DGÉNESIS - SOFTWARE PARA LA EJECUCIÓN DE ALGORITMOS GENÉTICOS DISTRIBUIDOS

Marcelo Mejía Olvera* y Erick Cantú Paz

Departamento Académico de Computación
Instituto Tecnológico Autónomo de México
Rio Hondo No. 1, Col. Tizapán San Angel
C.P. 01000 México, D.F.

Resumen: La mayoría de los sistemas que existen para experimentar con Algoritmos Genéticos Paralelos están diseñados para explotar la arquitectura de una computadora paralela específica, implementan un criterio fijo de migración y no son fácilmente accesibles. El presente trabajo introduce el sistema DGÉNESIS desarrollado en el ITAM para ejecutar Algoritmos Genéticos Paralelos sobre una red de estaciones de trabajo heterogéneas. DGÉNESIS permite al usuario especificar y modificar los parámetros que controlan la migración fácilmente. El sistema está disponible para cualquier persona interesada vía ftp anónimo.

Palabras clave: Algoritmos Genéticos, Procesamiento Paralelo, Sistemas Distribuidos.

1. INTRODUCCIÓN

Los Algoritmos Genéticos (AGs) son métodos de búsqueda y optimización que imitan el proceso de selección natural manipulando una población de individuos que contienen valores posibles de las variables de un problema específico [1]. Los AGs evalúan a los individuos y escogen a los más aptos para reproducirse, actúan sobre la población utilizando los operadores genéticos de cruce y mutación para obtener a los descendientes y formar con ellos una nueva generación. Los AGs utilizan la información histórica contenida en la población para guiar a los individuos hacia regiones prometedoras del espacio de búsqueda. El AG básico se muestra en la figura 1.

\[
\begin{align*}
\text{Generar la población inicial} \\
\text{Evaluación de la población} \\
\text{Mientras los criterios de terminación no se cumplan} \\
\{ \\
\text{Seleccionar a los individuos más aptos} \\
\text{Aplicarles los operadores de cruce y mutación} \\
\text{Evaluación de los nuevos individuos} \\
\}
\end{align*}
\]

Figura 1. Algoritmo Genético básico

En la naturaleza la aptitud se determina por la habilidad de un organismo para sobrevivir a obstáculos que le impiden llegar a la madurez y reproducirse. En
los AGs la función de evaluación es la función objetivo que se desea optimizar y representa el medio ambiente al que cada individuo se enfrenta. El AG debe encontrar a los individuos más aptos, es decir los que representan valores óptimos. La función de evaluación es el único componente del AG dependiente del problema que se está resolviendo, basta con cambiarla para cambiar el dominio del problema. El AG desconoce el proceso de evaluación y sólo utiliza la información cuantitativa provista por esta función como medida de bondad de los individuos.

Antes de poder evaluar a un individuo es necesario decodificar la información que contiene para encontrar el valor de las variables que se utilizan como entradas a la función de evaluación. Dado que la codificación es arbitraria, cada función de evaluación debe tener la capacidad de extraer la información de los cromosomas. Haciendo una analogía con la genética natural, el individuo que manipula el AG (la cadena con la información codificada) es el genotipo, la interpretación del genotipo que se evalúa es el fenotipo, y la función de evaluación es el medio ambiente que mide la aptitud de un individuo desarrollado.

En un AG es necesario evaluar a todos los miembros de la población al principio de cada experimento. Durante la ejecución del algoritmo, sus operadores modifican las estructuras de la población creando nuevos miembros que deben ser evaluados antes de incorporarse a la población. Si el proceso de evaluación de cada individuo requiere de un tiempo considerable de procesamiento, el tiempo de respuesta del algoritmo será muy elevado. Para mejorar el desempeño de los AGs la solución más natural es paralelizarlos.

El objetivo del presente trabajo es introducir el sistema DGENESIS desarrollado en el ITAM para ejecutar Algoritmos Genéticos Paralelos (AGPs) sobre una red de estaciones de trabajo heterogéneas. En la segunda sección del trabajo se establece el marco conceptual de los AGPs, mientras que las principales características de DGENESIS se describen en la tercera sección. Finalmente, se presentan las conclusiones obtenidas durante el desarrollo y uso de DGENESIS.

2. ALGORITMOS GENÉTICOS PARALELOS

Se pueden distinguir cuatro niveles de paralelismo entre los distintos modelos de Algoritmos Genéticos Paralelos (AGPs) propuestos. Ejecutar un AG convencional en una máquina paralela es el primer nivel de paralelismo posible; a pesar de que es muy probable que el compilador paralelize algún cómputo, la arquitectura de la máquina paralela no importa mucho porque el usuario no hace ningún intento por explotarla. El segundo nivel de paralelismo se encuentra cuando la evaluación de los individuos se hace de manera paralela usando explícitamente distintos procesadores, pero manteniendo una sola población. Estos dos primeros niveles de paralelismo no cambian el comportamiento del AG ya que procesan la población de la misma manera que un AG serial.

En el tercero y cuarto niveles se utiliza una forma muy natural de paralelizar un AG: se divide a la población en subpoblaciones, se ejecuta un AG convencional en cada subpoblación y se intercambian periódicamente individuos entre las subpoblaciones. El tercer nivel de paralelismo se explota en los
AGPs de grano burdo. La granularidad se define como una razón entre el tiempo dedicado a procesamiento y el tiempo dedicado a comunicación. Si esta relación es alta se dice que los procesos son de grano burdo. Dado que normalmente se implementa en arquitecturas de memoria distribuida a este modelo se le conoce comúnmente como AG Distribuido [2]. El nivel más alto de paralelismo se encuentra en los AGPs de grano fino. En este modelo la población se distribuye en una cantidad mucho mayor de sub poblaciones que en un AG distribuido y la comunicación es mucho más abundante. Se implanta fácil y eficientemente en máquinas masivamente paralelas con capacidades de comunicación superiores a las arquitecturas con memoria distribuida.

Actualmente existe una gran discusión sobre cual de los últimos dos modelos es mejor. Se han hecho y reportado resultados de experimentos que favorecen a los AGPs de grano burdo (e.g. [3]) y otros que favorecen a los de grano fino (e.g. [4]). Sin embargo, probablemente el AG Distribuido es un mejor modelo de una población natural. Fenómenos como la formación de especies, la interrupción del equilibrio y la migración se pueden simular y observar mejor en un AG de grano burdo que en uno de grano fino. Esta es una de las razones principales por las que en DGENESIS se decidió implementar un AG Distribuido.

El intercambio de individuos entre sub poblaciones se conoce como migración. Al utilizar la migración en un AGP surgen varias cuestiones:

1. ¿Con qué frecuencia migran individuos de una subpoblación a otra?
2. ¿Cuántos individuos migran cada vez?
3. ¿Hacia dónde migran?
4. ¿Qué individuos migran?

A cada una de estas preguntas corresponde un parámetro nuevo para el AGP. La primera pregunta se resuelve introduciendo el intervalo de migración, que se define como el número de generaciones que ocurren entre migraciones. La tasa de migración es la respuesta de la segunda pregunta y define la proporción de la población que migra en un momento dado. Es predecible que un AGP con grandes cantidades de intercambio de individuos entre las sub poblaciones tendrá un desempeño menor que uno que utilice menos los recursos de comunicación. Pero la cantidad de interacción también afecta la efectividad del AG. Un AGP sin comunicación es equivalente en la calidad de la búsqueda a ejecutar varios AGs seriales sobre sub poblaciones pequeñas. En el otro extremo, en un AGP con demasiada comunicación las ventajas de tener sub poblaciones se pierden, los buenos cromosomas de una subpoblación se extienden rápidamente a las demás y las evoluciones pierden independencia.

La tercera pregunta define la topología de comunicación entre las sub poblaciones. A pesar de que se han hecho algunos estudios comparando algunas topologías [3][5][6], hasta ahora no hay estudios concluyentes que demuestren que una topología es mejor que otras en una amplia variedad de problemas. Uno de los objetivos de DGENESIS es precisamente permitir la experimentación con diversas topologías usando una variedad de funciones con características diferentes para caracterizar a las topologías que funcionan mejor con problemas diversos. La última pregunta se refiere al criterio de selección de los migrantes. Aunque se puede seleccionar aleatoriamente a los migrantes, es bastante razonable pensar que en una población natural los individuos que migran son generalmente los más fuertes, pues migrar implica recorrer cierta distancia a través de obstáculos. Por lo tanto en un AGP
normalmente migran los individuos con mejores valores de evaluación. Al enviar migrantes de una subpoblación a otra en un AGP normalmente se envían copias de los individuos seleccionados para migrar y no se elimina a los migrantes de su subpoblación de origen; obviamente este fenómeno no es natural, pero posiblemente ayuda a que el algoritmo converja a una buena solución.

3. DGENESIS

DGENESIS es un sistema desarrollado en el ITAM que permite ejecutar AG Distribuidos en una red de estaciones de trabajo heterogéneas. Después de evaluar los sistemas mostrados en la tabla 1, se decidió construir DGENESIS sobre GENESIS 5.0 por su popularidad en la comunidad de investigadores en el área de AGs y por la sencillez y elegancia de su código. GAucsd y GENESys están basados en GENESIS 4.5, predecesor de la versión elegida, lo que permite suponer que si se desea modificar el sistema desarrollado para incluir algunas propiedades de estos programas, se podrá hacer sencillamente. Genetic Algorithm Workbench y BUGS son sistemas interactivos orientados a demostrar el funcionamiento de un AG y no se pueden utilizar en problemas de optimización de funciones. SGA-Cube y PARAGENESIS requieren de computadoras especiales, por lo que su utilidad se reduce y resultan inadecuados para ser utilizados como base para otros sistemas. PGA 2.4 tiene un mecanismo de migración poco flexible.

<table>
<thead>
<tr>
<th>Sistema</th>
<th>Autor</th>
<th>Lenguaje</th>
<th>S. O. (máquina)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GENESIS 5.0</td>
<td>John Grefenstette [7]</td>
<td>C</td>
<td>UNIX / DOS</td>
</tr>
<tr>
<td>GAucsd 1.4</td>
<td>Nicol Schraudolph [8]</td>
<td>C</td>
<td>UNIX</td>
</tr>
<tr>
<td>GENESys 1.0</td>
<td>Thomas Back [9]</td>
<td>C</td>
<td>UNIX</td>
</tr>
<tr>
<td>Genetic Algorithm</td>
<td>Mark Hughes [10]</td>
<td>C++</td>
<td>DOS</td>
</tr>
<tr>
<td>SGA-Cube</td>
<td>Jeff Earickson, Robert Smith y David Goldberg</td>
<td>C</td>
<td>NCUBE2</td>
</tr>
<tr>
<td>PGA 2.4</td>
<td>Geoffrey H. Ballinger y Peter Ross [14]</td>
<td>C</td>
<td>UNIX / DOS</td>
</tr>
</tbody>
</table>

Tabla 1. Sistemas evaluados

El diseño de DGENESIS se centró en modificar adecuadamente a GENESIS 5.0 para convertirlo en un programa distribuido que permita la experimentación con AGPs. La idea general es distribuir la población entre los nodos participantes, enviar a cada nodo los parámetros que controlan la ejecución del algoritmo, intercambiar individuos periódicamente entre los nodos y, al final, presentar un reporte con la información relevante. Para DGENESIS un nodo es un proceso UNIX que se comunica con otros nodos utilizando sockets de
Berkeley y los protocolos UDP/IP. Con esta definición de nodo es posible simular topologías grandes utilizando pocas máquinas.

A grandes rasgos, el funcionamiento del sistema se puede dividir en cuatro fases: la configuración, la comunicación inicial, la comunicación durante el experimento y la terminación.

3.1. Configuración del sistema

Antes de ejecutar los procesos del AGP es necesario que el usuario especifique la función de evaluación, los parámetros de operación del AG, el número de procesos y la manera en que se comunicarán entre sí. Esta información se almacena en archivos que serán utilizados posteriormente. Para facilitar la creación de estos archivos se modificaron algunos programas existentes en GENESIS y se crearon otros.

3.1.1. Función de evaluación

La función que el AG utilizará para evaluar a los individuos de su población debe ser proporcionada por el usuario. GENESIS tiene una manera bastante sencilla y práctica para especificar la función objetivo y se decidió incorporar esto sin modificaciones a DGÉNESIS. La función de evaluación que el usuario debe proporcionar debe estar escrita en C y tener el siguiente encabezado:

```c
double eval(str, length, vect, genes)
char str[]; /* representación en cadena */
int length; /* longitud de la cadena */
double vect[]; /* representación en punto flotante */
int genes; /* número de elementos en vect */
```

La función recibe el cromosoma como una cadena de ceros y unos (str) y/o como un arreglo de números en punto flotante (vect) que representan los valores de las variables del problema. Antes de ejecutar el sistema, el usuario establece si desea que DGÉNESIS decodifique el cromosoma como números reales antes de llamar a la función de evaluación. Si el problema no involucra variables reales el usuario es responsable de la decodificación del cromosoma y vect no contiene nada significativo.

3.1.2. Parámetros

Para especificar los parámetros que controlan la ejecución del AG el usuario debe utilizar el programa setup. Los parámetros se guardan en un archivo que será leído posteriormente por el AG y por el programa que genera los reportes. El programa que se utiliza en GENESIS para especificar los parámetros se modificó para utilizar parámetros usados en experimentos previos como valores por omisión. Así es muy sencillo cambiar directamente uno o dos parámetros del AG y volverlo a ejecutar (en GENESIS el usuario tiene que editar manualmente los archivos para modificar los parámetros). Algunos de los parámetros que se especifican con setup son:

- Número de experimentos. Un experimento es la ejecución completa de un AGP. Se pueden realizar varios experimentos utilizando la misma función de
evaluación y los mismos parámetros del AG, pero con poblaciones iniciales distintas.

- Número de evaluaciones por experimento en cada nodo. Este es el principal criterio para terminar el algoritmo.
- Tamaño y formato de las estructuras. setup pregunta si se desea que las variables codificadas en los individuos se interpreten como números reales y en caso positivo pregunta los valores extremos de cada variable y el formato que se utilizará para imprimirlo.
- Probabilidad de cruce y de mutación.
- Tamaño de cada subpoblación.
- Ventana de escalamiento. La aptitud de los individuos se escala, tomando en cuenta los valores extremos observados en las últimas n generaciones, para evitar que unos cuantos individuos dominen rápidamente a la población. El valor de n es el tamaño de la ventana de escalamiento.
- Intervalo entre respaldos. GENESIS permite crear un archivo de respaldo con la población actual y con las variables que controlan la ejecución de un experimento para poder interrumpirlo en un momento dado y continuar posteriormente. Este parámetro especifica el número de generaciones que pasan antes de crear este archivo.
- Semilla del generador de números aleatorios. Cada nodo del AGP recibe una semilla diferente para que las poblaciones iniciales sean distintas. La semilla de cada nodo es un múltiplo del número que se introduce aquí.

3.1.3. Especificación de la topología

El sistema es muy flexible en este aspecto ya que permite la conexión de los nodos participantes en formas muy diversas. La topología se especifica en un archivo llamado link.x, donde x es el nombre del experimento. El archivo tiene todas las ligas entre los procesos, así como el intervalo y la tasa de migración de cada liga según el siguiente formato:

<table>
<thead>
<tr>
<th>origen</th>
<th>destino</th>
<th>intervalo de migración</th>
<th>tasa de migración</th>
</tr>
</thead>
<tbody>
<tr>
<td>[int]</td>
<td>[int]</td>
<td>[int]</td>
<td>[float]</td>
</tr>
</tbody>
</table>

Los nodos que participan en el AGP están identificados por un número entero. Los campos origen y destino de link.x se refieren a este identificador. El intervalo de migración se refiere al número de generaciones que pasan entre cada migración y la tasa de migración es el porcentaje de la población del nodo origen que se copia y se envía al nodo destino.

El archivo donde se especifica la topología se puede editar y construir manualmente o se puede utilizar el programa links escrito para facilitar esta operación. links actualmente permite crear archivos de ligas para cuatro topologías: anillos, hipercubos, mallas y completamente conectadas, y es bastante sencillo de extender para crear archivos que describan otras topologías. El programa pregunta al usuario el número de nodos de la topología y la tasa y el intervalo de migración para todas las ligas. Si se desea cambiar los parámetros de migración de una liga particular se debe hacer manualmente editando el archivo resultante.
3.1.4. Ejecución de los procesos

Para ejecutar los procesos del AGP se escribió un programa llamado rgo. El programa usa la información contenida en el archivo usr/GHosts donde el usuario especifica los nombres de las máquinas que pueden participar en un experimento y la cantidad de procesos que soportará cada una. Se obtendrá un mejor desempeño si se colocan más procesos en las máquinas más rápidas.

rgo utiliza rsh para ejecutar los procesos y rcp para copiar los archivos de resultados, por esta razón en usr/GAHosts se debe usar el nombre de las máquinas y no su dirección IP. El usuario debe tener acceso a una cuenta en cada máquina registrada en este archivo y en cada cuenta debe otorgar permisos para ejecutar comandos remotos y copiar archivos registrando debidamente las cuentas de las demás máquinas en el archivo .rhosts. El programa rgo no sólo proporciona un medio sencillo para ejecutar los procesos del AGP sino que además juega un papel importante en la etapa de la comunicación inicial. Como se describe en la siguiente sección, los procesos deben ejecutarse en cierto orden y deben recibir algunos parámetros en la línea de comandos. rgo se encarga de que estos requisitos sean cumplidos adecuadamente y de manera transparente para el usuario del sistema.

3.2. Comunicación inicial

El propósito de la comunicación inicial es enviar a todos los procesos participantes en un experimento los parámetros necesarios para controlar el AGP y la migración. Existe un nodo (al que se llamará "principal") que se encarga de mandar esta información a los demás.

3.2.1. Registro de participantes

Una vez que el usuario ha creado los archivos necesarios para la ejecución del algoritmo, se ejecutan los procesos que intervienen en el AGP. El primer problema que se debe resolver en la etapa de comunicación inicial es la localización de los procesos que ejecutarán el AGP. Es necesario conocer las direcciones IP de las máquinas en que se ejecutarán los procesos y los puertos UDP asignados a sus sockets. El algoritmo que se utilizó para resolver este problema es el siguiente:

1. El proceso principal se ejecuta primero, encuentra la dirección IP de la máquina en que se está ejecutando y reserva diez puertos UDP. Para facilitar la implementación se reservan puertos consecutivos. El proceso principal se bloquea esperando que los demás procesos se registren.

2. Se ejecutan los demás procesos y reciben en la línea de comandos la dirección y el puerto inicial del principal. Cada proceso determina la dirección de la máquina en que se está ejecutando, reserva diez puertos consecutivos y envía un mensaje con esta información al nodo principal para registrarse. Idealmente sólo los nodos que están interesados en participar responden al nodo principal registrando su intención. El criterio que utiliza cada nodo para decidir si desea participar en un experimento puede estar basado en la carga del procesador donde se ejecuta. En la

---

1 Este número está definido como una constante en define.h y se puede cambiar fácilmente. Esta constante limita el número de ligas por las que se pueden recibir migrantes.
implementación actual todos los procesos se registran, pero en el futuro sería interesante implementar algún criterio de registro.

3. Al recibir las respuestas de los nodos que desean participar, el nodo principal registra sus direcciones y puertos. Uno de los parámetros que el usuario introduce al principio es el número de procesos que deben participar en el experimento. Si se reciben más respuestas de las requeridas, simplemente se ignoran. Si se reciben menos respuestas de las requeridas no es posible dar el servicio deseado, el programa indica un error y aborta su ejecución.

4. Se envían mensajes a los nodos seleccionados (por sus puertos iniciales) para que se preparen a recibir los parámetros del algoritmo. En este mensaje se envía el número de identificación definitivo de cada proceso, que será utilizado para crear archivos de salida únicos. Los procesos que no reciben este mensaje mueren después de un tiempo.

El nodo principal genera una estructura de datos con la topología deseada a partir de la información suministrada previamente por el usuario y de las direcciones IP y puertos UDP de los procesos registrados. Cada registro de la estructura tiene información de un proceso y tiene asociadas dos listas con las ligas por las que el nodo correspondiente debe enviar y recibir migrantes. Cada elemento de las listas tiene la frecuencia y la tasa de migración asociadas a la liga y una estructura con la dirección y el puerto del proceso con el que se hará la comunicación.

3.2.2. Envío de parámetros

Se envían a cada proceso los parámetros del AG, la plantilla que se utilizará para interpretar los cromosomas cuando estos representan variables reales y los datos de las ligas por las que enviarán y recibirán migrantes. Al recibir la información de sus ligas cada proceso construye una estructura de datos que se utilizará en la siguiente fase de ejecución del programa. Cada registro de estas estructuras representa una liga y tiene asociadas una frecuencia y una tasa de migración, la dirección del proceso con el que se realizará la migración y un campo que permite habilitar o deshabilitar la liga.

3.2.3. Población inicial

Es posible utilizar individuos generados por experimentos anteriores o por otros métodos para formar una parte de la población inicial. A primera vista puede parecer que en este caso lo más apropiado es enviar inicialmente a cada nodo un número de individuos proporcional al tamaño de su subpoblación. Esto es lógico si se toma en cuenta que en las subpoblaciones pequeñas los individuos con buenos valores de aptitud dominan más rápido a la población que en las grandes y normalmente es deseable que las poblaciones no converjan demasiado rápido para poder explorar adecuadamente el espacio de búsqueda. Sin embargo, también tiene cierto sentido enviar un número igual de individuos iniciales a cada nodo. Las poblaciones más pequeñas recibirían una proporción mayor de individuos iniciales y convergerían rápido. En un AG serial esto podría ser desastroso, pero en un AGP es posible que la convergencia prematura de algunas subpoblaciones permita alcanzar rápidamente el óptimo global. Esto es obvio en funciones unimodales, mientras que en funciones multimodales podría suceder que los óptimos locales tuvieran características comunes con el global y que al recombinarse con los individuos de otras subpoblaciones se acercaran al óptimo. DGENESIS tiene la posibilidad de usar las dos formas de
inicialización de la población inicial. Sin embargo, dado que mandar el mismo número de individuos a todos los nodos probablemente sólo funciona bien para algunas clases de problemas, por omisión el sistema envía un número proporcional de individuos iniciales a las subpoblaciones.

3.3. Comunicación durante el experimento

El propósito de esta fase es intercambiar individuos entre los procesos participantes utilizando la topología especificada por el usuario antes de la ejecución. En DGENSESIS la topología es estática, esto es, no cambia durante la ejecución del experimento. Es posible diseñar un algoritmo que cambie dinámicamente la topología según ciertos criterios como la carga de trabajo de las máquinas participantes o la falla de alguna. También tiene cierto sentido cambiar la topología si se detecta alguna condición relevante en las subpoblaciones del algoritmo como falta de diversidad. A pesar de que existen buenos argumentos en favor de tener topologías dinámicas se decidió no implementarlas en DGENSESIS debido a que el propósito del sistema es hacer experimentos con diferentes topologías para determinar el efecto que tienen las configuraciones físicas, el número de vecinos y la frecuencia e intensidad de las migraciones sobre el desempeño de los AGPs.

El ciclo principal del AG básico se modificó para incluir un punto de comunicación en el que primero se envían y después se reciben migrantes. El algoritmo modificado que se ejecuta en cada nodo se muestra en la figura 2 y se puede contrastar con el de la figura 1.

```
Generar la población inicial
Evaluar a la población
Mientras los criterios de terminación no se cumplan
{
    Seleccionar a los individuos más aptos
    Aplicarles los operadores de cruce y mutación
    Enviar migrantes si es necesario
    Recibir migrantes si es necesario
    Evaluar a los nuevos individuos
}
```

Figura 2. Ciclo principal del AG modificado

3.3.1. Envío de migrantes

Cada proceso registra en una estructura de datos la información de las ligas por las que debe enviar migrantes. Cada liga tiene distintos intervalos y tasas de migración, según las especificaciones del usuario. En cada generación se prueban los registros de esta tabla para determinar si ha llegado el momento de enviar información por alguna liga. Si el resultado es positivo se calcula el número de migrantes (tamaño de la población * tasa de migración) y se envía un mensaje con este dato al proceso correspondiente. A continuación se envían las cadenas y las evaluaciones correspondientes a los individuos que migran. En la versión actual de DGENSESIS los migrantes son los mejores individuos de la población actual y se envían copias de ellos a los procesos correspondientes. Existen otros criterios para seleccionar a los migrantes y
éstos podrían eliminarse de su subpoblación original después de ser enviados a otra. Se planea incluir estas opciones en versiones futuras de DGENESIS.

3.3.2. Recepción de migrantes

En la comunicación inicial se recibe la información de las ligas por las que se recibirán migrantes y con ella se forma la estructura de datos de ligas de recepción. Asociados a cada liga cada proceso tiene un socket y un buffer en el que se almacenan los mensajes que se reciben hasta que el proceso los lee. Cuando se reciben migrantes se crea una población extendida sobre la que se realiza el proceso de selección para que en cada generación siempre exista el mismo número de individuos. De esta manera, los migrantes son incorporados a la población si son dignos de ello de una manera muy natural. El algoritmo desconoce la procedencia de los individuos sobre los que hace la selección; pueden ser migrantes, pueden haber sido producidos por los operadores genéticos o pueden ser individuos de la generación anterior.

Por medio de setup el usuario especifica el modo, asincrono o síncrono, de recepción de migrantes. En el modo asincrono la tabla de información de las ligas por las que se reciben migrantes se ignora y en cada generación se pulean los sockets de recepción. Si existe información disponible en alguno se lee y se incorpora a la población extendida. En el modo síncrono el proceso revisa cada generación la estructura de las ligas de recepción para determinar si debe haber alguna comunicación. Si el intervalo de migración se ha cumplido puede suceder que el proceso tenga que esperar a que el otro proceso alcance la generación indicada en la estructura y mande los migrantes. De esta manera los procesos se sincronizan: los más rápidos deben esperar a que los lentos los alcancen. Los procesos lentos simplementeleen de sus sockets la información que les llegó previamente y continúan.

3.4. Terminación

Cada proceso en DGENESIS termina cuando alcanza el número de evaluaciones especificado por el usuario. Es posible que algunos procesos terminen antes que otros y esto es un problema si se está utilizando el modo síncrono de recepción de migrantes ya que cuando un proceso termina puede dejar bloqueado permanentemente a otro. Para evitar este problema se implantó una solución muy sencilla que asegura que en ningún momento existirán procesos bloqueados esperando a otro que terminó antes. Cuando un proceso termina la ejecución del experimento envía un mensaje de notificación al nodo principal y a los procesos registrados en su estructura de ligas de envío de migrantes. Entonces cuando un proceso determina que ha llegado el momento de esperar migrantes por una de sus ligas, debe revisar si en el buffer correspondiente al socket de la liga se encuentra el mensaje de terminación. Si se encuentra este mensaje la liga se deshabilita y el proceso continúa. Cuando el nodo principal detecta la terminación de todos los procesos, rco copia los archivos de salida que cada proceso generó y ejecuta el programa que genera los reportes.

DGENESIS cuenta con un programa que toma la salida producida por el AG y presenta un reporte que incluye los parámetros utilizados por el algoritmo, tablas que registran variables que indican el progreso del algoritmo y, cuando se realizan varios experimentos, el promedio y la varianza de las variables reportadas. Este programa se amplió considerablemente en DGENESIS. Se obtiene el promedio y la varianza de los resultados individuales de todos los procesos.
que intervinieron en el experimento y se cuenta con opciones que permiten dejar en archivos separados esta información. Los archivos de resultados pueden ser utilizados fácilmente por otros programas de análisis o graficación. En GENESIS el tamaño de los reportes está limitado a 200 líneas por una constante definida en el código. En el programa modificado se utiliza memoria dinámica para almacenar los datos, por lo que el límite del tamaño del reporte es la capacidad de memoria de la máquina. Adicionalmente, se puede obtener la generación y la evaluación en que se encontró al mejor individuo y su valor. Si se ejecutaron varios experimentos se presentan las estadísticas del mejor individuo de cada experimento y al final se obtiene la media y varianza de las generaciones y evaluaciones de todos los experimentos realizados.

4. CONCLUSIONES

La primera motivación para utilizar AGPs es exactamente la misma que existe para utilizar cualquier tipo de procesamiento paralelo: se desea mejorar el desempeño de los programas para poder enfrentar problemas mayores y entregar los resultados en un tiempo menor. Otra motivación es que los AGPs no sólo reducen el tiempo de procesamiento sino que en muchas ocasiones entregan resultados mejores que los AGs seriales alcanzando la meta global de la Inteligencia Artificial Distribuida: combinar sistemas individuales en sistemas distribuidos complejos con el fin de extender las capacidades de resolución de problemas más allá de la suma de las capacidades de los sistemas individuales [15].

Los AGs basan su funcionamiento en la simplificación de un proceso que ocurre en la naturaleza: la evolución a través de selección natural. Dado que la evolución es un proceso inherentemente paralelo los AGPs representan la dinámica de una población mejor que un AG serial, incluyendo aspectos como formación de especies y migración.

No es la intención de este trabajo resolver un problema específico con AGs sino presentar una herramienta que permite experimentar con AGPs para determinar cuál es la influencia de los diferentes parámetros del AGP en la calidad de la solución obtenida. Algunos resultados obtenidos con DGENESIS se reportan en [16]. DGENESIS representa un paso importante en el desarrollo de herramientas dedicadas a la experimentación con AGPs. Los resultados experimentales que se obtengan con estas herramientas deberán ayudar a comprender las características de los AGPs y a construir un marco teórico formal. Al mismo tiempo es necesario utilizar DGENESIS en problemas del mundo real para sacar provecho de las ventajas de los AGPs antes mencionadas. Actualmente se está aplicando DGENESIS a un problema de minimización de la energía potencial de moléculas de proteínas. DGENESIS y su manual de usuario [17] pueden obtenerse vía ftp anónimo en ftp.aic.nrl.navy.mil.

REFERENCIAS


