada por el autor.

Figura 15. Modulo Demo PSO 2D



Fuente: Realizada por el autor.

Este modulo al igual que los demás consta de una botonera esencial, con los botones Play, Reset y Pause. Además de la posibilidad de ingresarle la Cantidad

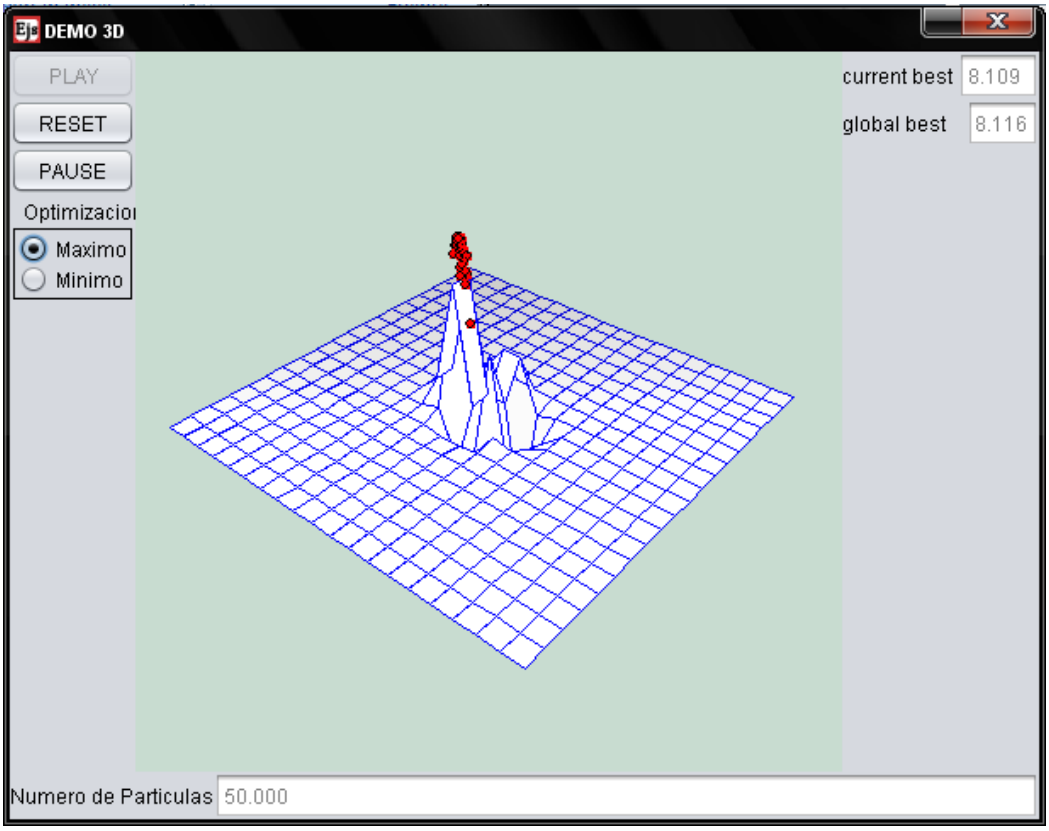
de agentes que actuaran en el algoritmo. También contiene la visualización de datos de la mejor posición encontrada por el Algoritmo.

4.1.1.2. Modulo Demo 3D PSO Este modulo muestra una optimización de una función de dos variables dependientes. En la figura 16 se puede apreciar el modulo.

El Usuario podrá escoger si la optimización será de un mínimo o será de un máximo, además de la cantidad de partículas que conformaran el enjambre.

Este modulo al igual que los demás consta de una botonera esencial, con los botones Play, Reset y Pause. Además de la posibilidad de ingresarle la Cantidad de agentes que actuaran en el algoritmo. También contiene la visualización de datos de la mejor posición encontrada por el Algoritmo.

Figura 16. Modulo Demo PSO 3D

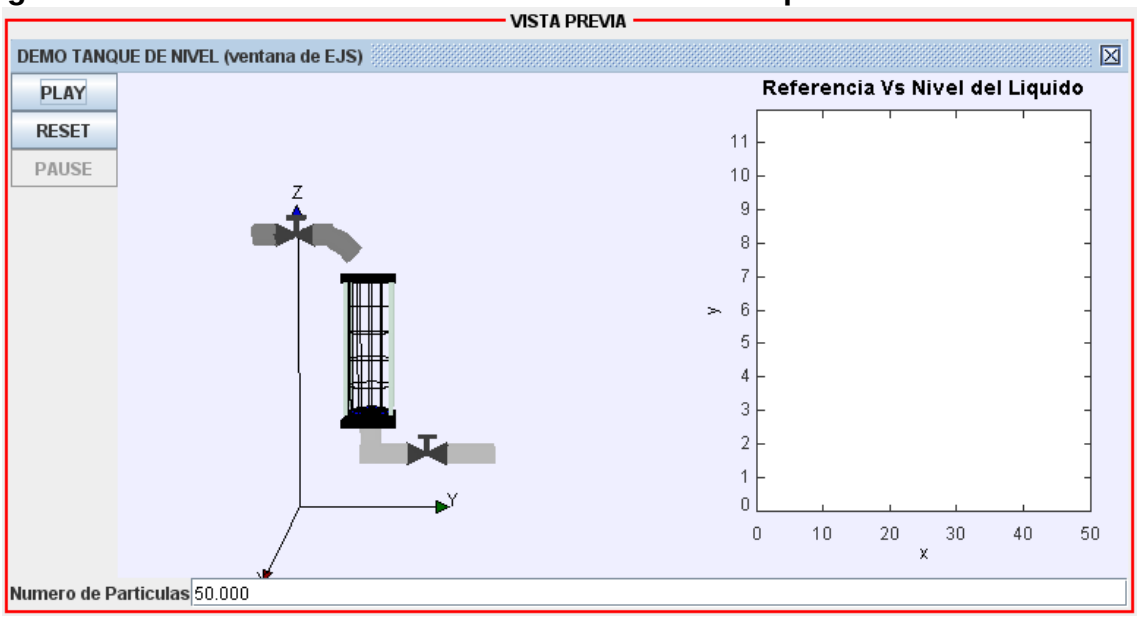


Fuente: Realizada por el autor.

4.1.1.2. Modulo Demo Control de Nivel del Tanque Este modulo muestra el control del nivel de agua de un tanque, por medio del algoritmo PSO. Ver figura 17.

Este demo expone el comportamiento del PSO ante cambios de estimulo de control con una señal de pulsos, simulando la orden de llenado y vaciado del tanque.

Figura 17. Modulo Demo Control del Nivel de un Tanque



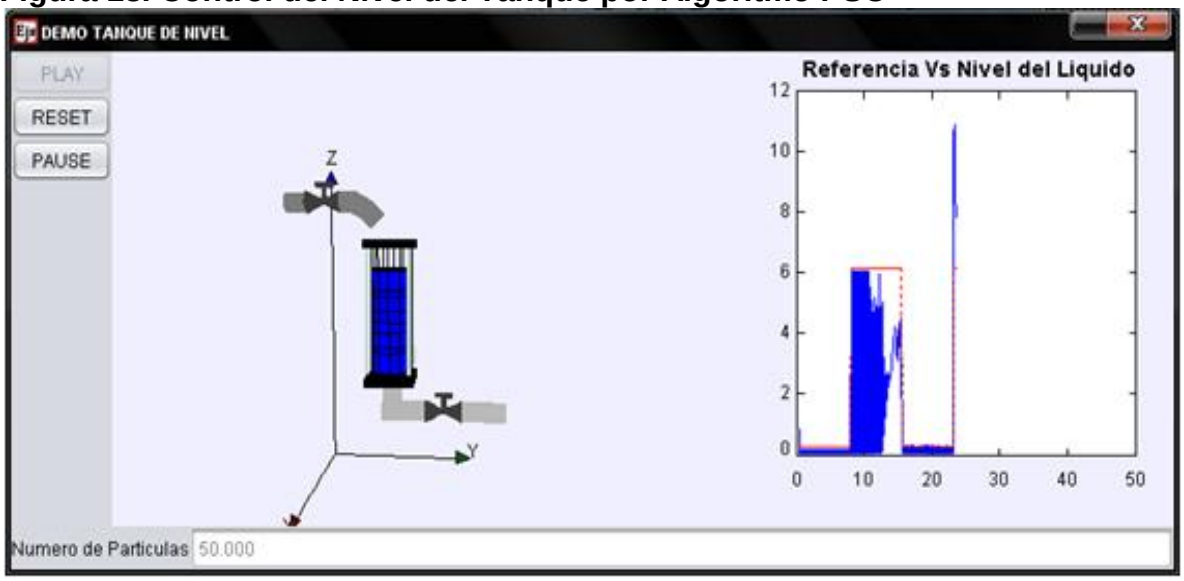
Fuente: Realizada por el autor.

En la figura 18 podemos observar el seguimiento de la señal a este impulso para así controlar el nivel del tanque.

Este modulo al igual que los demás consta de una botonera esencial, con los botones Play, Reset y Pause. Además de la posibilidad de ingresarle la Cantidad de agentes que actuaran en el algoritmo.

4.1.2. Modulo Demos BSFO Siguiendo el esquema general de interfaz gráfica propuesta en la sección **3.3.3.1 Figura 12**, se crea una ventana con los cuatro paneles principales. Al abrir el Modulo Demo de BSFO, el usuario se encuentra con una pantalla de inicio, la cual al igual que el modulo PSO le proporciona la posibilidad de elegir alguno de los dos demos: Demo 2D, Demo 3D o el Demo Control Nivel de un Tanque. En la figura 19 se puede apreciar esta pantalla de inicio.

Figura 18. Control del Nivel del Tanque por Algoritmo PSO



Fuente: Realizada por el autor.

Figura 19. Pantalla de inicio modulo Demo BSFO

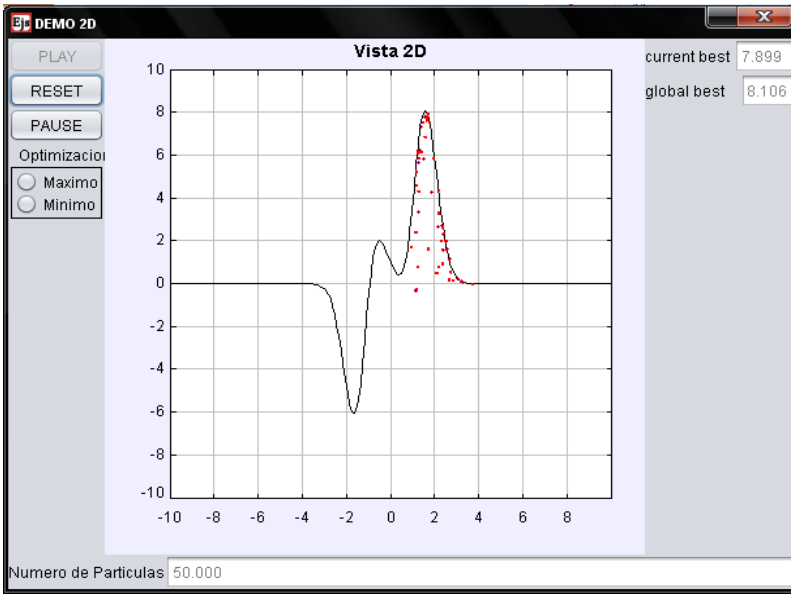
Fuente: Realizada por el autor.

4.1.2.1 Modulo Demo 2D BSFO Este modulo muestra una optimización de una función de de una sola variable dependiente. La función a optimizar escogida fue la función “Peaks”, escogida por sus picos positivos y negativos de la misma amplitud, pudiendo así mostrar al usuario una optimización de dicha función en su máximo o mínimo punto. En la figura 20 se puede apreciar el modulo.

El Usuario podrá escoger si la optimización será de un mínimo o será de un máximo, además de la cantidad de partículas que conformaran el enjambre.

Este modulo al igual que los demás consta de una botonera esencial, con los botones Play, Reset y Pause. Además de la posibilidad de ingresarle la Cantidad de agentes que actuaran en el algoritmo. También contiene la visualización de datos de la mejor posición encontrada por el Algoritmo.

Figura 20. Modulo Demo BSFO 2D



Fuente: Realizada por el autor.

4.1.2.2. Modulo Demo 3D BSFO Este modulo muestra una optimización de una función de dos variables dependientes mediante el algoritmo BSFO. En la figura 20 se puede apreciar el modulo.

El Usuario podrá escoger si la optimización será de un mínimo o será de un máximo, además de la cantidad de partículas que conformaran el enjambre.

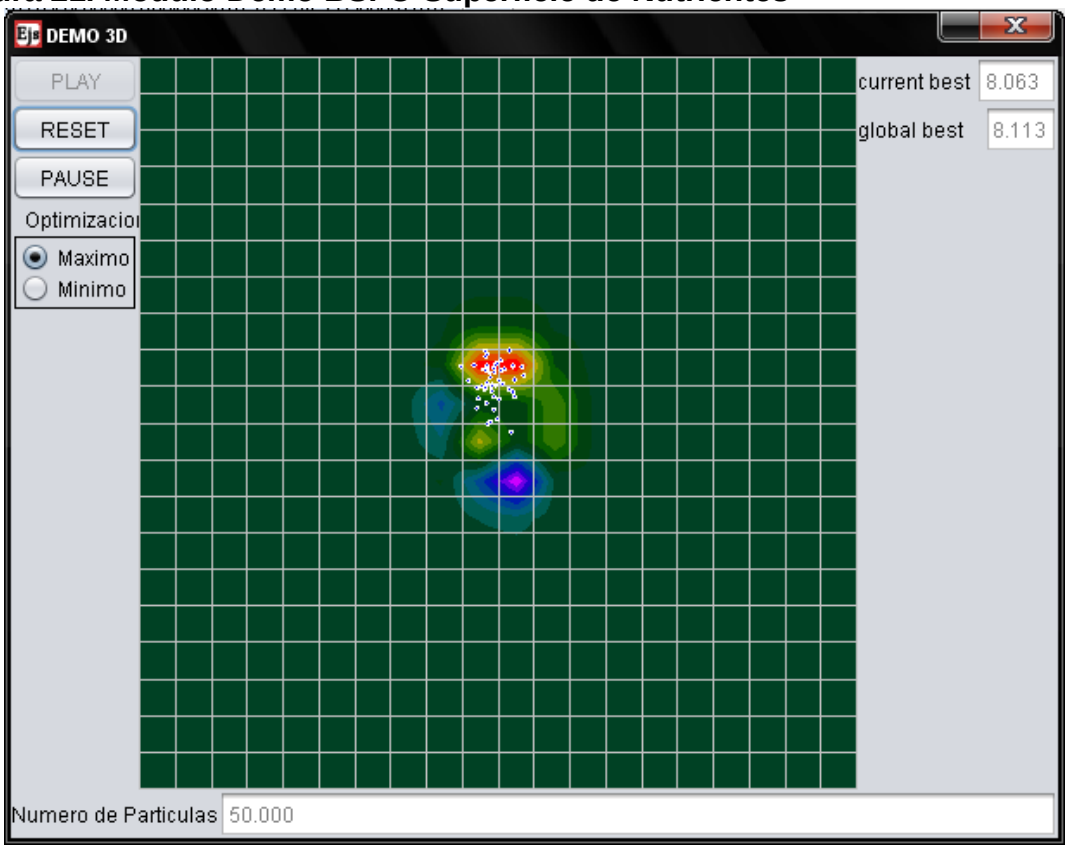
Este modulo al igual que los demás consta de una botonera esencial, con los botones Play, Reset y Pause. Además de la posibilidad de ingresarle la Cantidad de agentes que actuaran en el algoritmo. También contiene la visualización de datos de la mejor posición encontrada por el Algoritmo.

4.1.2.2. Modulo Demo Control de Nivel del Tanque Este modulo muestra el control del nivel de agua de un tanque, por medio del algoritmo BSFO. Ver figura 17.

Este demo expone el comportamiento del BSFO ante cambios de estimulo de control con una señal de pulsos, simulando la orden de llenado y vaciado del tanque.

En la figura 22 podemos observar el seguimiento de la señal a este impulso para así controlar el nivel del tanque.

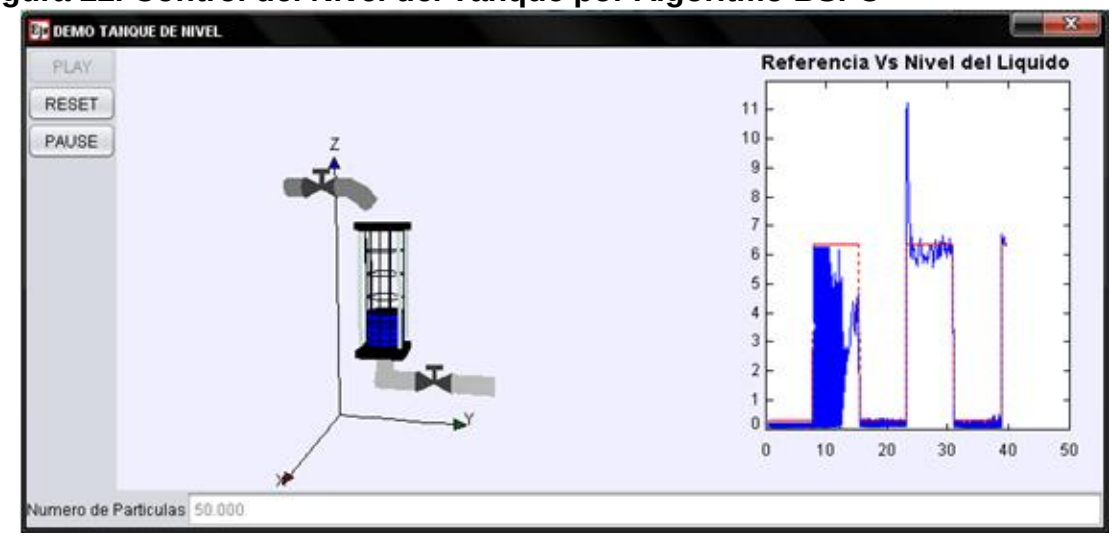
Figura 21. Modulo Demo BSFO Superficie de Nutrientes



Fuente: Realizada por el autor.

Este modulo al igual que los demás consta de una botonera esencial, con los botones Play, Reset y Pause. Además de la posibilidad de ingresarle la Cantidad de agentes que actuaran en el algoritmo.

Figura 22. Control del Nivel del Tanque por Algoritmo BSFO



Fuente: Realizada por el autor.

4.2. SEGUNDA ETAPA: MODULOS GENERICOS

Esta segunda etapa consiste en módulos flexibles que permiten la modificación de la función de optimización, así generando posibilidades para la creación de prácticas y/o laboratorios para una mayor comprensión de la inteligencia de enjambres, facilitando la enseñanza por parte del docente de la materia correspondiente.

Este modulo esta divido en dos partes: Modulo Genérico PSO y Modulo Genérico BSFO. Como se puede observar en la figura 23 la cual presenta la pantalla de inicio del modulo genérico.

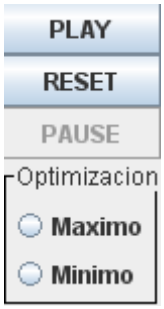
4.2.1. Modulo Genérico PSO Siguiendo el esquema general de interfaz gráfica propuesta en la sección **3.3.3.1 Figura 12**, se crea una ventana con los cuatro paneles principales. Al abrir este modulo el usuario se encontrara con una ventana que consta de 4 partes: panel de simulación, registros gráficos, panel de resultados y el panel de variables interactivas.

- *Panel de Simulación:* Esta contiene la botonera esencial; Play, Reset y Pause. Además de dos *RadioBotones* que permite al usuario elegir si la optimización será bajo un punto máximo o un punto mínimo. Ver figura 24.

Figura 23. Página Principal Modulo Genérico

Fuente: Realizada por el autor.

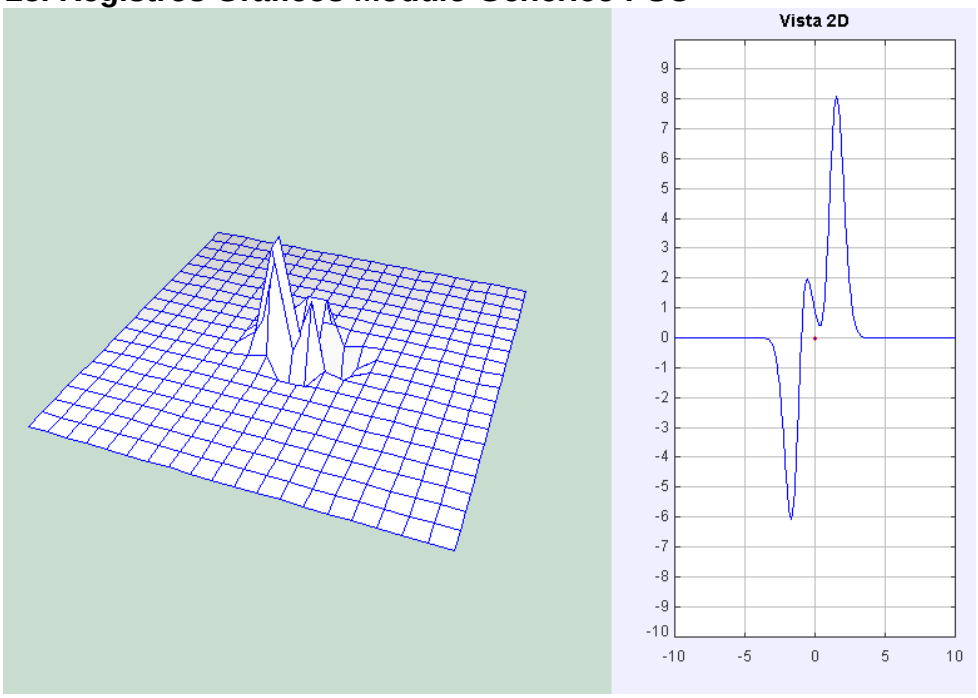
Figura 24. Panel de Simulación Modulo Genérico PSO.



Fuente: Realizada por el autor.

- *Registros Gráficos:* Esta contiene dos gráficos, uno en 3D para mostrar los resultados de funciones de optimización de dos variables, y una en 2D para mostrar resultados de funciones de optimización de una sola variable. Ver figura 25.

Figura 25. Registros Gráficos Modulo Genérico PSO



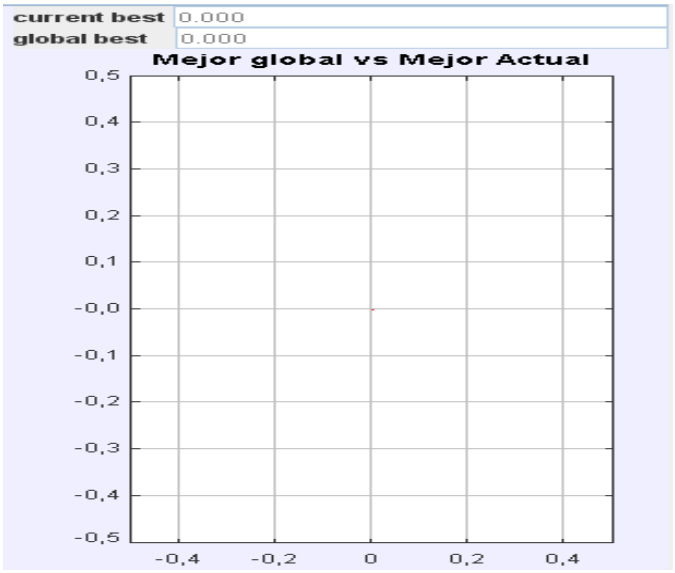
Fuente: Realizada por el autor.

- *Panel de Resultados*: Esta contiene los datos obtenidos por la optimización: La mejor posición local y la mejor posición global, además de sus respectivas graficas. Ver Figura 26

Panel de Variables Interactivas Esta contiene los campos numéricos usados para modificar el número de individuos del enjambre y modificar la función de optimización de una o dos variables, en caso de ser de una sola variable, deberá ser tomada en función de y. Ver figura 27

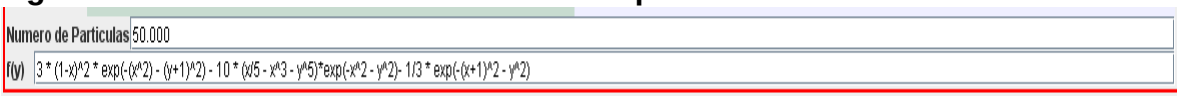
Al unir estas cuatro partes obtenemos El modulo Genérico PSO el cual puede ser observado en la figura 28

Figura 26. Panel de Resultados Modulo Genérico PSO



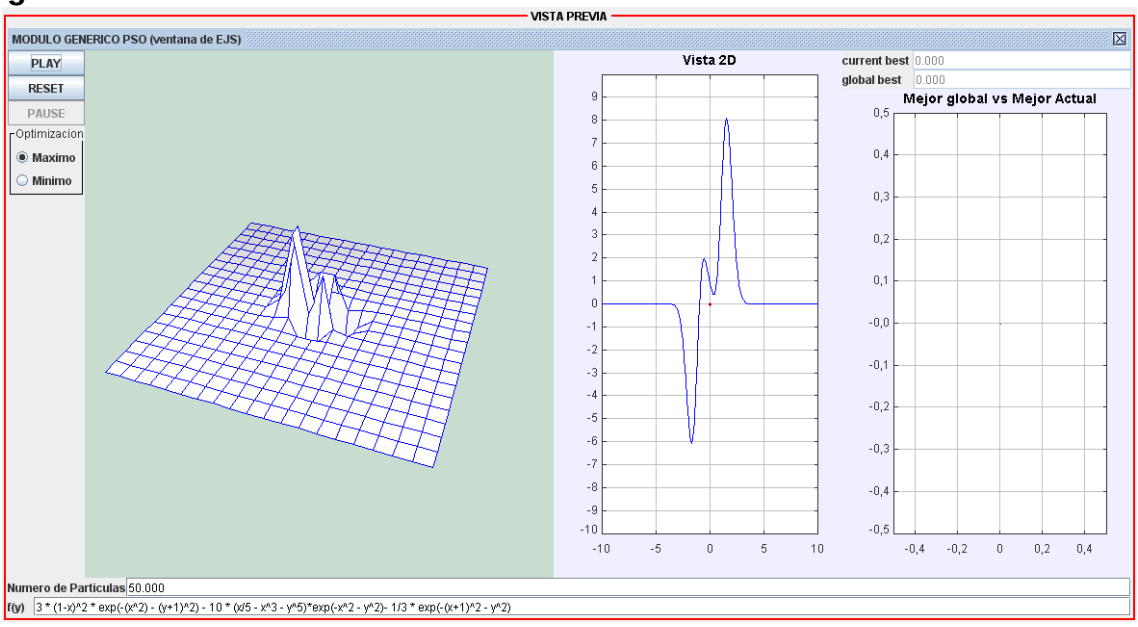
Fuente: Realizada por el autor.

Figura 27. Panel de Variables Interactivas para Modulo Genérico PSO



Fuente: Realizada por el autor.

Figura 28. Modulo Genérico PSO

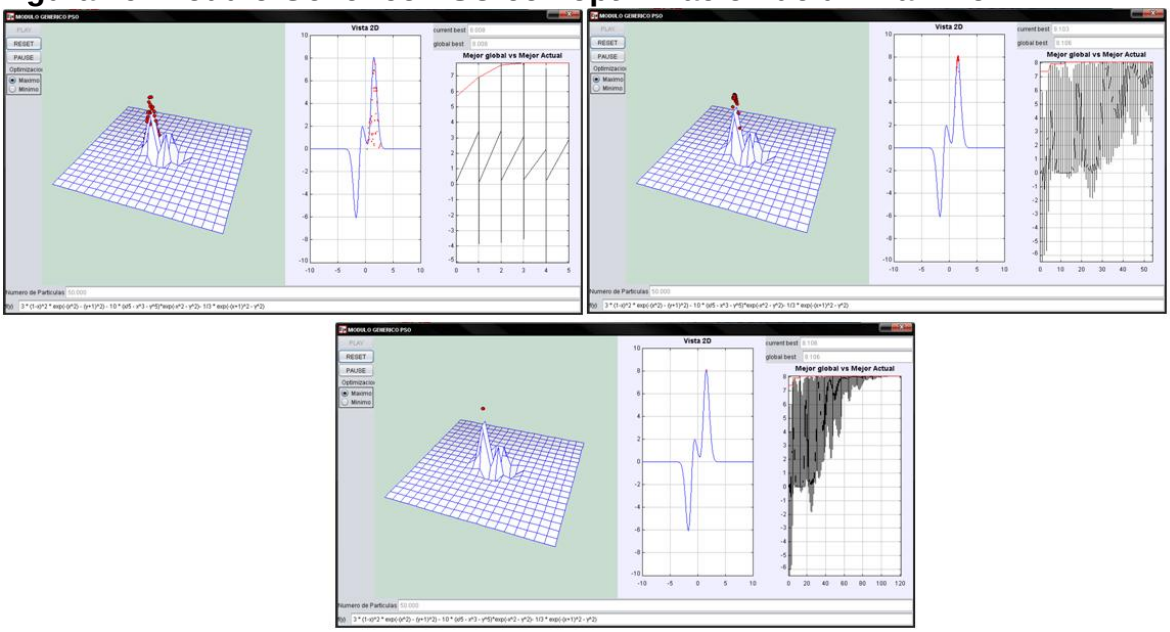


Fuente: Realizada por el autor.

El programa puede verse funcionando siguiendo el punto máximo de la función Peaks en la figura 29.

El programa puede verse funcionando siguiendo el punto mínimo de la función Peaks en la figura 30.

Figura 29. Modulo Genérico PSO con optimización de un máximo

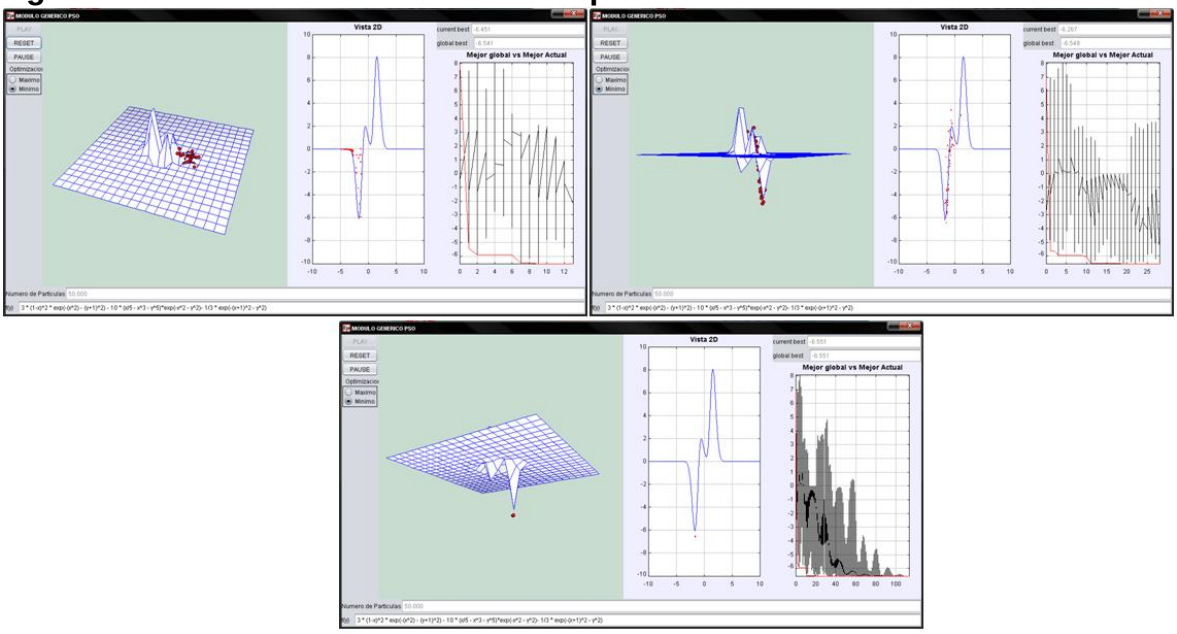


Fuente: Realizada por el autor.

4.2.2. Modulo Genérico BSFO Siguiendo el esquema general de interfaz gráfica propuesta en la sección **3.3.3.1 Figura 12**, se crea una ventana con los cuatro paneles principales. Al abrir este modulo el usuario se encontrara con una ventana que consta de 4 partes: panel de simulación, registros gráficos, panel de resultados y el panel de variables interactivas.

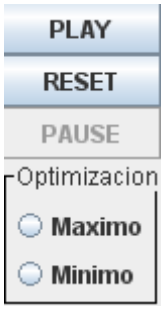
- *Panel de Simulación:* Esta contiene la botonera esencial; Play, Reset y Pause. Además de dos *RadioBotones* que permite al usuario elegir si la optimización será bajo un punto máximo o un punto mínimo. Ver figura 31.

Figura 30. Modulo Genérico PSO con optimización de un mínimo



Fuente: Realizada por el autor.

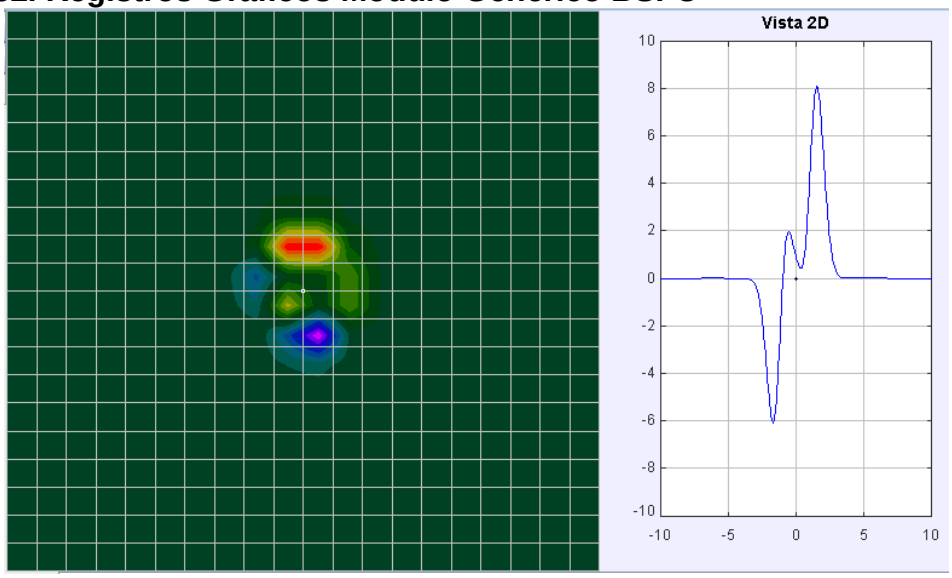
Figura 31. Panel de Simulación Modulo Genérico BSFO.



Fuente: Realizada por el autor.

- *Registros Gráficos:* Esta contiene dos gráficos, uno en campo escalar simulando la cantidad de proteínas y ambientes hostiles para las bacterias para mostrar los resultados de funciones de optimización de dos variables, y una en Plano cartesiano para mostrar resultados de funciones de optimización de una sola variable. Ver figura 32.

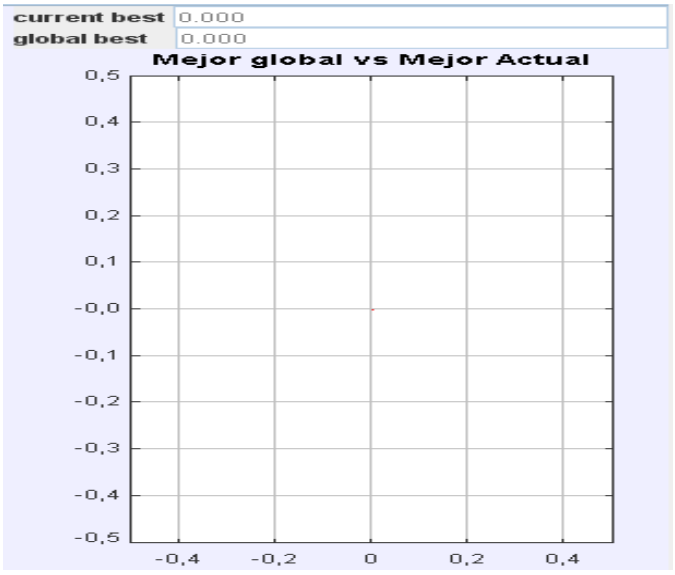
Figura 32. Registros Gráficos Modulo Genérico BSFO



Fuente: Realizada por el autor.

- *Panel de Resultados*: Esta contiene los datos obtenidos por la optimización: La mejor posición local y la mejor posición global, además de sus respectivas graficas. Ver Figura 33

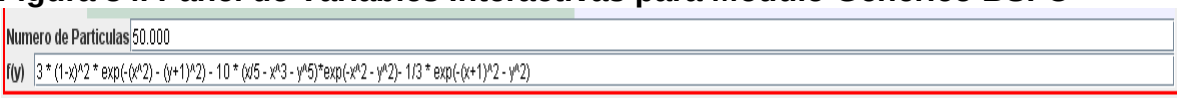
Figura 33. Panel de Resultados Modulo Genérico BSFO



Fuente: Realizada por el autor.

Panel de Variables Interactivas Esta contiene los campos numéricos usados para modificar el número de individuos del enjambre y modificar la función de optimización de una o dos variables, en caso de ser de una sola variable, deberá ser tomada en función de y. Ver figura 34

Figura 34. Panel de Variables Interactivas para Modulo Genérico BSFO

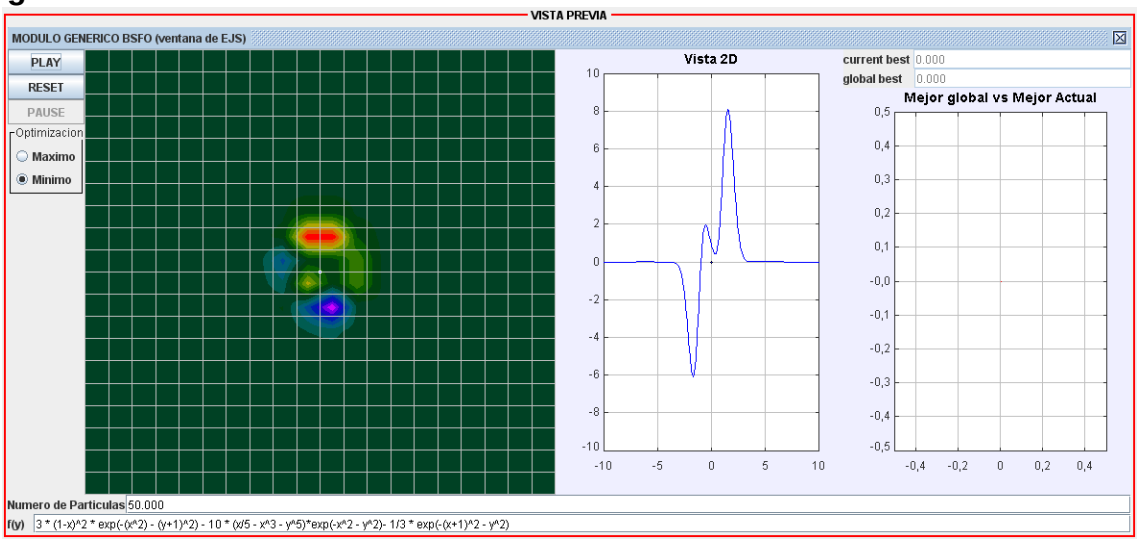


Fuente: Realizada por el autor.

AL unir estas cuatro partes obtenemos El modulo Genérico BSFO el cual puede ser observado en la figura. 35

El programa puede verse funcionando siguiendo el punto máximo de la función Peaks en la figura 36.

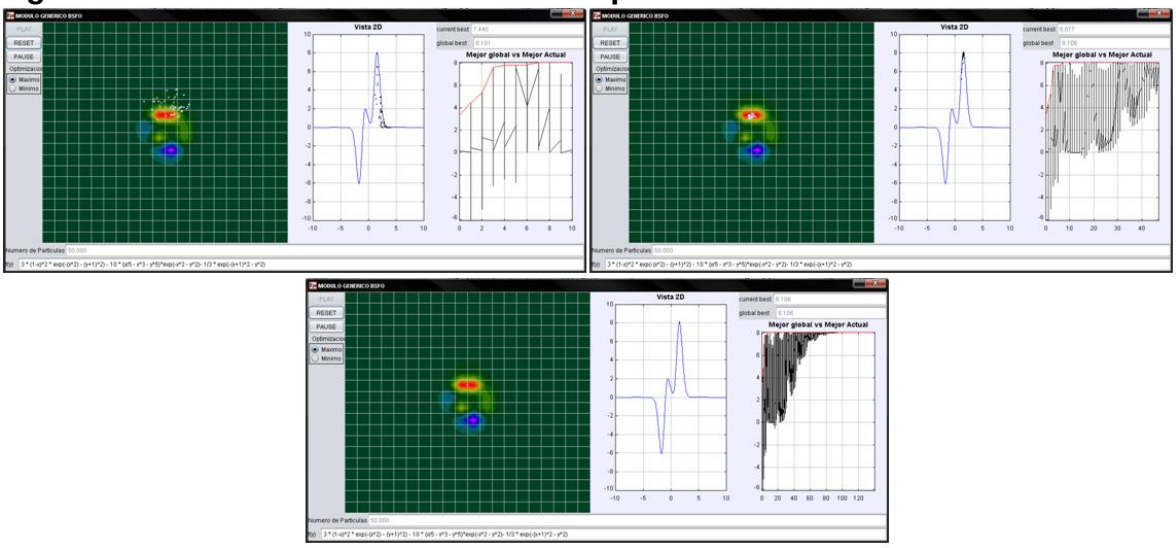
Figura 35. Modulo Genérico PSO



Fuente: Realizada por el autor.

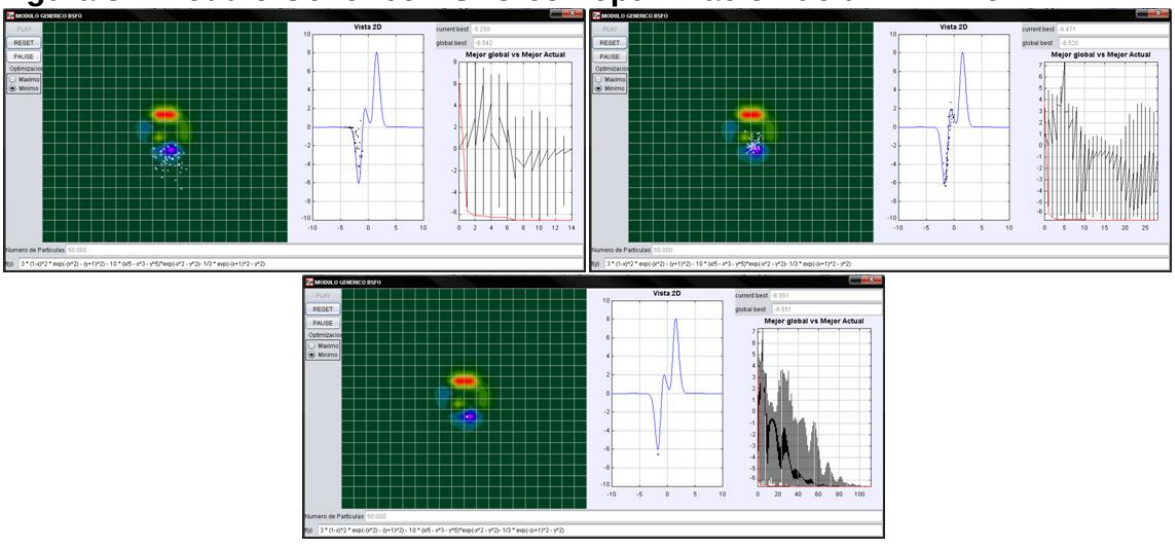
El programa puede verse funcionando siguiendo el punto mínimo de la función Peaks en la figura 37.

Figura 36. Modulo Genérico BSFO con optimización de un máximo



Fuente: Realizada por el autor.

Figura 37. Modulo Genérico BSFO con optimización de un mínimo



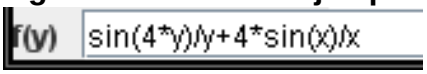
Fuente: Realizada por el autor.

4.2.3 Ejemplos Aplicativos Modulo Genérico La experiencia en la aplicación consiste en:

Al empezar el usuario se encontrara con una pantalla de inicio como en la figura 23. Donde el usuario puede elegir el algoritmo de optimización entre PSO y BSFO.

4.2.3.1 Ejemplo Aplicativo Modulo Genérico PSO Al elegir el modulo PSO, el usuario se encuentra con la ventana de la figura 28. Lo primero que el usuario debe hacer es digitar la función a optimizar ya sea de una o dos variables, en este caso de ejemplo se digito una función como en la figura 38, esta función nos muestra una grafica como en la figura 39

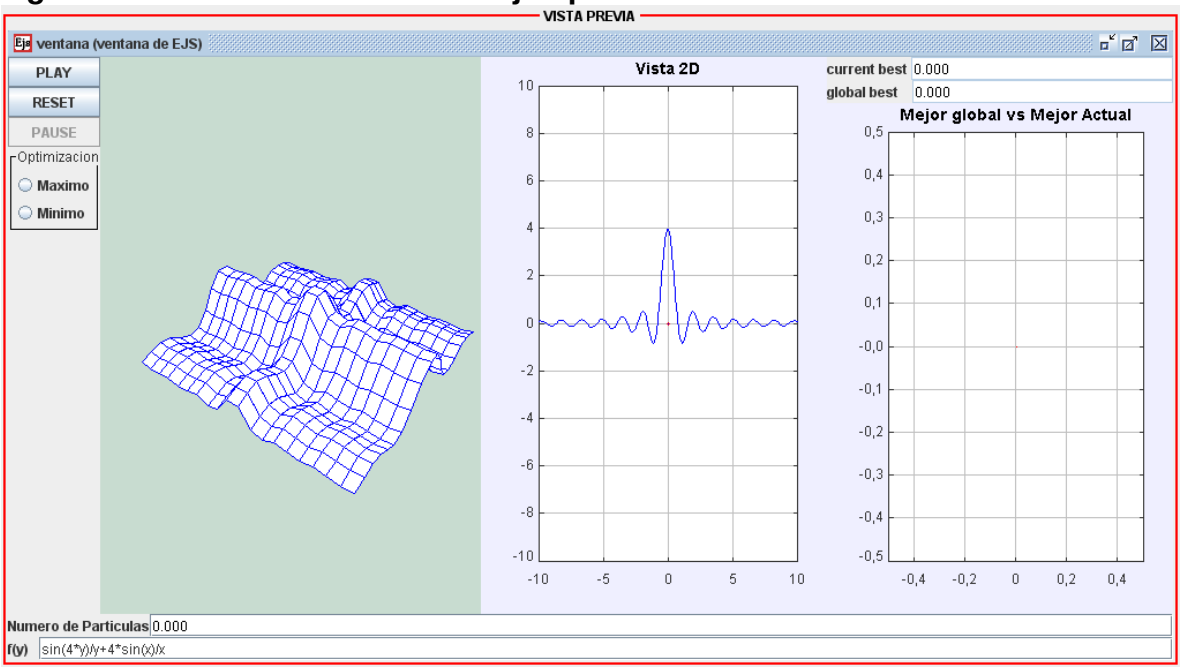
Figura 38. Función Ejemplo PSO



Fuente: Realizada por el autor.

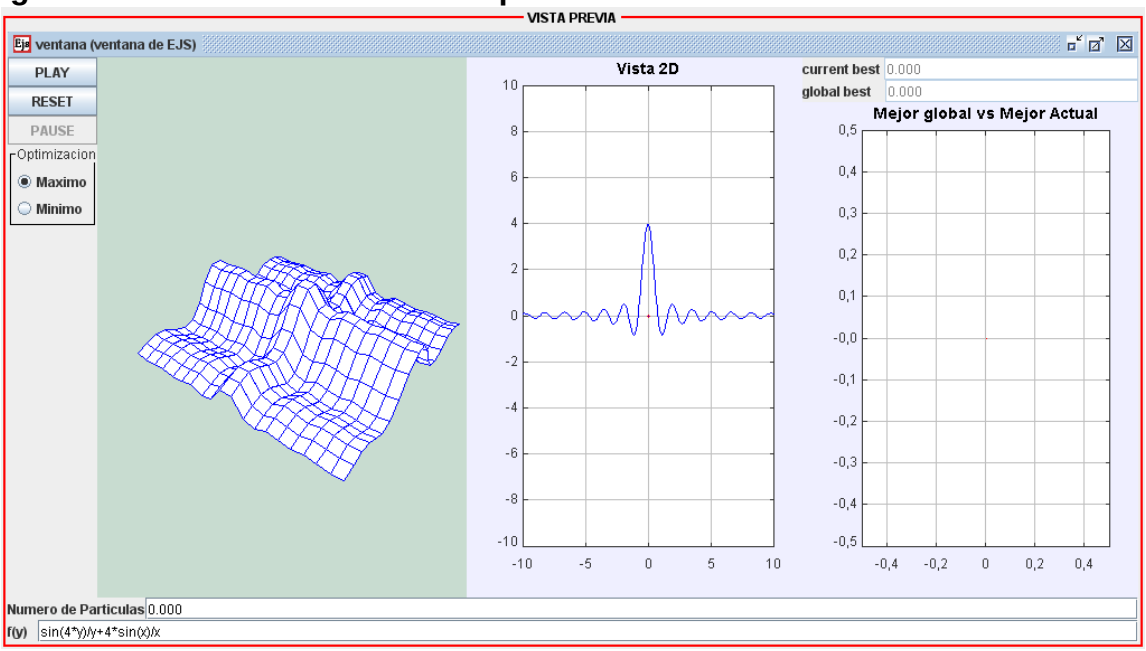
Paso seguido debemos elegimos si se quiere optimizar el mínimo o el máximo de la función, como en este caso la función ingresada fue una modificación para 3D de la función seno cardinal función la cual se caracteriza por poseer un punto máximo, para la muestra se elige buscar el máximo de dicha función tal como se observa en la figura 40.

Figura 39. Grafica de la Función Ejemplo PSO



Fuente: Realizada por el autor.

Figura 40. Elección de Punto de Optimización de la Grafica PSO

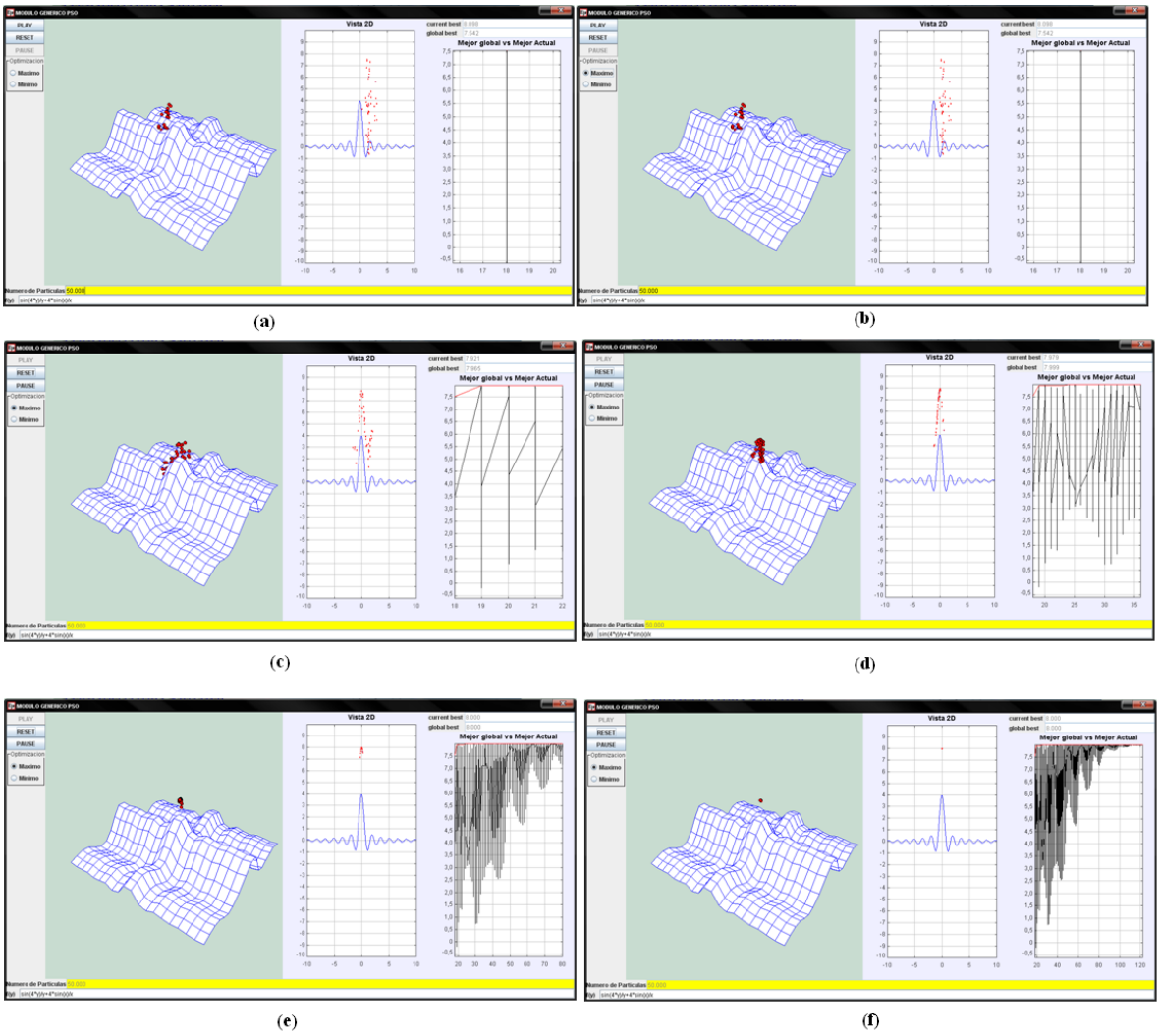


Fuente: Realizada por el autor.

Al ser presionado el botón Play vemos la animación de cómo las partículas poco a poco van encontrando el punto Máximo de la función elegida tal como lo muestra la figura 41.

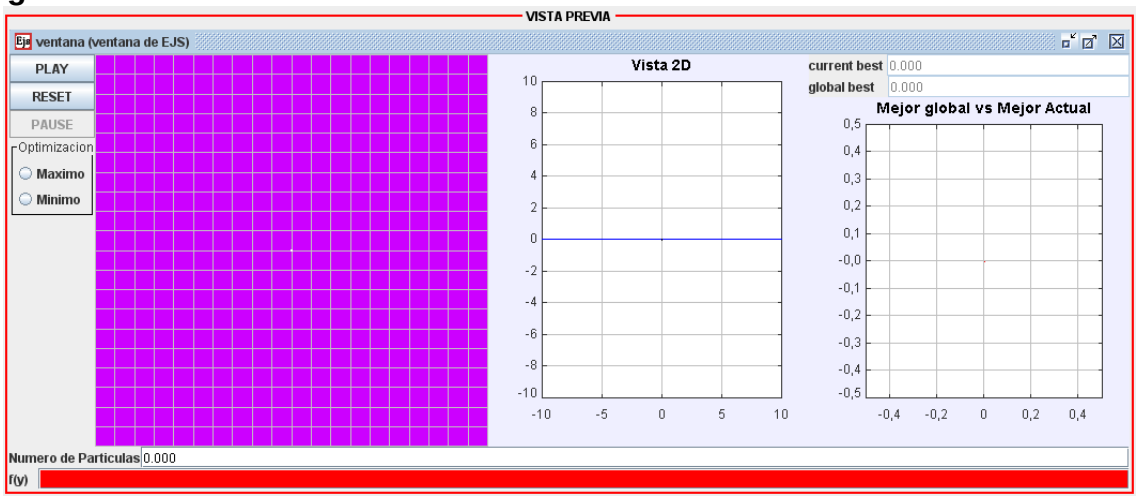
4.2.3.2 Ejemplo Aplicativo Modulo Genérico BSFO Al elegir el modulo BSFO, el usuario se encuentra con la ventana de la figura 42. Lo primero que el usuario debe hacer es digitar la función a optimizar ya sea de una o dos variables, en este caso de ejemplo se digito una función que se muestra en la figura 43, esta función nos muestra una grafica como en la figura 45

Figura 41. Secuencia de Imágenes de la Optimización de la Función Ejemplo PSO



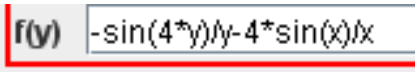
Fuente: Realizada por el autor.

Figura 42. Ventana Modulo Genérico BSFO



Fuente: Realizada por el autor.

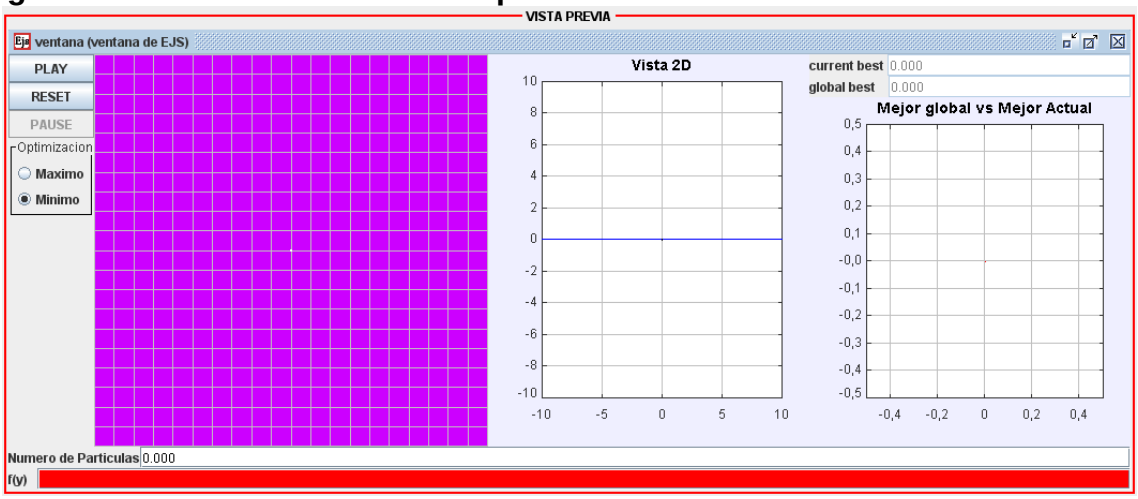
Figura 43. Función Ejemplo BSFO



Fuente: Realizada por el autor.

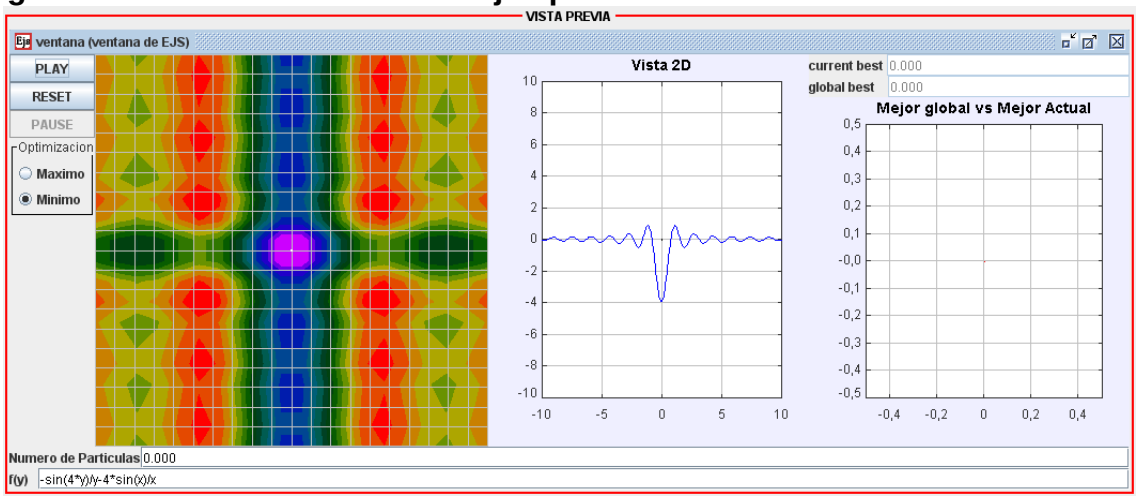
Paso seguido debemos elegimos si se quiere optimizar el mínimo o el máximo de la función, como en este caso la función ingresada fue una modificación para 3D de la función seno cardinal negativo elegimos un mínimo, tal como se observa en la figura 44.

Figura 44. Elección de Punto de Optimización del Modulo Genérico BSFO



Fuente: Realizada por el autor.

Figura 45. Grafica de la Función Ejemplo BSFO



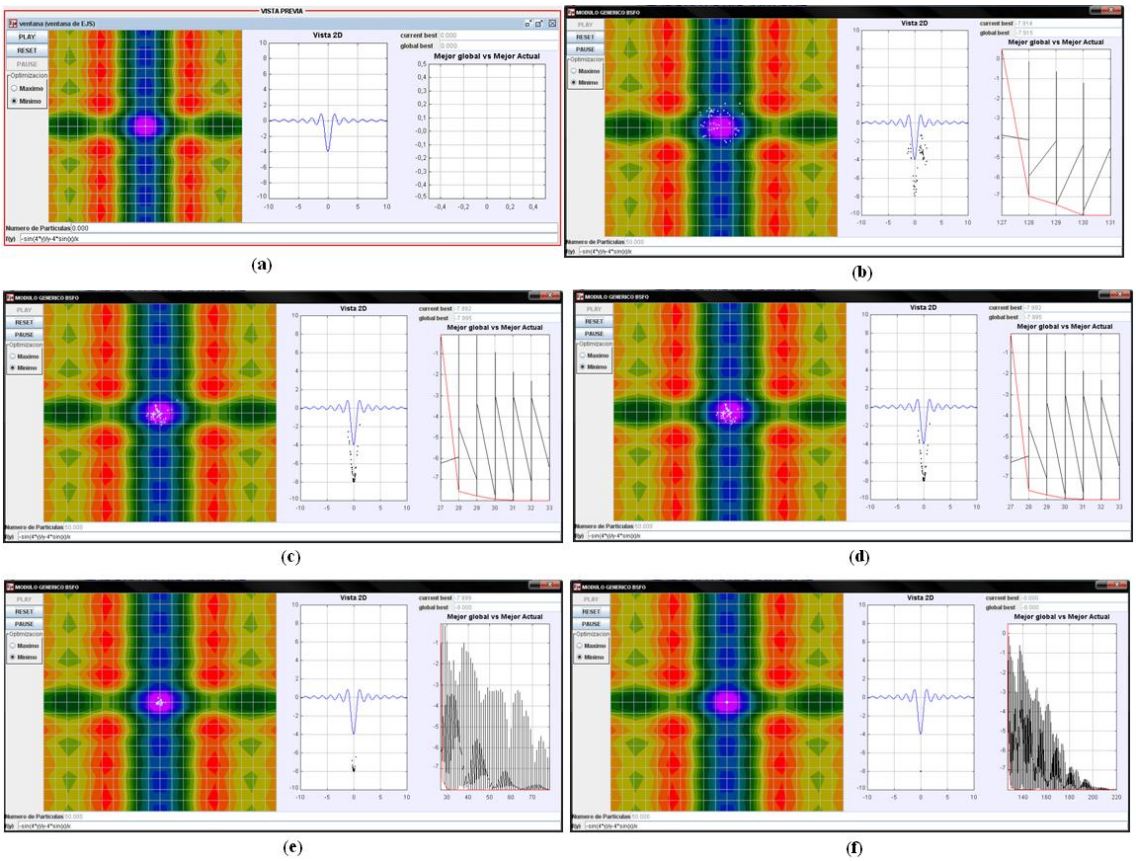
Fuente: Realizada por el autor.

Al ser presionado el botón Play vemos la animación de cómo las partículas poco a poco van encontrando el punto Máximo de la función elegida tal como lo muestra la figura 46.

4.2.4 Diseño de la Página Web Como se ha comentado anteriormente los módulos fueron montados en una página WEB, la cual será mostrada brevemente a continuación.

Al ingresar a la pagina que contiene el laboratorio virtual de inteligencia de enjambres, el usuario se encuentra con una página de introducción (Ver Figura 47) la cual es meramente estética, al presionar iniciar el usuario será dirigido a la pagina que almacena la teoría general de Inteligencia de enjambres (Ver Figura 48).

Figura 46. Secuencia de Imágenes de la Optimización de la Función Ejemplo BSFO

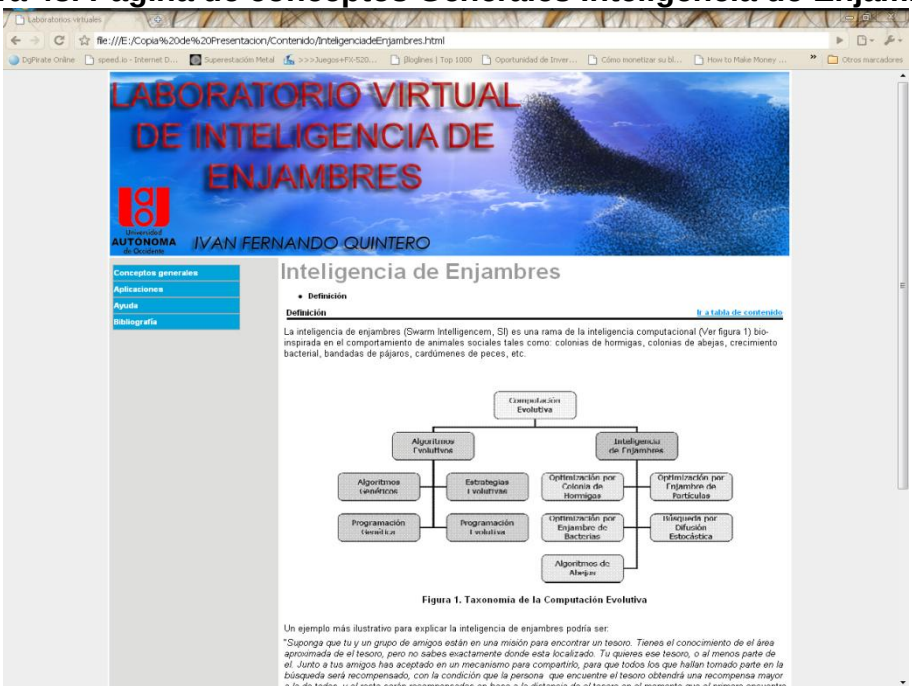


Fuente: Realizada por el autor.

Figura 47. Página Introducción Laboratorio Virtual de Enjambres

Fuente: Realizada por el autor.

Figura 48. Pagina de conceptos Generales Inteligencia de Enjambres



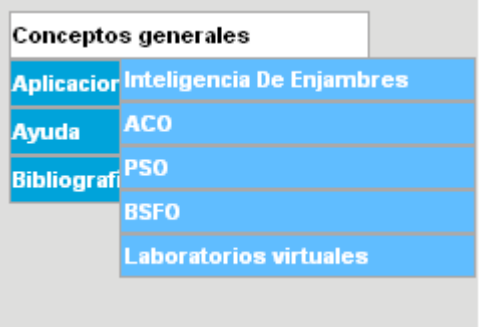
Fuente: Realizada por el autor.

En la parte izquierda está ubicado el menú de la pagina, donde el usuario puede navegar libremente entre la teoría módulos ayudas y bibliografías.

El menú de Conceptos Generales contiene un submenú (Ver figura 49), en los cuales se explica de manera general la teoría necesaria para comprender la inteligencia de enjambres y sus métodos de optimización.

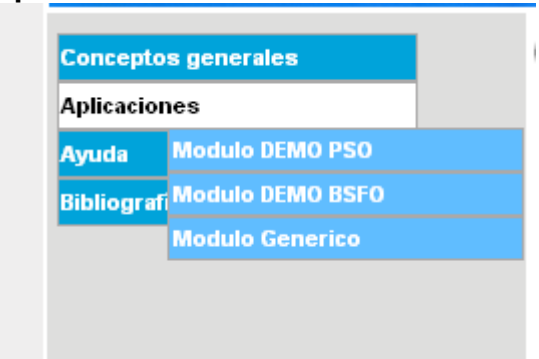
El menú de Aplicaciones contiene un submenú (Ver figura 50), en el cual el usuario puede aplicar los conocimientos generados en la teoría para ya sea ver algunos demos (Ver figura 51) o experimentar con los módulos genéricos (Ver Figura 52).

Figura 49. Submenú Conceptos Generados



Fuente: Realizada por el autor.

Figura 50. Submenú Aplicaciones



Fuente: Realizada por el autor.

Figura 51. Página Introducción Modulo Demo BSFO

Fuente: Realizada por el autor.

Figura 52. Página Introducción Modulo Genérico

Fuente: Realizada por el autor.

El menú de ayuda Contiene un manual de usuario para el manejo de los módulos, explicándolo detalladamente (Ver Figura 53) para una mejor experiencia del usuario con el laboratorio.

Por último el menú Bibliografía como su nombre lo dice, contiene toda la bibliografía necesaria para una compresión más a fondo de la inteligencia de enjambres (Ver Figura 54).

Figura 53. Pagina Ayuda o Manual de Usuario

Fuente: Realizada por el autor.

Figura 54. Pagina Bibliografia

Fuente: Realizada por el autor.

5.0. CONCLUSIONES Y TRABAJO FUTURO

5.1. CONCLUSIONES.

En este proyecto los fundamentos teóricos sobre La inteligencia de Enjambres, específicamente los algoritmos de Optimización por Enjambres de partículas y Optimización por Enjambres de Bacterias fueron requisitos necesarios para el desarrollo del mismo, además de tener la capacidad de diseñar nuevas soluciones para su funcionamiento aplicativo.

Existen varias maneras de implementar un laboratorio virtual, sin embargo en este proyecto se decidió trabajar con Easy Java Simulations que es una herramienta muy usada para desarrollar aplicaciones en física ya que su ventaja principal es la facilidad de implementar interfaces con el usuario bastante amigables y además, la posibilidad de generar las páginas WEB con las aplicaciones embebidas usando applets.

Los algoritmos PSO y BSFO de la inteligencia de enjambres demostraron ser excelentes en la optimización de funciones complejas, debido a que ofrece resultados óptimos en un relativo corto tiempo a un bajo costo computacional.

Debido a la complejidad multidimensional del algoritmo BSFO, se presentaron inconsistencias en el diseño por lo que se tuvo que revaluar esta fase, retrasando así el cronograma previsto.

En el algoritmo BSFO el autor trabaja con matrices mayores a 2 dimensiones debido a que Matlab facilita esta codificación, cosa contraria ocurre en Easy Java Simulations, donde trabajar con Matrices mayores a 2 dimensiones es demasiado complejo, por esto el algoritmo tuvo que ser reducido disminuyendo la cantidad de variables con las que trabajo el autor.

En el proceso de diseño de las aplicaciones que conforman el laboratorio virtual implementado, se determino por estandarizar la interfaz gráfica de las diferentes simulaciones para así permitir una interacción con el usuario más amigable e intuitiva, ya que de esta manera se utiliza un mismo ambiente en todas las aplicaciones del laboratorio virtual.

En el Demo “control del nivel del tanque”, podemos apreciar claramente la posibilidad que tienen estos algoritmos para aplicaciones de control en la ingeniería.

Finalmente se concluye que para propósitos pedagógicos, es viable diseñar y utilizar recursos interactivos que brinden apoyo a la docencia y a la práctica de la Inteligencia de enjambres sin necesidad de ser un especialista en computación, siempre y cuando se manejen de forma clara los conceptos teóricos que se desean exponer y se tengan los conocimientos suficientes sobre el modelado y simulación.

5.2 .TRABAJO FUTURO

Como trabajo futuro se plantea la posibilidad de adicionar los otros algoritmos de Inteligencia de Enjambres, como: Optimización por Colonia de Hormigas, Búsqueda por difusión Estocástica, y Algoritmos de Abejas.

Además, se espera realizar el control de dos o más plantas de la universidad por medio de los algoritmos PSO y BSFO, para así apreciar la aplicación en ingeniería que ofrecen estos métodos.

Se espera realizar funcionalidades especiales para los docentes del área de Inteligencia de Enjambres, con el fin de ampliar sus capacidades de orientar y controlar el avance de sus estudiantes.

BIBLIOGRAFIA

[1] DORIGO, M. DI CARO, G. GAMBARDELLA, L. “Ant Algorithms for Discrete Optimization”. En: Artificial Life. Vol. 5 Nº 2, p. 137-172. 1999.

[2] DORIGO, M. MANIEZZO, V. COLORNI, A. “Ant System: Optimization by a Colony of Cooperative Agents”. En: IEEE Transactions on System, Man and Cybernetics- Part B Vol. 26 Nº 1, p. 29-41. 1996.

[3] DORIGO, M. MANIEZZO, V. COLORNI, A. “Possitive feedback as a search strategy”. Technical Report Nº 91-016. Dipartimento di Elettronica Politecnico di Milano. Italy. 1991.

[4] DORIGO, M. GAMBARDELLA, L.. “Ant Colonies for the Traveling Salesman Problem”. En: BioSystems, Vol. 43, p. 73-81. 1997.

[5] ENGELBRECTH, Andries P. Computational Intelligence An Introduction. John Wiley & Sons, LTd. Segunda Edicion, 2007. 597p.

[6] KENNEDY, J. y EBERHART, R. Particle Swarm Optimization. En.: Memorias del 1995 IEEE International Conference on Neural Networks. 1995, p. 1942-1948.

[7] KENNEDY, J. y EBERHART, R. Swarm Intelligence. Morgan Kaufmann Publishers, 2001.

[8] KENNEDY, J. y EBERHART, R. Particle swarm optimization. Memorias del IEEE International Conference on Neural Networks, 1995, p. 1942–1498.

[9] Laboratorio virtual [en línea]. Madrid: Departamento de informática y automática UNED [consultado 1 abril de 2010]. Disponible en Internet: http://lab.dia.uned.es/rlab/contenido/labvirtual.html?page=3

[10] LYHNE, Anders C (et al). Morphogenesis: Shaping Swarms of Intelligence Robots. A video for the 22nd conference on Artificial Intelligence (AAAI-07) [Consultado el 10 de diciembre de 2010]. Disponible en internet: <http://iridia.ulb.ac.be/~alyhne/aaai-07/>

[11] MARTIN, Carla, DORMIDO, Sebastián, URQUIA, Alfonso. Modelado orientado a objetos de laboratorios virtuales con aplicación a la enseñanza del control de procesos químicos [en línea]. Madrid: UNED, 2005 [consultado 1 abril de 2010]. Disponible en Internet: http://e-spacio.uned.es/fez/view.php?pid=bibliuned:733.

[12] MUÑOZ, Mario A. LOPEZ, Jesús A. CAICEDO, Eduardo F. Inteligencia de enjambres: sociedades para la solución de problemas (una revisión). Revista de ingeniería en investigación, Vol 28. No. 2, 2008, p. 119-130.

[13] PASSINO, K. M.: Biomimicry of Bacterial Foraging for Distributed Optimization and.Control. IEEE. Control System Magazine, 2002, p. 52-67.

[14] PASSINO, Kevin. Distributed Optimization and Control Using Only a Germ of Intelligence. Memorias del 2000 IEEE International Symposium on Intelligent Control, 2000, p. 5-13.

[15] REYNOLDS, Craig W. Flocks, Herds, and Schools: A distributed Behavioral Model. ACM Computer Graphics, Vol 21, No 4, 1987, p. 25-34.

[16] SÁNCHEZ, José, MORILLA, Fernando, DORMIDO, Sebastián. Laboratorios virtuales y remotos para la práctica a distancia de la automática [en línea]. Madrid: Departamento de informática y automática UNED, 2000 [consultado 1 abril de 2010]. Disponible en Internet: http://e-spacio.uned.es/fez/view.php?pid=bibliuned:783.