Algoritmos genéticos para la búsqueda de políticas óptimas en procesos de decisión de Markov

Genetics Algorithms for Searching Optimal Policies on Markov Decision Processes

ROBERTO EMILIO SALAS RUIZ
Ingeniero de Sistemas de la Universidad del Norte, Magíster en Ingeniería de Sistemas de la Universidad Nacional de Colombia. Profesor Universidad Distrital Francisco José de Caldas de la Facultad Tecnológica.
Correo electrónico: resalasr@udistrital.edu.co

Clasificación del artículo: investigación

Fecha de recepción: 13 de abril de 2007

Fecha de aceptación: 19 de julio de 2007

Palabras clave: proceso de decisión de Markov, política, algoritmo genético, acción, estado.

Key words: decision processes (PDM), policy, genetic algorithm (AG), action, state.

RESUMEN

Este artículo muestra la implementación de un algoritmo genético (AG) para resolver problemas de decisión de Markov (PDM). Presenta los conceptos básicos de los PDM y del AG, también muestra una descripción del modelo propuesto, la forma en la que fue formulado y los resultados obtenidos aplicados a dos ejemplos.

ABSTRACT

This paper shows the implementation of a Genetics Algorithm - AG *(from "Algorítmo Genético" in Spanish)* to solve Markov Decision Processes – PDM *(from "Problemas de Decisión de Markov" in Spanish)*. It presents basic concepts of both, PDM and AG; it also shows a description of the proposed model, the way it was formulated and the results obtained in two examples.

* * *

1. Introducción

Este artículo se centra en un modelo para la toma de decisiones secuenciales bajo incertidumbre, los llamados procesos de decisión de Markov (PDM). Este modelo consta de un conjunto de estados, un conjunto de decisiones disponibles (acciones) para cada estado, un costo o recompensa, de acuerdo con la acción que se tome en cada estado. Dependiendo del estado en que se encuentre el sistema y de la acción que se tome en ese estado se transitará con cierta probabilidad a otro estado; esta probabilidad es independiente de los estados y de las acciones pasadas en el sistema. La idea del tomador de decisiones es encontrar una secuencia de acciones (política) que se van a seguir en cada estado, de tal manera que el sistema, después de un número determinado de iteraciones o en ciertos casos a largo plazo, produzca una recompensa o un costo óptimos.

En las últimas décadas ha habido un notable resurgimiento en la investigación teórica y aplicada de estos modelos. Estos modelos surgieron como una ramificación de la investigación de operaciones en la década del cincuenta, y en años posteriores han ganado reconocimiento en diversos campos como la ecología, la economía y la ingeniería [1]. La tarea fundamental en el estudio de estos modelos es el diseño de algoritmos eficientes para encontrar políticas óptimas.

El objetivo principal de este artículo es el uso de un algoritmo genético simple como una alternativa para encontrar buenas políticas ("cercanas a las óptimas") en este tipo de modelo, el cual se aplicó a un problema clásico.

La teoría, la metodología y los resultados de este trabajo se consignan en el siguiente orden. En el segundo capítulo se revisa, brevemente los conceptos básicos de procesos de decisión de Markov y su solución, incluyendo trabajos relacionados en el área; igualmente, en este apartado se incluye la teoría básica de algoritmos genéticos. Los apartados 3, 4 y 5 desarrollan el algoritmo genético propuesto para la búsqueda de políticas óptimas en procesos de decisión de Markov y su implementación en el problema del reemplazo del automóvil y un modelo de apuesta. Por último, se enuncian las conclusiones.

2. Generalidades

2.1. Procesos de decisión de Markov

Un proceso de decisión de Markov es similar a una cadena de Markov, con la diferencia que la matriz de transición depende de la acción tomada por un agente en cada paso del tiempo. El objetivo es hallar una función llamada política, la cual especifica qué acción se va a tomar en cada estado, de modo que optimice alguna función de costo o de recompensa. Un proceso de decisión de Markov, está definido por:

- Un conjunto de posibles estados S.
- Un conjunto de posibles acciones A.
- Una función real de costo o recompensa $R(s,a)$, en la cual $s \in S$ y $a \in A$.
- Una descripción T de los efectos de cada acción sobre cada estado.

El proceso de decisión de Markov cumple la propiedad de Markov: "Los efectos de una acción tomada en un estado, dependen sólo del estado actual y no de la historia anterior". Las acciones que se toman pueden ser de dos clases:

- Acciones deterministicas: $T: SxA \rightarrow S$. Para cada estado y acción, especificamos un nuevo estado.
- Acciones probabilísticas: $T : SxA \rightarrow Prob(S)$. Para cada estado y acción, especificamos una distribución de probabilidad sobre los próximos estados. La distribución se representa por $P(S'|s,a)$.

2.2. Políticas

Una política es una regla que especifica qué acción tomar en cada punto en el tiempo. En general, las decisiones especificadas por una política pueden:

- Depender del estado actual del proceso que describe el sistema.

- Ser aleatoria, es decir, dependen, de algún evento externo aleatorio.
- Depender de estados pasados o decisiones.

Una política estacionaria es definida por una función de acción que asigna una función a cada estado, independiente de previos estados, previas acciones y tiempo. Bajo una política estacionaria, un proceso de decisión de Markov es una cadena de Markov [2].

2.3. Obtención de la política óptima en procesos de decisión de Markov

Para hallar la política óptima de un proceso de decisión de Markov, existen varios métodos para obtenerla, entre los cuales podemos mencionar: solución por enumeración exhaustiva, solución por programación lineal, solución por el algoritmo de mejoramiento de políticas y por otras técnicas. De las dos primeras podemos encontrar amplia información en [3] y sobre el algoritmo de mejoramiento de políticas en [4].

Solución por otras técnicas

Como antecedente de trabajos en este campo de búsquedas de políticas óptimas en PDM, por métodos alternativos, está el de Hansen et al (1999) [5], en el cual proponen el algoritmo LAO* que es una derivación del algoritmo clásico de búsqueda heurística AO*, en el cual se formaliza el MDP como un problema de búsqueda en un grafo, en el cual cada nodo del grafo corresponde a un estado del problema y cada arco es una acción que causa una transición de un estado a otro y sobre este grafo se aplica todo el proceso de búsqueda; la clave está en que no se hace una búsqueda exhaustiva, lo cual hace que sea una búsqueda eficiente. Igualmente, en este trabajo se generalizan algunos análisis teóricos de búsquedas en problemas simples de PDM. Además, está el trabajo hecho por el autor del presente artículo [6], en que se desarrolló un artículo llamado *Simulated Annealing para resolver problemas de procesos de decisión de Markov*, en la cual se probó que se puede enfrentar con éxito este tipo de problemas con técnicas alternativas, tal y como fue el caso con el mencionado algoritmo.

Sobre la utilización de algoritmos genéticos para la obtención de políticas óptimas en PDM está el trabajo de Danny Barash en la Universidad de California (1999) [7] en el cual presenta una nueva metodología para resolver PDM basado en algoritmos genéticos. En este trabajo resolvió un problema sencillo de PDM por el método clásico de iteración de políticas y por el método propuesto del algoritmo genético y como conclusión obtuvo que el algoritmo de iteración de políticas funcionó mejor que el algoritmo genético, con lo cual se puede deducir que en casos en que se puedan resolver utilizando el algoritmo de iteración de políticas es preferible que se utilice este método, ya que implica un trabajo menor y se obtiene el mejor resultado. No obstante, en problemas cuyo espacio de búsqueda es muy grande y en los cuales los métodos de programación dinámica no son muy precisos, los algoritmos genéticos podrían dar buenas aproximaciones.

2.4. Algoritmos genéticos

Los algoritmos genéticos son técnicas de búsqueda estocástica basadas en los mecanismos de selección natural y en la genética. Los algoritmos genéticos inician con un conjunto inicial de soluciones aleatorias denominadas población. La información de cada individuo en la población se llama cromosoma y representan una solución al problema que se tiene a la mano. Un cromosoma es una cadena de símbolos que, usualmente, pero no necesariamente, es una cadena binaria. Los individuos evolucionan a través de sucesivas iteraciones, llamadas generaciones. Durante cada generación, los individuos se evalúan, usando alguna medida de adaptación. A la próxima generación de nuevos individuos se les llama descendencia y son formados de la siguiente manera:

- Seleccionando y reproduciendo copias de los individuos, según sus valores de adaptación.
- Reemplazando o generando una población de igual tamaño, algunos de los individuos pueden permanecer entre generaciones.
- Combinando los cromosomas de los individuos de la actual generación utilizando un operador de cruce.

- Modificando el cromosoma de un individuo utilizando un operador de mutación.

Los individuos mejor adaptados tienen más alta probabilidad de seguir siendo seleccionados. Después de varias generaciones, los algoritmos convergen a una población de individuos, los cuales se espera que representen cercana a la solución óptima o subóptima del problema. Sea *P(t)* la actual población en la generación *t*; la estructura general de los algoritmos genéticos es descrita en el cuadro 1(Procedimiento: Algoritmos Genéticos).

$t \leftarrow 0$;
inicializar $P(t)$;
evaluar $P(t)$;
mientras (no condición de terminación) **hacer**
seleccionar $P'(t+1)$ de $P(t)$
cruzar y mutar $P'(t+1)$ para producir $P(t+1)$
evaluar $P(t+1)$
$t \leftarrow t+1$
fin_mientras
fin_procedimiento

Cuadro 1. Procedimiento: Algoritmos Genéticos

Usualmente, la inicialización es aleatoria. En consecuencia, para usar un algoritmo genético se requiere determinar:

- La función de evaluación de los individuos.
- La representación del cromosoma.
- El operador de selección.
- Los operadores de cruce y mutación.
- La creación de la población inicial.
- Criterio de terminación.

Gran parte del éxito o del fracaso en la aplicación del algoritmo genético para resolver problemas de optimización está en la adecuada escogencia y selección de los seis anteriores aspectos. En general, los algoritmos genéticos han sido empleados en una amplia variedad de aplicaciones prácticas por dos razones principales: la simplicidad de su aplicación y la ausencia de procedimientos más eficaces para resolver esos problemas [8].

3. Algoritmo genético para resolver problemas de decisión de Markov

A continuación se presenta la estructura propuesta de un algoritmo genético para la búsqueda de políticas óptimas en procesos de decisión de Markov.

3.1. Algoritmo propuesto

La idea del algoritmo propuesto para los problemas de decisión de Markov es encontrar buenas políticas con un algoritmo genético. Para ello se propone usar un algoritmo genético que evolucione políticas de decisión asociadas a un problema dado. El algoritmo genético no garantiza encontrar una política óptima, pero sí una buena política. Para la implementación del mismo se utilizó la herramienta GAOT (Genetic Algorithms Optimization Toolbox for MATLAB) desarrollada en North Carolina State University (NCSU) [9], la cual utiliza el procedimiento descrito en el cuadro 1.

3.2. Representación de los individuos

Los individuos representan políticas en el proceso de decisión de Markov y se representan mediante un vector, en el que cada posición del mismo indica qué decisión se toma en cada estado. Por ejemplo, se debe suponer que un problema de decisión de Markov consta de tres posibles estados {1, 2, 3} y que en cada estado se pueden tomar cuatro decisiones {1, 2, 3, 4}, un posible individuo para este caso sería (1, 1, 3) o (2, 4, 1), etc. Para el problema resuelto en el presente trabajo, los individuos se representan como un vector de enteros.

3.3. Operadores genéticos

La forma más simple de un algoritmo genético involucra un operador de selección y los operadores genéticos de cruce y mutación. Para el caso del algoritmo propuesto para los procesos de decisión de Markov, se utilizaron los operadores de selección que se incluían en la herramienta GAOT, adicionalmente, se implementó el siguiente criterio elitista, en cada generación de individuos, los cinco mejores eran almacenados antes de evolucionar la población y luego de evolucionada, estos cinco mejores reemplazaban a los cinco peores de la nueva generación producida, lo cual nos garantiza que no se pierdan los mejores individuos de cada generación.

Utilizando la flexibilidad del GAOT se programaron varios operadores de mutación que aprovechaban características especiales del problema, en particular se introdujo una forma de mutación llamada mutación iterativa, la cual muta un individuo (política) mediante una iteración de la rutina de mejoramiento de políticas. Estos operadores dan una mayor eficacia en la búsqueda en aquellos casos en los que los operadores básicos del algoritmo genético no producían una búsqueda exitosa.

3.4. Representación de los individuos

La inicialización fue básicamente una generación aleatoria de individuos, que representan cada una de las políticas, las cuales luego fueron evolucionando. Esta inicialización fue puramente aleatoria y no se utilizó ningún criterio adicional para la generación de los mismos, eso sí, teniendo en cuenta que los individuos representaran políticas válidas. Para la terminación, se utilizó el criterio del número de generaciones, es decir, el algoritmo genético evolucionaba un determinado número de generaciones y se toma como solución el mejor individuo de esa última generación. La función de adaptación utilizada dependió del problema en particular que se iba a solucionar, por lo tanto, la misma se presentará en la solución del problema propuesto.

4. Implementación del algoritmo genético en el problema del reemplazo del automóvil

4.1. Problema del reemplazo del automóvil [2]

Se consideró el problema de reemplazo de un automóvil sobre un intervalo de tiempo de diez años. Se está de acuerdo en revisar su actual situación cada tres meses y tomar una decisión de mantener el actual carro o negociarlo por uno nuevo en ese momento. El estado del sistema i es descrito por la edad del carro en periodos de tres meses; i puede ir desde 0 hasta 40. A fin de mantener el número de estados finitos, un carro de edad 40 es desechado (se considera que ya está gastado). En cada estado se puede comprar un carro de cualquier edad entre 0 y 39, sea a el índice de la acción que representa: comprar un carro de edad a - 2 para a= 2, 3, 4,..., 41. La acción con índice a=1 es mantener el carro actual. Así, se tienen 41 estados en los que se pueden tomar 41 decisiones posibles, de modo que hay 41^{41} posibles políticas.

Los datos suministrados son los siguientes:

C_i, es el costo de comprar un carro de edad i.

T_i, es el valor de negociar un carro de edad i, este costo se le deduce al vendedor de un carro de edad i.

E_i, es el costo de operar un carro de edad i hasta que alcance la edad i+1.

P_i, es la probabilidad de que un carro de edad i sobreviva a la edad i+1, sin incurrir en un costo prohibitivo de reparación. Esta probabilidad es necesaria para limitar el número de estados a los que puede transitar el sistema.

Suponemos que un carro de cualquier edad que tiene una avería irreparable es inmediatamente enviado al estado 40. Naturalmente, P_{40}=0, porque se considera imposible en el modelo, para un carro, alcanzar la edad 41. Los valores específicos de los parámetros C_i, T_i, E_i y P_i, se encuentran en [2].

Igualmente tenemos,

$$q_i^a = -E_i, \text{ para } a=1, \quad (2)$$

$$p_{ij}^a = \begin{cases} p_i & si \quad j = i+1 \\ 1-p_i & si \quad j = 40 \\ 0 & para \quad otra \quad j \end{cases} \text{para } a=1 \quad (3)$$

$$q_i^a = T_i - C_{a-2} - E_{a-2}, \text{ para } a > 1, \quad (4)$$

$$p_{ij}^a = \begin{cases} p_{a-2} & si \quad j = a-1 \\ 1-p_{a-2} & si \quad j = 40 \\ 0 & para \quad otra \quad j \end{cases} \text{para } a > 1. \quad (5)$$

En la cual q_i^aes el costo del estado i bajo la acción a. p_{ij}^a es la probabilidad de transición del estado i al estado j por la acción a.

4.2. Solución por la rutina de mejoramiento de políticas

La solución encontrada por el algoritmo de mejoramiento de políticas es:

$$f(D) = -150.945836$$

mejor política D=(14, 14, 14, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14)

La mejor política es: si se tiene un carro de edad 0, 1 ó 2 o de edad mayor de 25, lo mejor es negociarlo por un automóvil de edad 12. Si la edad del carro se encuentra entre 3 y 25 lo mejor es mantenerlo y no negociarlo por otro. Igualmente, se observó que independiente de la política en la que se empiece a iterar, este algoritmo siempre llega a la misma solución.

4.3. Solución por algoritmos genéticos

4.3.1. Representación de los individuos

Como la idea de aplicar el algoritmo genético (AG) es hallar la política óptima, entonces cada individuo en el AG representa a un conjunto de acciones que conforman esa política. Como en el presente problema hay 41 estados y 41 posibles acciones para cada uno de los estados. El individuo, que es la política, se representa por medio de un vector fila de tamaño 41, en el cual la posición 0 del vector representa el estado 0, la posición 1 representa el estado 1 y así sucesivamente. El contenido de cada posición en el vector es la decisión tomada para ese estado, y puede tomar valores entre 1 y 41. Por ejemplo, si $D(4) = 10$, significa que en el estado 4 se optó por la acción o decisión 10, es decir, reemplazar un carro de edad 4 por uno de edad 8. El conjunto de valores en el vector de 41 posiciones representan la política. La tabla 1 muestra el ejemplo de un individuo.

Tabla 1. Ejemplo de un individuo (política)

Decisión $D(i)$	4	1	1	10	10	...	14	14	1	1	30
Estado i	0	1	2	3	4	...	36	37	38	39	40

4.3.2. Función de adaptación

Para cada representación de una política por un individuo, un valor de adaptación se asigna. El cálculo de este valor se hace por la siguiente fórmula:

$$f(D) = \sum_{i=0}^{40} q_i \pi_i \quad (6)$$

en la cual:

$f(D)$ es el valor de la función de adaptación para el vector de la política D. π_i es la probabilidad de estado estable para la cadena de Markov asociada a la política D. Es decir, a cada política se le halla la matriz de transición de la cadena de Markov asociada a la misma de acuerdo con (3) y (5), y a partir de ésta se le hallan las probabilidades de estado estable. q_i representa el costo esperado en el estado i y está determinado por (2) y (4) dependiendo de la acción que se tome.

4.3.3. Estrategias de selección y evolución

Para la selección se utilizó la Selección Normal Geométrica [9], ya que en las pruebas realizadas este criterio de selección fue el que mejor resultado dio, por encima de estrategias como la de ruleta y torneo, puesto que se observó que producían en generaciones sucesivas mejores individuos.

4.4. Resultados obtenidos

El algoritmo utilizado es el que se ilustró en el cuadro 1 al cual se le variaron los parámetros del mismo en diferentes pruebas y la que mejor resultado dio fue:

- Tamaño de la población: 80.
- Estrategia de selección utilizada: selección normal geométrica con q=0,08.
- Estrategia de cruce: cruce simple y cruce aritmético.
- Número de individuos para el cruce: 24, es decir el 30%.
- Estrategia de mutación utilizada: mutación uniforme y mutación iterativa.
- Número de individuos mutados en la mutación uniforme: 5 (6,25% de los individuos).
- Número de individuos mutados en la mutación iterativa: 2 (2,5% de los individuos).
- Números de genes mutados en la mutación uniforme: 10 (aproximadamente el 25%).
- Números de genes mutados en la mutación iterativa: 41 (100%).
- Número de generaciones: 30.

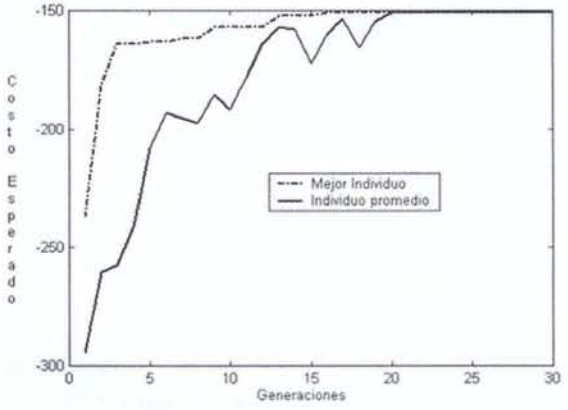
Con base en los anteriores parámetros y después de diversas pruebas se ejecutó el algoritmo genético. Como resultado, siempre se llegó al siguiente valor de la función de adaptación, la cual tiene un valor de función de adaptación igual a la rutina de mejoramiento de políticas:

$$f(D) = -150.945836$$

política D=(14, 14, 14, 14, 5, 5, 5, 14, 7, 5, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14)

La grafica1 muestra el comportamiento del AG. Se observa generación tras generación cuál es el mejor individuo de la población (línea punteada), así como el individuo promedio (línea sin puntear).

En esta gráfica también se observa la rápida convergencia a una buena solución, ya que en menos



Gráfica 1. Comportamiento del AG

de veinte generaciones se obtuvo una buena política; esto se debe a que en el operador genético de mutación iterativa se le aplicó una iteración completa de la rutina de mejoramiento de políticas, que asegura que la política que se obtiene después de esa iteración es igual o mejor a la anterior. Es de anotar que sólo se aplicó a dos individuos, por lo costoso de la iteración y lo lento que vuelve el proceso; no obstante, siempre se llegó a una buena política. Otra conclusión interesante que se obtuvo es que se produjeron diferentes políticas que las de la rutina de mejoramiento de políticas, que tienen el mismo valor de función de adaptación, lo cual plantea un problema de múltiples soluciones. Otras políticas con igual valor de función de adaptación son las siguientes:

D=(13, 14, 13, 15, 5, 5, 16, 12, 12, 7, 3, 15, 5, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 14, 14, 14, 14, 14, 14, 14, 17, 18, 12, 17, 12, 13, 14, 14)

D=(14, 17, 18, 16, 6, 6, 5, 16, 5, 3, 5, 15, 12, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14)

5. Implementación del algoritmo genético en el problema de un modelo de apuesta [10]

5.1. Presentación

En cada jugada del juego un apostador puede apostar una cantidad no negativa de su presente fortuna

y ganará o perderá esta cantidad con probabilidades p y q=1-p, respectivamente. Al apostador le está permitido hacer n apuestas, y su objetivo es maximizar la esperanza del logaritmo de su fortuna final. La idea es determinar una estrategia para optimizar este fin.

5.2. Solución analítica

La solución de este problema de manera analítica se encuentra en [10]. Se utilizan procedimientos sencillos de cálculo y se demuestra que si el apostador tiene una probabilidad de ganar de $p > 0{,}5$, su mejor estrategia es apostar p-q de su presente fortuna. Si $p \leq 0{,}5$ la mejor opción es apostar 0, es decir, no apostar.

Es de anotar que por la característica de este problema no es factible aplicar la rutina de mejoramiento de política para hallar una buena política, ya que el conjunto de acciones no es finito.

5.3. Solución por algoritmos genéticos

5.3.1. Forma de representación

Como en el presente problema la idea es determinar qué proporción de su presente fortuna debe apostar el jugador en cada una de las n apuestas que le son permitidas, entonces cada uno de los individuos que conforman la población en el algoritmo genético se representan por medio de un vector fila de tamaño n, en el cual cada posición del vector es un valor real entre 0 y 1 que indica la proporción de la presente fortuna que debe apostar en cada momento. Por ejemplo, para n=5, tenemos el siguiente D=(0,3, 0, 2, 0,5, 0,4, 0,53), el cual significa que el apostador apuesta 30% de su presente fortuna en la primera apuesta, en su segunda hace una apuesta del 20% de su presente fortuna y así sucesivamente. La idea es encontrar el vector D que maximiza la esperanza del logaritmo de su fortuna final, es decir, después de la n-ésima apuesta; obviamente, esto depende de los valores de p y q. La tabla 2 muestra lo que sería el ejemplo de un individuo con n=10 para este ejemplo.

Tabla 2. Ejemplo de un individuo para el problema del modelo de apuesta

Decisión $D(i)$	0,4	0,3	0,25	0,45	0,55	0,15	0,6	0,3	0,4	0,7
i-ésima apuesta	1	2	3	4	5	6	7	8	9	10

5.3.2. Función de adaptación

Para cada representación de una estrategia por un individuo, se le asigna un valor de adaptación, que es la ganancia promedio esperada del apostador después de hacer la n-ésima apuesta. El cálculo de ese valor es hecho por la siguiente relación recursiva:

$$
\begin{aligned}
&V_0(x) = \log(x) \\
&V_1(x) = pV_0(x + D(1)x) + qV_0(x - D(1)x) \\
&V_2(x) = pV_1(x + D(2)x) + qV_1(x - D(2)x) \\
&\cdot \\
&\cdot \\
&V_n(x) = pV_{n-1}(x + D(n)x) + qV_{n-1}(x - D(n)x) \\
&f(D) = V_n(x)
\end{aligned}
\qquad (7)
$$

en la cual:

$f(D)$ es el valor de la función de adaptación para el vector de la política D.

$V_i(x)$ es la ganancia esperada para el apostador después de la i-ésima apuesta.

x es la fortuna presente que tiene el apostador.

5.3.3. Estrategias de selección y evolución

Para la selección se utilizó la normal geométrica y para la evolución de las poblaciones se utilizaron alguno de los operadores genéticos que trae incorporado el GAOT. De acuerdo con la prueba utilizada, en el siguiente apartado se mencionan, qué operador se usó. Para la mutación se implementó un operador denominado *gamblingmutation* que muta todos los genes sumándole un número aleatorio que siguen una distribución normal con media 0 y

desviación 0,02, a fin de que el valor actual del gen pudiera aumentar o disminuir pero en valores muy pequeños. Igualmente, se utilizaron estrategias de mutación que trae incorporadas GAOT tal y como *inversionMutation, adjswapMutation, shiftMutation, swapMutation, threeswapMutation.*

5.3.4. Resultados obtenidos

El algoritmo utilizado es el que se mostró en el cuadro 1, al cual en las diferentes pruebas se le variaron los parámetros del mismo. Estas pruebas se realizaron con los siguientes parámetros:

- n=10.
- Tamaño de la población: 80.
- Estrategia de selección utilizada: selección normal geométrica con q=0,08.
- Estrategia de cruce: cruce simple y cruce aritmético.
- Número de individuos para el cruce: 24, es decir, el 30%.
- Estrategia de mutación utilizada: *inversionMutation, adjswapMutation, shiftMutation, swapMutation, threeswapMutation y GamblingMutation.*
- Número de individuos mutados: dos en las cinco primeras estrategias de mutación; cinco en la *GamblingMutation.*
- Números de genes mutados en *GamblingMutation:* 10 (100%).
- Número de generaciones: 130.

p: varía de acuerdo a la prueba.

A continuación se relacionan las pruebas.

Conjunto de pruebas No. 1. Se realizaron tres pruebas con p=0.6, y se obtuvieron los siguientes resultados:

- Número 1: D=(0.1957, 0.1957, 0.1957, 0.1957, 0.1957, 0.1957, 0.1957, 0.1957, 0.1957, 0.1957) $f(D)$= 4.80643079684621.
- Número 2: D=(0.1999, 0.1999, 0.1999, 0.1999, 0.1999, 0.1999, 0.1999, 0.1999, 0.1999, 0.1999) $f(D)$= 4.80652529715884.
- Número 3: D=(0.2006, 0.2006, 0.2006, 0.2006, 0.2006, 0.2006, 0.2006, 0.2006, 0.2006, 0.2006) $f(D)$= 4.80652305459128.

Del análisis de los anteriores resultados se puede concluir que con una probabilidad de ganar de p=0,6, la mejor estrategia que debe hacer el jugador es apostar 0,2 (20%) de la presente fortuna en todo momento, ya que esto maximiza el logaritmo de su fortuna final. Este resultado concuerda con el resultado analítico obtenido.

Conjunto de pruebas No 2. Se realizaron dos pruebas con p=0,75, y se obtuvieron los siguientes resultados:

- Número 1: D={0.5003, 0.5003, 0.5003, 0.5003, 0.5003, 0.5003, 0.5003, 0.5003, 0.5003, 0.5003} $f(D)$= 5.91328967928355.
- Número 2: D={0.4979, 0.4979, 0.4979, 0.4979, 0.4979, 0.4979, 0.4979, 0.4979, 0.4979, 0.4979} $f(D)$= 5.91326358672105

Del análisis de los anteriores resultados se puede concluir que con una probabilidad de ganar de p=0,75, la mejor estrategia que debe hacer el jugador es apostar 0,5 (50%) de la presente fortuna en todo momento, ya que esto maximiza el logaritmo de su fortuna final. Este resultado concuerda con el resultado analítico obtenido.

Conjunto de pruebas No. 3. Se realizaron dos pruebas con p=0,4 y p=0,2, y se obtuvo el siguiente resultado:

D={0, 0, 0, 0, 0, 0, 0, 0, 0, 0} $f(D)$=4.60517018598809.

El análisis del anterior resultado permite inferir, que para valores $p < 0,5$, la mejor estrategia, para el jugador es no apostar; concordando con el resultado analítico obtenido.

De la misma forma se puede concluir que para $p > 0,5$, la mejor estrategia es apostar p-q de su fortuna actual.

6. Conclusiones

En este trabajo se demostró que es factible encontrar con algoritmos genéticos buenas políticas cercanas a óptimas en procesos de decisión de Markov. En las pruebas experimentales que se plantearon, los algoritmos genéticos produjeron resultados al menos tan buenos como los producidos por las estrategias tradicionales disponibles (calculo e iteración de políticas) para estos problemas.

Asimismo, se comprobó que la combinación de técnicas, como la de iteración de políticas con el algoritmo genético le da, en el caso probado, más eficiencia a la búsqueda de soluciones para ciertos tipos de problemas de estos modelos, como se observó en el problema del reemplazo del automóvil en el cual la combinación de estas dos técnicas produjo buenas políticas diferentes a la que produce la rutina de mejoramiento de políticas.

Igualmente, se probó en los problemas resueltos que conceptualmente es más fácil realizar este tipo de búsquedas para estos modelos con algoritmos genéticos que por métodos desarrollados para los mismos, ya que éstos requieren un especial conocimiento y entrenamiento; en cambio, las operaciones que maneja el algoritmo genético son simples y fáciles de entender.

Si bien, la estructura del algoritmo genético propuesto se probó en un problema que es relativamente pequeño, es factible utilizar el mismo en problemas mucho más grandes, en los cuales son inaplicables técnicas como la rutina de mejoramiento de políticas por lo costosa que es, o en problemas en los cuales las probabilidades de transición no están disponibles y hay que aplicar otras técnicas para derivar una política óptima tal como el problema del modelo de apuesta.

Para futuros trabajos en esta rama se recomienda investigar sobre la aplicación de algoritmos genéticos para realizar búsquedas de políticas probabilísticas en un proceso de decisión de Markov, en la cual cada decisión que se tome en cada estado tiene asociada una probabilidad.

Referencias bibliográficas

[1] Puterman, M. L. (1994) *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.

[2] Bellman, R. and Dreyfus, S. (1962) *Applied Dynamic Programming*. Princeton University Press, p. 297-320.

[3] Hillier, F. and Lieberman, G. (1994) *Introducción a la investigación de operaciones*. México: McGraw Hill, 5a ed., p. 833-863.

[4] Howard, R. (1960) *Dynamic Programming and Markov Process*. [Cambridge, MA]: MIT press.

[5] Hansen E. and Zilberstein, S. (1999) *A Heuristic Search Algorithm for Markov Decision Problems* [online]. Recuperado el 3 de noviembre de 2003. Disponible en <http://citeseer.nj.nec.com/ 338309.html>.

[6] Salas R. *Simulated Annealing* para la busqueda de políticas óptimas en procesos de decisión de Markov [online]. 2006. Revista Vínculos, ed. 3. Disponible en <http://www.udistrital.edu.co/comunidad/ dependencias/revistavinculos/VINCULOS/articulos/3ed/Roberto%20salas/SIMULATED%20ANNEALING.pdf>.

[7] Barash, D. (1999) *A Genetic Search In Policy Space For Solving Markov Decision Processes* [online]. [Livermore, California]: Department of Applied Science University of California. Recuperado el 27 de octubre de 2003. Disponible en <http://citeseer.nj.nec.com/barash99genetic.html>.

[8] Fogel, D. (1995) *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press.

[9] Houck, C., Joines J. and Kay, M. *A Genetic Algorithm for Function Optimization: A Matlab Implementation*. North Carolina State University. Disponible en <http://www.ie.ncsu.edu/mirage/GAToolBox/gaot/papers/gaotv5.ps>. 2004.

[10] Ross, S. (1983) *Introduction to Stochastic Dynamic Programming*. Academic Press, Inc., p. 1-27.