ARTÍCULO

Reconocimiento de Patrones en el Tráfico de Red Basado en Algoritmos Genéticos

Carlos Catania, Carlos García Garino

Laboratorio para la Producción Integrada por Computadora (LAPIC)
Instituto Tecnológico Universitario, Universidad Nacional de Cuyo
Lencinas s/n, Mendoza, (5500) (Argentina)
ccatania@itu.uncu.edu.ar

Facultad de ingeniería y LAPIC, ITU, Universidad Nacional de Cuyo
Centro Universitario, Mendoza, (5500) (Argentina)
cgarcia@itu.uncu.edu.ar

Resumen

El reconocimiento de patrones en el tráfico de red es uno de los componentes fundamentales de los sistemas de detección de intrusos. En este trabajo se estudian las posibilidades de aplicación de un algoritmo genético para obtener reglas que permitan reconocer las instancias de tráfico normales. El enfoque propuesto es distinto respecto a otros trabajos en donde se busca obtener los patrones de tráfico de las instancias que presentan anomalías. En el presente trabajo se discuten y proponen los ajustes necesarios a un algoritmo genético canónico, fundamentalmente en lo que se refiere a la función de fitness y a las técnicas para garantizar la convergencia hacia múltiples soluciones.

Palabras Claves: Detección de intrusos, Algoritmos Genéticos, Redes de computadoras.

1. Introducción

La seguridad de las redes de datos se ha transformado en un serio problema en los últimos años. El crecimiento vertiginoso que ha presentado la Internet ha permitido mostrar las fallas de seguridad en las implementaciones de los protocolos de red subyacentes. Situación que resulta comprensible, ya que muchos de estos protocolos originalmente fueron pensados para unir de 10 a 50 computadoras de las universidades de los Estados Unidos.

Las falencias en la seguridad de protocolos como *ARP*, *TCP*, *TELNET*, *SMTP*, *FTP* han sido la causa de ataques contra la confidencialidad, la disponibilidad y la autenticidad de los datos transportados. Si bien estos problemas han sido corregidos a lo largo de los años, continuamente se van descubriendo nuevas maneras de realizar estos ataques.

El ingeniero en seguridad de redes debe estar alerta para detectar estos ataques, informándose de las nuevas vulnerabilidades descubiertas o tipos de ataques perpetrados. Sin embargo una gran cantidad de estos ataques tienen lugar antes que se conozcan siquiera las vulnerabilidades o fallas que los provoca.

Para afrontar este problema es que en los últimos años han surgido propuestas para la aplicación de técnicas de inteligencia artificial en el ámbito de la seguridad en redes.

En este trabajo se presenta una propuesta de un algoritmo genético para el reconocimiento de patrones en el tráfico de red como punto de partida para abordar un problema de mayor envergadura

como lo es la detección de intrusos por anomalías en el tráfico de red.

El algoritmo propuesto parte de una población de individuos conformados por instancias de tráfico de red elegidas al azar, para obtener, al final del proceso, el conjunto de reglas que más coincidencias encuentre en el tráfico de la red. Toda nueva instancia de tráfico que se aparte del comportamiento normal de la red se considera una anomalía [1]. Este enfoque presenta diferencias con respecto a otros trabajos en donde se utilizan algoritmos genéticos con el objetivo de encontrar patrones sobre las instancias de tráfico anómalas.

En el presente trabajo se proponen alternativas sobre la codificación elegida para representar a la población de individuos y las funciones de optimización utilizadas. También se discuten las técnicas de nicho, como crowding y sharing utilizadas para garantizar la convergencia hacia soluciones múltiples .

El trabajo esta organizado de la siguiente manera: En la sección 2 se discuten los trabajos más relevantes relacionados con la aplicación de algoritmos genéticos al dominio de interés. Posteriormente en la sección 3 se discuten y proponen los ajustes necesarios a un algoritmo genético general, para aplicarlo en la búsqueda de patrones en el tráfico de red.

La sección 4 presenta los resultados obtenidos utilizando el algoritmo propuesto para un caso de estudio conocido [2].

2. Antecedentes

La utilización de algoritmos genéticos para la detección de intrusos ha sido estudiada con anterioridad [3, 4, 5, 6].

Gong et al. [4] y Li [5] exploran la utilización de algoritmos genéticos para la generación de reglas de clasificación en el tráfico de red, para lo cual buscan los patrones más comunes en las instancias de tráfico que presenten anomalías. Con este fin se necesita un conjunto de instancias de tráfico que previamente haya sido analizado por un experto en seguridad de redes, en el cual éste haya resaltado las instancias conteniendo anomalías. Una vez finalizado el entrenamiento, el algoritmo es capaz de generar alertas cuando observe alguno de los patrones aprendidos.

Sinclair et al. [6] propone la utilización del algoritmos genéticos para encontrar reglas sencillas que permitan encontrar patrones en el tráfico de la red combinado con arboles de decisión. Otros autores como Lu [7] y Yin [8] utilizan programación genética para obtener las reglas de clasificación. Esta variante permite obtener reglas de mayor complejidad. En otros trabajos [9, 10, 11] se estudian diferentes aplicaciones de los algoritmos genéticos en el contexto de la detección de intrusos.

Se discuten a continuación los aspectos más relevantes de las implementaciones mencionadas.

2.1. Representación de los datos

La manera de representar los individuos de la población es uno de los principales componentes de los algoritmos genéticos que necesita ser adaptado al dominio de aplicación.

Los trabajos mencionados en los antecedentes destacan los atributos más relevantes para la detección de anomalías en una conexión de red.

En Sinclair et al. [6] se consideran los atributos provenientes de la cabecera del paquete IP y del segmento TCP como: puerto origen, puerto destino, dirección IP origen y dirección IP destino. Gong et al. [4] y Li [5] proponen además, otros atributos como información sobre el tiempo de duración de la conexión y la cantidad de bytes recibidos y transferidos.

En los trabajos de Gong [4], Li [5], Lu [7] y Sinclair [6] surge la posibilidad de descartar de manera aleatoria algún gen dentro del cromosoma del individuo con el fin de obtener individuos más generales que puedan encontrar coincidencias con un mayor número de reglas. A este proceso, propuesto originalmente por De Jong [12], se lo conoce como operador de descarte de condición.

2.2. Función de optimización

Crosbie [3] utiliza una función de optimización que penaliza a los individuos en función de un ranking que indica el grado de dificultad presentado para la detección de esa instancia de tráfico. Li [5] adapta la propuesta anterior y plantea la utilización de pesos para destacar ciertos atributos de las instancias que tráfico que se consideran

más importantes en el dominio de la detección de intrusos.

En los trabajos de Gong [4] y Lu [7] se utiliza una función de fitness basada en el *support-confidence framework* [13], en donde si una regla es descripta como *si A entonces B* se puede determinar el fitness de dicha regla siguiendo la ecuación:

$$\begin{aligned} support &= |A \cup B|/N \\ confidence &= |A \cup B|/|A| \\ fitness &= w_1 * support + w_2 * confidence \end{aligned}$$

Donde al variar los valores de w_1 y w_2 se pueden detectar las anomalías de manera general o también clasificar de manera precisa los distintos tipos de anomalías.

2.3. Otros operadores

Para los operadores de mutación, cruzamiento y selección no se detallan implementaciones más allá de las propuestas de la literatura clásica de algoritmos genéticos [14].

2.4. Técnicas de nicho

Como mencionan Miller y Shaw [15], los algoritmos genéticos utilizan poblaciones que con el tiempo convergen hacia una solución optima, en general única. Si se aplica al dominio de este trabajo, se obtendría un solo individuo que tenga coincidencias con la mayor cantidad instancias de tráfico. Dado que es altamente improbable que un solo individuo pueda ser un buen representante de todas las instancias de tráfico, resulta mucho más efectivo contar con un número mayor de individuos que presenten coincidencias con algunas instancias de tráfico.

En este contexto, en la literatura [16, 15, 17] se mencionan técnicas de nicho, como crowding y sharing con el fin evitar la convergencia hacia una única solución. Las mismas se basan en el concepto de proximidad, expresado mediante una función distancia, que permite encontrar a los individuos más cercanos.

Crowding se basa en escoger unos pocos individuos al azar y reemplazar uno de los mismos por un nuevo individuo en base a criterios de proximidad [15].

Sharing en cambio degrada el fitness de un individuo en función de la cantidad de individuos que estén próximos al mismo.

En el trabajo de Li [5] se menciona la utilización de una variante de crowding aunque no se menciona cual, ni se dan detalles de su implementación. Sinclair et al. en [6] plantea la utilización de una variante de crowding utilizando distancia de Hamming para encontrar los individuos más cercanos.

Lu [7] propone una técnica basada en la competición por token. Cada instancia de tráfico del conjunto de entrenamiento contiene un token. Si un individuo coincide con la instancia de tráfico adquiere el token. La prioridad para obtener el token se basa en una probabilidad basada en la calidad de cada individuo. Finalmente la cantidad de tokens obtenidos por cada individuo se utiliza como un parámetro de la función de fitness.

3. Algoritmo genético propuesto

A partir de los antecedentes discutidos en la sección 2 y empleando como base un algoritmo genético general, se propone un algoritmo para la búsqueda de patrones que permitan reconocer las instancias normales en el tráfico de red.

Este enfoque es distinto al propuesto en los trabajos mencionados en los antecedentes (ver sección 2), en donde se busca obtener los patrones de tráfico de las instancias que presentan anomalías. Sin embargo muchas de las técnicas utilizadas en el estado del arte se pueden emplear en el contexto del presente trabajo.

En la figura del Algoritmo 1 se presenta el pseudocódigo del algoritmo genético propuesto. Se genera la población inicial P a partir de instancias de tráfico generadas al azar. Posteriormente en la línea 2 se calcula el fitness de cada uno de los individuos que componen la población. Luego se ingresa al bucle principal del algoritmo genético, en donde la condición de salida esta fijada por un número máximo de generaciones que el algoritmo debe alcanzar (línea 3).

En primer lugar se selecciona probabilística men-

te a dos individuos (p_1 y p_2) sobre los cuales se aplica el operador de cruzamiento. Estos individuos junto a los dos hijos recién obtenidos (o_1 y o_2), se evalúan mediante un operador de crowding. En el operador de crowding se aplica un criterio para determinar si alguno de los hijos recientemente obtenidos, será agregado a la población P. Este procedimiento se repite un número de veces determinado por el factor de cruzamiento $xfactor$.

El algoritmo sigue un esquema de tipo estacionario [18]. Los dos individuos (o_1 y o_2) obtenidos, pueden ser agregados de forma inmediata a la población P. De esta manera el individuo que recién ingresa a la población, puede ser seleccionado para cruzamiento sin necesidad de que el algoritmo pase a la siguiente generación.

Algoritmo 1 Esquema general del algoritmo genético propuesto

```
1:  initializarPoblacion(P)
2:  calcularFitness(P)
3:  while NroMaxGeneracionNoAlcanzado do
4:    for i = 0 to xfactor do
5:      p1 = seleccionar(P)
6:      p2 = seleccionar(P)
7:      (o1, o2) = cruzamiento(p1, p2)
8:      crowding(p1, p2, o1, o2)
9:    end for
10:   for i = 0 to dfactor do
11:     p = seleccionar(P)
12:     descarte(p)
13:   end for
14:   for i = 0 to mfactor do
15:     p = seleccionar(P)
16:     mutacion(p)
17:   end for
18:   calcFitness(P)
19: end while
20: selecionarIndividos(P)
```

Posteriormente en la línea 12, se selecciona probabilísticamente un nuevo individuo p el cual es utilizado por el operador de descarte de condición (dropping condition). Al igual que en el caso del operador de cruzamiento este proceso se repite el número de veces indicado por el factor de descarte de condición $dfactor$. Un esquema similar se sigue con el operador de mutación. Luego se recalcula el fitness de aquellos individuos de la población que hayan sido agregados o modificados por los operadores mencionados con anterioridad.

Finalmente (línea 20) se seleccionan los mejores individuos de la población P para formar el conjunto de reglas que representan las instancias normales en el tráfico de red. En las siguientes subsecciones se discuten la representación de la población, la función de optimización y las técnicas utilizadas para mantener un conjunto de soluciones posibles, las cuales conforman el algoritmo propuesto. Los operadores de selección, cruzamiento y mutación utilizados en el algoritmo propuesto no presentan aportes respecto al estado del arte.

3.1. Representación de la población

En este trabajo, en base a criterios disciplinares se seleccionan 6 atributos de una instancia de tráfico: tiempo de duración de la conexión, tipo de protocolo, puerto origen, puerto destino, dirección IP origen y dirección IP destino.

Los atributos se representan como una lista de genes, de acuerdo a la estructura indicada en la Tabla 1.

Tabla 1. Estructura del cromosoma de un individuo.

Atributo	Nro. de genes	Valor máx
HH:MM:SS	3	60
Puerto origen	1	65535
Puerto destino	1	65535
Dirección origen	4	255
Dirección destino	4	255
Protocolo	1	1024

A continuación se muestra un ejemplo de un individuo. El cual representa una conexión http (la cual se identifica con el número 5) de una duración de 2 segundos con puerto origen 2114, puerto destino 80, dirección IP destino 192.168.1.1 y dirección IP origen 192.168.1.2.

(0,0,2,5,2114,80,192,168,1,1,192.168,1,2)

Los individuos de la población admiten la posibilidad de generalizar alguno de sus genes, asignándose al mismo el valor -1. El cromosoma anterior se puede representar de la siguiente manera:

(0,0,2,5,2114,80,192,168,1,-1,192.168,1,-1)

Este individuo puede encontrar coincidencias con las instancias de tráfico dirigidas u originadas en

los nodos de la red comprendidos entre las direcciones IP 192.168.1.0 y 192.168.255.255.

La población inicial se genera a partir de instancias de tráfico del conjunto de entrenamiento y se seleccionan de forma aleatoria en base a una función de probabilidad uniforme.

3.2. Función de optimización

Se propone una función de optimización definida según la ecuación (1):

$$f(r) = \frac{\sum_{j=1}^{m} \prod_{i=1}^{n} \alpha(r_i, d_{ji})}{|D|} \quad (1)$$

Para calcular el valor de fitness un individuo r, se comparan los genes del mismo con los correspondientes d_{ji} de cada una de la instancias de tráfico perteneciente al conjunto de entrenamiento D. Para ello se emplea una función α definida como:

$$\alpha(r_j, d_{ji}) = \begin{array}{c} w_i \text{ si } r_j = d_{ji} \\ 0 \text{ si } r_j \neq d_{ji} \end{array} \quad (2)$$

Donde cada gen tiene un peso asociado w_i con el objeto de favorecer a aquellos atributos que por experiencia disciplinar resultan más relevantes. En caso de no presentar coincidencias en por lo menos un gen, la función α asigna cero. Luego, como resultado de la productoria, la función de fitness penaliza a los individuos que no presenten coincidencias en todos sus genes asignándoles también un valor cero.

Con el fin de favorecer a los genes que presentan una frecuencia de aparición relativamente alta, se introduce una variante α' de la función de peso, definida en la ecuación (3), la cual en lugar de asignar cero cuando algún gen de un individuo no presente coincidencia, le asigna un peso w'_i, que se ajusta en la práctica al 25 % del valor de w_i

$$\alpha'(r_j, d_{ji}) = \begin{array}{c} w_i \text{ si } r_j = d_{ji} \\ w'_i \text{ si } r_j \neq d_{ji} \end{array} \quad (3)$$

De esta manera la función de fitness, mediante α' permite obtener individuos que en futuras generaciones, pueden originar nuevas reglas que coincidan con un significativo número de instancias de tráfico.

3.3. Operador descarte de condición

Como se ha discutido en la sección 2.1 el operador descarte de condición se puede utilizar para obtener individuos más generales. [4, 5, 7, 6].

En este trabajo se utiliza un operador de descarte de condición que selecciona mediante una función de probabilidad uniforme algún gen del individuo. El gen seleccionado es posteriormente reemplazado por el valor -1. Como se menciona en la sección 3.1 los genes con un valor de -1 no son tenidos en cuenta durante las operaciones de comparación.

Pueden surgir individuos con demasiada generalidad, que si bien van a presentar coincidencias con muchas instancias de tráfico, proveen poca información. Para evitar la aparición de estos individuos, se utiliza un parámetro que establece un límite máximo en el número de genes de un individuo que pueden ser descartados.

3.4. Crowding determinístico

En los antecedentes si bien se menciona la utilización de técnicas para garantizar la convergencia hacia múltiples soluciones, fundamentalmente crowding, no se detallan con suficiente profundidad las posibilidades de aplicación al dominio.

En el presente trabajo se utiliza la técnica conocida como crowding determinístico propuesta originalmente por Mahfoud [16] que introduce la competencia entre padres e hijos por la misma área del espacio de soluciones posibles. Esta elección se basa en que crowding determinístico presenta buenos resultados sin aumentar la complejidad computacional del algoritmo, como se menciona en Sereni et. al [19].

Las técnicas como sharing no resultan viables en este caso, ya que el proceso de búsqueda de los individuos de la población que se encuentran en el mismo nicho, resulta demasiado costoso en términos de tiempo computacional. A lo que se agrega la necesidad de recalcular el fitness de todos los individuos en cada generación, lo que eleva en varios ordenes de magnitud el tiempo total de ejecución.

La implementación de crowding determinístico utilizada en este trabajo es descripta por Leon et. al. [20]. Realizada la operación de cruzamiento, cada hijo reemplaza a alguno de sus padres más próximo, si y sólo si, posee un valor de fitness más alto.

Como se menciona en los antecedentes la noción de proximidad está dada por una función de distancia. Mahfoud [16] observa que dicha función debe poseer la mayor cantidad de información posible sobre el dominio de aplicación del problema. Luego se propone la utilización de una función de distancia euclidiana definida en la ecuación (4), que al igual que la función de optimización favorece a través de pesos a ciertos atributos seleccionados en base a la experiencia disciplinaria.

$$\begin{aligned} Distancia &= \sqrt{Distancia'} \qquad (4)\\ Distancia' &= (p_i - p_j)^2 w_1 + (s_i - s_j)^2 w_2 \\ &\quad + (s'_i - s'_j)^2 w_3 + (d_i - d_j)^2 w_4 \\ &\quad + (d'_i - d'_j)^2 w_5 \end{aligned}$$

donde p significa el puerto origen, mientras que s, s' y d, d' son los 2 bytes más significativos de las direcciones IP de origen y destino respectivamente. Los w_k indican los pesos asignados a los términos de la ecuación (4)

3.5. Obtención de los mejores individuos

Debido a la aplicación de técnicas de nicho, existen dentro de la población grupos o clústeres de individuos que presentan soluciones muy similares. Con el objeto de obtener los mejores representantes de cada uno de estos grupos, se utiliza una heurística cuyo pseudocódigo es presentando en el algoritmo 2. El individuo p con mayor fitness se extrae de la población P. Se verifica que no exista en el conjunto de reglas R un individuo r con genes de idéntico valor a p. Durante la comparación solamente se utilizan los genes que no hayan sido modificados por el operador de descarte de condición. Si ambos individuos presentan coincidencias en sus genes, se prefiere a aquel que presente el mayor número de genes generalizados y se procede a descartar al otro. Este procedimiento se repite hasta que el conjunto de reglas R esté integrado por un número N de individuos.

Algoritmo 2 Heurística para la selección de mejores individuos

```
P=Población
N=Número máximo de reglas
R=Conjunto de reglas
for p ∈ P hasta N do
  if p ≠ algún r ∈ R then
    R = R ∪ p
  else
    R = R∪ IndividuoMasGeneralizado(r,p)
  end if
end for
```

4. Caso de estudio: Aplicación al conjunto de instancias de tráfico de DARPA

El objetivo de los experimentos es determinar la capacidad del algoritmo propuesto para obtener una población de individuos que presenten coincidencias con el mayor número de instancias de tráfico. Además de comprobar la influencia de la función de fitness utilizada y las funciones de distancias utilizadas en la técnica de crowding.

Para realizar los experimentos se utiliza el conjunto de datos provisto por DARPA [2] el cual ha sido muy utilizado en trabajos dentro del área [4, 20, 5, 7, 11].

El algoritmo itera a lo largo de 1200 generaciones, sobre 2 conjuntos conformados con instancias de tráfico tomadas de los datos provistos por DARPA. El primer conjunto se utiliza para la etapa de entrenamiento y contiene 9000 entradas que representan 4 horas de tráfico. El segundo conjunto se emplea para la fase de prueba y contiene 35000 instancias de tráfico que representan 24 horas. Se realizan 40 ejecuciones del algoritmo con el objetivo de validar su comportamiento.

Al finalizar el entrenamiento, los individuos obtenidos se utilizan para buscar coincidencias en el conjunto de prueba y se presenta el porcentaje de errores cometidos. Se considera que un individuo presenta coincidencias con una instancia de tráfico si y sólo si los atributos no generalizados del mismo coinciden con los atributos correspondientes de la instancia de tráfico.

4.1. Estudio de la influencia de la función distancia en crowding

En esta subsección se estudia la influencia de las funciones distancia mencionadas en la sección 3.2, para los operadores crowding y crowding determinístico. Los resultados, para los conjuntos de prueba y entrenamiento, se presentan mediante histogramas de frecuencias relativas en función del error porcentual.

Una comparación de los resultados obtenidos cuando se aplica la función distancia Hamming y la función distancia euclidiana por pesos, para el operador de crowding clásico, se muestran en la figura 1 y la figura 2, respectivamente.

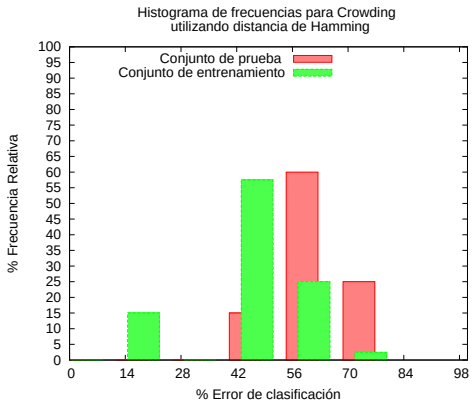

Figura 1. Histograma de frecuencias relativas para crowding utilizando distancia de Hamming

La figura 1 muestra el histograma con los resultados en los conjuntos de entrenamiento y prueba cuando se aplica la función distancia de Hamming. El eje de abscisas corresponde al porcentaje del error de clasificación, mientras que el eje de ordenadas al origen corresponde a la frecuencia relativa.

Para el conjunto de entrenamiento en la figura 1 se obtiene el 42 % y el 56 % del error de clasificación para el 55 % y el 25 % de las pruebas de validación procesadas respectivamente. En el caso del conjunto de prueba se obtiene el 56 % y el 70 % de error de clasificación para el 60 % y el 25 % de las pruebas de validación.

En la figura 2 se presentan los resultados al utilizar la función de distancia euclidiana con pesos. Se observa que el valor más alto en el histograma para el conjunto de entrenamiento ahora ha descendido a valores entre el 14 % y el 42 % de error de clasificación. En el conjunto de prueba se alcanzan valores entre el 42 % y el 56 %.

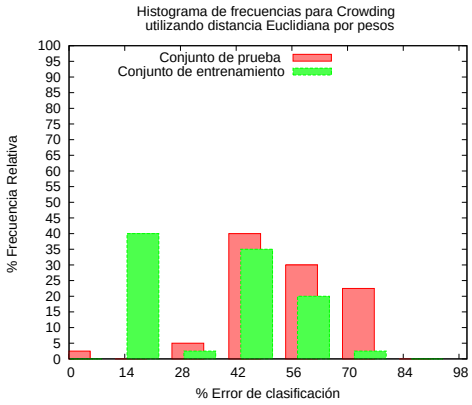

Figura 2. Histograma de frecuencias relativas para crowding utilizando distancia euclidiana

Como surge de la comparación de las figuras 1 y 2 la distancia euclidiana por pesos muestra mejores resultados que la distancia de Hamming, tanto para el conjunto de prueba como para el conjunto de entrenamiento. Sin embargo, como se observa en la figura 2, aún en este caso se obtienen errores de clasificación altos (42 % y 56 %) para un número importante de casos.

Las figuras 3 y 4 muestran los resultados obtenidos para el caso de crowding determinístico cuando se aplica distancia de Hamming y euclidiana por pesos, respectivamente. De las mismas surge que los resultados mejoran considerablemente a los obtenidos con crowding.

En la figura 3 se muestran los resultados al aplicar distancia de Hamming. Se observa que alrededor del 80 % de las validaciones ejecutadas han obtenido un error de clasificación para el conjunto de entrenamiento entre el 14 % y el 28 %. Para el conjunto de prueba los valores más altos del histograma se encuentran entre el 14 % y 28 % del error de clasificación para el 25 % y el 40 % de las validaciones realizadas, respectivamente.

En la figura 4 se muestran los resultados al aplicar la función de distancia euclidiana. Puede verse que para el conjunto de entrenamiento alrededor del 60 % de las 40 pruebas de validación ejecutadas finalizan con un error de clasificación cercano al 15 %. Para el conjunto de prueba el histograma esta más distribuido ya que se obtiene aproximadamente un 14 % de error para el 25 % de las validaciones, un 28 % de error para el 40 %

de las validaciones y un 42 % para el 25 % de las validaciones.

En este último caso los porcentajes en la frecuencia relativa resultan similares a los de la Figura 3 tanto para el conjunto de entrenamiento como para el conjunto de prueba. Luego se desprende que la utilización de la propuesta de función de distancia euclidiana basada en pesos no ofrece ventajas significativas sobre la función de distancia de Hamming al utilizar crowding determinístico.

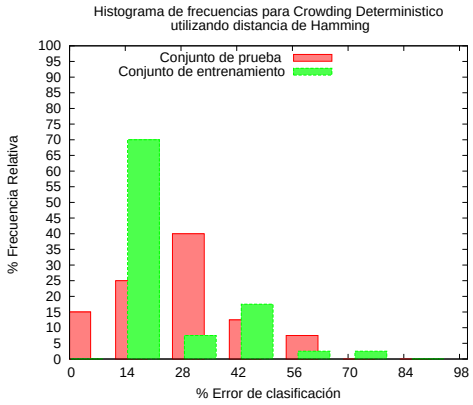

Figura 3. Histograma de frecuencias relativas para crowding determinístico utilizando distancia de Hamming

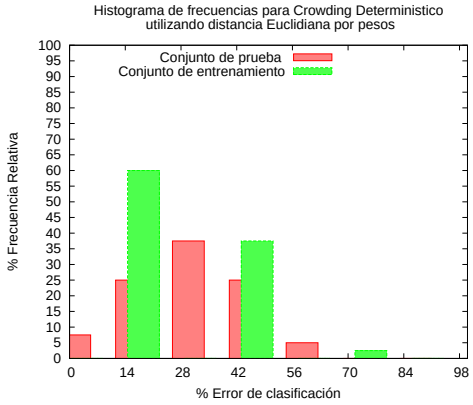

Figura 4. Histograma de frecuencias relativas para crowding determinístico utilizando distancia euclidiana

4.2. Estudio de la influencia de la función de peso en la función de optimización

En esta subsección se comparan las distintas variantes para la función de peso α y α' desarrolladas en las sección 3.2.

Las reglas obtenidas por el algoritmo genético al aplicar la función de peso α en la función de optimización se muestran en Tabla 2. Las tres primeras celdas indican el número de regla y las instancias de tráfico que fueron exitosamente reconocidas en el conjunto de entrenamiento (IRE) y conjunto de prueba (IRP). Se observa en este caso que el algoritmo ha convergido a una única solución. Esto se debe a que una gran cantidad de individuos no han sido capaces de encontrar coincidencias en por lo menos una instancia de tráfico del conjunto de entrenamiento y en consecuencia fueron penalizados y, eventualmente, descartados.

Los individuos obtenidos al finalizar la ejecución del algoritmo utilizando la función de peso α' se muestran en la Tabla 3, que en este caso conforman 16 reglas. Se observa que muchas de las reglas no han sido capaces de encontrar coincidencias en ninguna instancia de tráfico del conjunto de entrenamiento y el conjunto de prueba. Este resultado responde por un lado a las características de la función de peso α' que ha favorecido a individuos con atributos con alta frecuencia de aparición y por otro a la heurística detallada en la sección 3.5 que selecciona a los individuos que presentan mayores diferencias entre si. De esta manera se ha podido encontrar reglas que podrían ser capaces de clasificar instancias de tráfico no presentes en el conjunto de entrenamiento.

Las figuras 5 y 6 muestran la evolución del error del mejor resultado obtenido, para las 40 validaciones efectuadas, utilizando las funciones α y α', respectivamente.

Como se observa en la Figura 5 el algoritmo que utiliza la función α converge a una solución a partir de la generación 800 y obtiene resultados apenas inferiores al 60 % de error de clasificación en el conjunto de entrenamiento y muy cercanos al 80 % en el conjunto de prueba.

La Figura 6 muestra los mejores resultados obtenidos con el algoritmo cuando se emplea la función α' definida en la ecuación 3. En este caso la convergencia del algoritmo se observa a partir de la generación 800 con una solución cercana al 15 % de error de clasificación. Mientras que en el conjunto de prueba recién a partir de la generación 900 se converge hacia una solución cercana al 20 % de error de clasificación.

Tabla 2. Reglas obtenidas utilizando α

	IRP	IRE	HH	MM	SS	Proto	sPort	sPort	sIP	sIP	sIP	sIP	dIP	dIP	dIP	dIP
1	7620	4120	0	0	1	5	-1	80	172	16	-1	-1	207	-1	-1	-1

Tabla 3. Reglas obtenidas utilizando α'

	IRP	IRE	HH	MM	SS	Proto	sPort	dPort	sIP	sIP	sIP	sIP	dIP	dIP	dIP	dIP
1	7184	2912	0	0	1	5	-1	80	172	16	117	-1	-1	-1	-1	-1
2	5086	2669	0	0	1	5	-1	80	172	16	116	-1	-1	-1	-1	-1
3	5592	2658	0	0	1	5	-1	80	172	16	115	-1	-1	-1	-1	-1
4	5004	0	0	0	1	5	-1	80	172	16	112	-1	-1	-1	-1	-1
5	3491	0	0	0	1	5	-1	80	172	16	113	-1	-1	-1	-1	-1
6	0	0	0	0	1	5	-1	80	172	16	60	-1	-1	-1	-1	-1
7	0	0	0	0	1	5	-1	80	172	16	118	-1	-1	-1	-1	-1
8	0	0	0	0	1	5	-1	80	172	16	101	-1	-1	-1	-1	-1
9	0	0	0	0	1	1	-1	80	172	16	116	-1	-1	-1	-1	-1
10	0	0	0	0	1	1	-1	80	172	16	115	-1	-1	-1	-1	-1
11	0	0	0	0	1	7	-1	80	172	16	117	-1	-1	-1	-1	-1
12	0	0	0	0	1	5	-1	84	172	16	117	-1	-1	-1	-1	-1
13	0	0	0	0	1	5	-1	16	172	16	117	-1	-1	-1	-1	-1
14	0	0	0	0	1	4	-1	80	172	16	116	-1	-1	-1	-1	-1
15	0	0	0	0	1	7	-1	80	172	16	116	-1	-1	-1	-1	-1
16	0	0	0	0	1	5	-1	88	172	16	116	-1	-1	-1	-1	-1

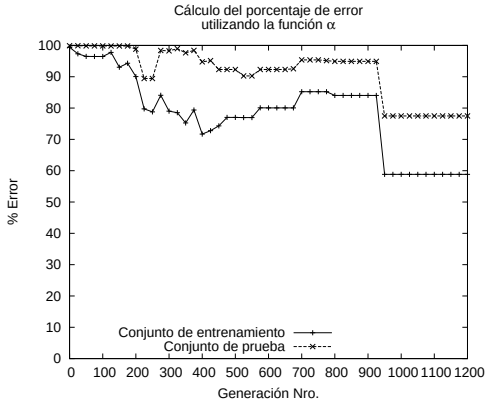

Figura 5. Porcentaje de error en los conjuntos de prueba y entrenamiento utilizando la función α definida en la ecuación (2)

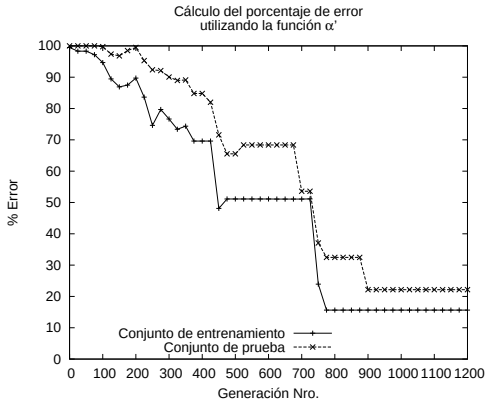

Figura 6. Porcentaje de error en los conjuntos de prueba y entrenamiento utilizando la función α' definida en la ecuación (3)

5. Conclusiones y trabajos futuros

En el presente trabajo se han clasificado exitosamente patrones de tráfico normal mediante el algoritmo genético propuesto.

El algoritmo genético para el reconocimiento de patrones de tráfico normal observa resultados prometedores en el caso de estudio planteado. Se generan reglas con atributos que permiten encontrar coincidencias del orden del 85 % para el conjunto de entrenamiento y alrededor de un 80 % para el conjunto de prueba.

La utilización de la función de peso α' definida en la ecuación (3) constituye una mejora significativa a la función de fitness. Como se ha señalado en la sección 4.2 se han podido encontrar reglas de clasificación que, potencialmente son capaces de clasificar instancias de tráfico no presentes en el conjunto de entrenamiento.

De las técnicas de crowding evaluadas, crowding determinístico ha presentado los resultados más prometedores a costa de no aumentar la complejidad computacional del algoritmo. Crowding determinístico muestra no ser sensible a la función de distancia utilizada, situación que no se presenta en otras variantes de crowding.

Se observa que las funciones de fitness descriptas en el trabajo son costosas en términos de recursos computacionales. Luego surge la posibilidad de investigar el comportamiento del algoritmo propuesto en ambientes distribuidos. Otra posibili-

dad es la de estudiar la aplicación de algoritmos genéticos distribuidos [21, 22]. Estos algoritmos permiten acelerar la convergencia de las soluciones a la vez que exploran distintos espacios de soluciones.

En trabajos futuros se comprobará el funcionamiento de las reglas obtenidas en la detección de anomalías aplicadas a situaciones de tráfico real.

Referencias

[1] B. Mukherjee, L. T. Heberline, and K. Levitt, "Network instruction detection.," *IEEE Network*, vol. 8, pp. 26–41, 1994.

[2] R. Lippmann, J. W. Fried, D. J. Korba, and K. Das, "The 1999 darpa off-line intrusion detection evaluation," *Computer Networks*, vol. 34, pp. 579–595, 2000.

[3] M. Crosbie and G. Spafford., "Applying genetic programming to intrusion detection.," in *AAAI Fall Symposium on Genetic Programming*, 1995.

[4] R. Gong, M. Zulkernine, and P. Abolmaesmumi, "A software implementation of a genetic algorithm based approach to network intrusion detection.," in *Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNDP/SWAN'05)*, vol. 0, pp. 246–253, 2005.

[5] W. Li, "A genetic approach to network intrusion detection," tech. rep., SANS Institute, 2003.

[6] C. Sinclair, P. Lyn, and S. Matzer, "An application of machine learning to network intrusion detection.," in *15th Annual Computer Security Applications Conference*, 1999.

[7] W. Lu and I. Traore., "Detecting new forms of network intrusion using genetic programming.," *Computational Intelligence,*, vol. 20, pp. 475–494, 2004.

[8] C. Yin, S. Tian, H. Huang, and J. He, "Applying genetic programming to evolve learned rules for network anomaly detection," in *LNCS ICNC 2005*, vol. 3612, pp. 323–331, 2005.

[9] S. Bridges and R. Vaughn, "Fuzzy data mining and genetic algorithms applied to intrusion deteccion," in *National Information Systems Security Conference*, 2000.

[10] J. Gomez, D. Dasgupta, F. Gonzalez, and O.Ñasraoui, "Complete expression trees for evolving fuzzy classifier systems with genetic algorithms and application to network intrusion detection.," in *Annual Fuzzy Information Society*, pp. 469–474, 2002.

[11] T. Shon, Y. Kim, C. Lee, and J. Moon, "A machine learning framework for network anomaly detection using svm and ga," in *6th IEEE Information Assurance Workshop*, 2005.

[12] K. A. D. Jong, W. M. Spears, and D. F. Gordon, "Using genetic algorithms for concept learning," *Machine Learning*, vol. 13, pp. 161–188, 1993.

[13] M. Wong and K. Leung, *Data Mining Using Grammar-Based Genetic Programming and Applications.* Kluwer Academic Publishers Norwell, 2000.

[14] D. Goldberg, *Genetic Algorithms in search Optimization and Machine Learning.* Addison Wesley, 1989.

[15] B. Miller and M. Shaw, "Genetic algorithms with dynamic niche sharing for multimodal-function optimization.," Tech. Rep. 95010, University of Illinois at Urbana-Champaing., 1995.

[16] S. Mahfoud, "Crowding and preselection revisited.," Tech. Rep. 92004, University of Illinois at Urbana-Champaing., 1992.

[17] V. R. Vemuri and W. Cedeno, *Practical Handbook of Genetic Algorithms*, vol. 2, ch. Multi-Niche Crowding for Multi-Modal Search. CRC Press, 1995.

[18] D. Whitley, "The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best," in *Proceedings of the Third International Conference on Genetic Algorithms* (J. D. Schaffer, ed.), (San Mateo, CA), Morgan Kaufman, 1989.

[19] B. Sareni and L. Krahenbuhl., "Fitness sharing and niching methods revisited.," *IEEE Transactions on Evolutionary computation*, vol. 2, September 1998.

[20] E. Leon, O.Ñasraoui, and J. Gomez, "Anomaly detection based on unsupervised niche clustering with application to network intrusion detection.," in *IEEE Congress on Evolutionary Computation*, 2004.

[21] E. Alba and J. Troya, "A survey of parallel distributed genetic algorithms.," *Complexity Volume*, vol. 4, no. 4, 1999.

[22] E. C. Paz, "A summary of research on parallel genetic algorithms.," Tech. Rep. 95007, University of Illinois at Urbana-Champaing., 1995.