IEM-I-09-04

METODOLOGÍA DE SINTONIZACIÓN DE SISTEMAS DE INFERENCIA DIFUSOS MEDIANTE ALGORITMOS HÍBRIDOS, PARA LA APROXIMACIÓN DE LA FUNCIÓN DE VALOR, EN EL PROBLEMA DE APRENDIZAJE POR REFUERZO

HAROLD ROBERTO MARTÍNEZ SALAZAR

UNIVERSIDAD DE LOS ANDES

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

BOGOTÁ D.C.

2004

IEM-I-09-04

METODOLOGÍA DE SINTONIZACIÓN DE SISTEMAS DE INFERENCIA DIFUSOS MEDIANTE ALGORITMOS HÍBRIDOS , PARA LA APROXIMACIÓN DE LA FUNCIÓN DE VALOR, EN EL PROBLEMA DE APRENDIZAJE POR REFUERZO

HAROLD ROBERTO MARTÍNEZ SALAZAR

Proyecto de grado para optar al titulo de Magíster en Ingeniería Electrónica

Asesor

Alain Gauthier S.

PhD., M.Sc. Ingeniero Eléctrico

UNIVERSIDAD DE LOS ANDES

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

BOGOTÁ D.C.

2004

IEM-I-09-04

Bogotá D.C. 3 de Agosto de 2004

Doctor

Roberto Bustamante.

Director Departamento de Ingeniería Eléctrica y Electrónica

UNIVERSIDAD DE LOS ANDES

La Ciudad.

Apreciado Doctor:

Por medio de la presente someto a consideración suya el Proyecto de Grado titulado ”Metodología de Sintonización de Sistemas de Inferencia Difusos Mediante Algoritmos Híbridos, para la Aproximación de la Función de Valor, en el Problema de Aprendizaje por Refuerzo”, que tiene como objetivo plantear una metodología para utilizar un sistema de inferencia Takagi-Sugeno (TKS) como aproximador de la función de valor del problema de aprendizaje por refuerzo, con el fin de explotar las ventajas de los algoritmos de sintonización híbridos utilizados para ajustar los parámetros de los sistemas TKS. Considero que este Proyecto de Grado cumple con los objetivos propuestos y por lo tanto lo presento como requisito parcial para optar por el título de Magíster en Ingeniería Electrónica.

Cordialmente,

Harold Roberto Martínez Salazar

COD. 200227645

Bogotá D.C. 3 de Agosto de 2004

Doctor

Roberto Bustamante.

Director Departamento de Ingeniería Eléctrica y Electrónica

UNIVERSIDAD DE LOS ANDES

La Ciudad.

Apreciado Doctor:

Someto a consideración suya la tesis de magíster titulada "Metodología de Sintonización de Sistemas de Inferencia Difusos Mediante Algoritmos Híbridos, para la Aproximación de la Función de Valor, en el Problema de Aprendizaje por Refuerzo", desarrollada por el estudiante Harold Roberto Martínez Salazar. Certifico como asesor, que este trabajo de investigación cumple con los objetivos propuestos y por lo tanto califica como requisito para optar por el título de Magíster en Ingeniería Electrónica.

Cordialmente,

Alain Gauthier S.

AGRADECIMIENTOS

Alain Gauthier S. Ph.D. M.Sc. Ingeniero Eléctrico. Coordinador de la Maestría de Ingeniería Eléctrica y Electrónica. Universidad de los Andes. Asesor. Su valiosa colaboración y aportes fueron de vital importancia en la elaboración de éste proyecto.

CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

RESUMEN

En este Trabajo de grado se presenta una metodología para utilizar un sistema de inferencia Takagi-Sugeno (TKS) como aproximador de la función de valor del problema de aprendizaje por refuerzo (AR), con el fin de explotar las ventajas de los algoritmos de sintonización híbridos, utilizados para ajustar los parámetros de los sistemas TKS. En el primer capítulo se describen las características del aprendizaje por refuerzo y los algoritmos más conocidos y utilizados. En el segundo capítulo se hace una descripción de los sistemas de inferencia difusa. En el tercer capítulo se presenta como puede mezclarse estas dos técnicas de la computación suave para utilizar las técnicas de AR en espacios continuos, y los problemas que muestran los resultados de este ejercicio; por último las conclusiones señalan las desventajas y desventajas encontradas en este desarrollo así como también posibles desarrollos futuros.

1. APRENDIZAJE POR REFUERZO

1.1. INTRODUCCIÓN

El aprendizaje por refuerzo (AR) se ha planteado como una metodología para hacer que las maquinas aprendan a realizar sus funciones [1] [2] [3]. Está basado en las relaciones que generan los seres vivos con el ambiente. No se lleva a cabo diciéndole a la máquina que acciones debe tomar como en la mayoría de algoritmos, en lugar de esto, el agente debe descubrir que acciones reciben mayor recompensa intentándolas. Una de las características más importantes es que las acciones tomadas no solo pueden afectar la recompensa inmediata, sino también, la recompensa futura. Estas características de ensayo y error, búsqueda y recompensa futura, son unas de las características más importantes del aprendizaje por refuerzo.

En el problema de aprendizaje se deben tener en cuenta los aspectos más importantes que definen la situación real, estos determinan la interacción del agente con su ambiente y una medida del éxito en el proceso de búsqueda de la meta por parte del agente. Información de la interacción del agente con el ambiente es establecida en la señal de estado. La medida del éxito del agente en su búsqueda de las metas esta determinada por la función de valor, la cual es

función de la señal de estado. Por este motivo el agente debe ser capas de reconocer el estado en que se encuentra y las acciones que elija deben generar un cambio del estado del mismo. La formulación del problema necesita incluir estos tres aspectos: sensaciones, acciones y metas, de la forma más sencilla sin trivializarlos.

Ilustración 1. Relación entre el Agente y el Ambiente.

1.2. ELEMENTOS DEL APRENDIZAJE POR REFUERZO

Sin embargo, más allá de estos elementos primordiales, se pueden encontrar más elementos que juegan en el proceso de aprendizaje del agente, como lo son: la política, la función de recompensa, la función de valor y/o el modelo de ambiente.

La política define el aprendizaje alcanzado por el agente en algún instante. Esta es la encargada de asociar a cada estado una acción que debe tomar el agente. La política es la clave del aprendizaje por refuerzo, pude ser una simple tabla de búsqueda o comprender un complejo algoritmo de selección.

La función de recompensa define las metas, asocia a cada estado en el que se encuentra el agente un valor de recompensa, indicando indirectamente que tan deseable es. A la asociación de acumulación de las recompensas que espera recibir el agente, hasta alcanzar la meta establecida en el problema, se le conoce como función de valor.

El modelo de ambiente intenta replicar las respuestas del entorno y sus transiciones, permitiendo predecir el estado al que llegará el agente, conociendo de antemano que éste se encontraba en un estado inicial y tomó una acción.

1.3. ASPECTOS IMPORTANTES EN EL APRENDIZAJE POR REFUERZO

Una de las características más importantes que diferencian el aprendizaje por refuerzo de otras técnicas de inteligencia artificial, es que evalúa las acciones tomadas por el agente en lugar de indicar las acciones que debe tomar. Por este motivo se genera una necesidad de exploración del ambiente. La retroalimentación de la evaluación indica que tan buena es la acción desarrollada por el agente, sin embargo no indica cuál es la mejor o peor acción posible.

1.3.1. Explotación Versus Exploración

En los algoritmos de aprendizaje de máquina se manejan dos aspectos muy importantes que pueden generar una mayor generalización o mejor aprendizaje,

estos aspectos son conocidos como exploración y explotación. Las acciones que toma un agente tienen una recompensa esperada, a la máxima recompensa se le conoce como una acción golosa, cuando el agente toma esta acción esta explotando el conocimiento adquirido hasta ese momento. Si en lugar de esto el agente selecciona una acción que no le genera la mayor recompensa esperada, se dice que esta explorando el ambiente. La exploración garantiza que el agente encuentre una respuesta máxima a largo plazo, mientras que la explotación maximiza la recompensa inmediata.

En estos algoritmos se deben balancear ambos aspectos para lograr encontrar una política con un desempeño adecuado. Generalmente se utiliza una política conocida como e-greedy, la cual permite una elección aleatoria de la acción que es función de la recompensa encontrada hasta ese momento. La elección exitosa hasta el momento es más probable que las que no han sido, esta política permite una exploración del ambiente realizada por el agente. Aunque esta es la más conocida no es la única, también es posible la comparación reforzada o los métodos de persuasión. La primera intenta variar la probabilidad en función de la recompensa obtenida por la elección de la acción. La probabilidad de la acción será entonces igual a la probabilidad anterior más un factor conocido como □. que escala la comparación de la recompensa con una recompensa esperada. Los segundos al igual que la primera intentan variar la probabilidad de elección de la acción en función de la recompensa, sin embargo no tienen en cuenta una recompensa de referencia, simplemente, aumentan la probabilidad de la mejor acción y disminuyen la probabilidad de las otras acciones.

e-greedy	$P_{t+1}(a^*) = 1 - e \quad ; \quad P_{t+1}(a) = e/(n-1)$ P_t= probabilidad de selección de acción r_t= recompensa a^*= acción con mejor recompensa a= acción con menor recompensa e= parámetro de elección. N= número de acciones.
Comparación por refuerzo.	$P_{t+1}(a) = P_t(a) + \beta(r_t - \bar{r}_t) \quad ; \quad \bar{r}_{t+1} = \bar{r}_t + \alpha(r_t - \bar{r}_t)$ P_t= probabilidad de selección de acción r_t= recompensa $\bar{r}_t$= recompensa de referencia
Método de persuasión	$P_{t+1}(a^*) = P_t(a^*) + \beta(1 - r_t) \quad ; \quad P_{t+1}(a) = P_t(a) + \beta(0 - r_t)$ P_t= probabilidad de selección de acción r_t= recompensa a^*= acción con mejor recompensa a= acción con menor recompensa

Tabla 1. Ajuste de política de acción.

1.3.2. Propiedad de Markov

La propiedad de Markov es muy importante en el aprendizaje por refuerzo. El agente basa sus decisiones en la señal del ambiente llamada el estado de

ambiente, esta señal contiene toda la información disponible para él, la cual no es más que la información de los sensores que porta éste. Idealmente se busca que la señal de estado resuma toda la información relevante del pasado, esto generalmente requiere más que puramente las señales de los sensores, sin embargo se necesita menos que los datos históricos. La señal de estado que alcance a cumplir con el objetivo de guardar esta información relevante es Markov o que tiene la propiedad de Markov. Para nuestro problema de aprendizaje por refuerzo se puede decir que una señal de estado cumple con la propiedad de Markov si la respuesta siguiente del ambiente depende únicamente de la acción del agente y de la señal de estado. Esto es muy importante en el aprendizaje por refuerzo debido a que la función de valor y la función de recompensa, son función de la señal de estado.

1.4. APRENDIZAJE DE MÁQUINA UTILIZANDO APRENDIZAJE POR REFUERZO

Casi todos los algoritmos de aprendizaje por refuerzo están desarrollados para estimar la función de valor, la cual indica la conveniencia de que un agente esté en ese estado o que tan conveniente es para el agente tomar una acción en el mismo. La medida de esta conveniencia, esta dada por la acumulación de recompensas que recibirá el agente hasta que alcance su meta. En lugar de trabajar con la función de valor se puede trabajar con la función de valor-acción. Una buena estimación de la función de valor permite que el agente encuentre una política que maximice la recompensa.

1.4.1. Programación Dinámica

Un método que se ha desarrollado para estimar la función de valor es el de programación dinámica, la idea es utilizar la función de valor para organizar y estructurar la búsqueda de buenas políticas. Se puede encontrar fácilmente las políticas óptimas una vez se conozcan la función de valor óptima, la cual cumpla con la ecuación de optimización de Bellman.

Ecuación de optimización para la función de valor	Ecuación de optimización para la función de valor estado-acción
$V^*(s) = \max_a E\left[r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a\right]$ $V^*(s) = \max_a \sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma V^*(s')\right]$	$Q^*(s,a) = E\left[r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a') \mid s_t = s, a_t = a\right]$ $Q^*(s,a) = \sum_{s'} P^a_{ss'}\left[R^a_{ss'} + \gamma \max_{a'} Q^*(s', a')\right]$

Tabla 2. Ecuaciones de optimización de Bellman.

Lo primero que se hace en este algoritmo es definir la política de acción del agente y con ésta se estima la función de valor.

Para producir el valor de la función en un tiempo siguiente se toma el valor de la función un tiempo anterior en cada uno de los estados. El valor de la función será cambiado por el de valor de los estados siguientes mas la recompensa esperada a lo largo de todas las transiciones posibles bajo la política establecida sobre todos los estados. Esta operación es conocida como *full backup*, debido a que se hace

una copia del valor de la función para cada estado para así aproximar la función de valor.

La convergencia del algoritmo se puede medir al ver la diferencia de la función de valor después de cada iteración. Cuando el cambio es muy pequeño se puede finalizar el algoritmo. La razón por la cual se encuentra la función de valor para una política es para evaluar su conveniencia y optimizarla. Con esta nueva política se halla de nuevo la función de valor para reiniciar el proceso.

Inicializar V(s) y π(s) arbitrariamente s. 2. Repetir : $\Delta \leftarrow 0$ Para cada s. v$\leftarrow$V(s) V(s)$\leftarrow \Sigma_s^f P^{\pi(s)}_{ss}{}^f [R^{\pi(s)}_{ss}{}^f + \gamma V(s^f)]$ $\Delta \leftarrow$max (Δ,abs(v-V(s)) Hasta que $\Delta < \theta$ (positivo pequeño) 3. Política-estable$\leftarrow$verdadero Para cada s b$\leftarrow$ π(s) π(s)$\leftarrow$ arg max$_a$ $\Sigma_s^f P^{\pi(s)}_{ss}{}^f [R^{\pi(s)}_{ss}{}^f + \gamma V(s^f)]$ si b es igual a π(s), la política es estable entonces pare si no ir a 2

Tabla 3. Algoritmo de programación dinámica.

1.4.2. Monte Carlo

Monte Carlo es un método que permite estimar las políticas óptimas y la función de valor de estas. Este método solo necesita experiencia, ejemplos de secuencias, estados, acciones, recompensas e interacciones con el ambiente. Está basado en

respuestas promedio, para asegurar que se van a tener respuestas bien definidas. Se establece el método de Monte Carlo solo para épocas, se asume que la experiencia esta dividida en episodios y que estos se terminan eventualmente sin importar las acciones que se realicen. Al final de cada época los valores estimados y la política de acción son modificados.

Al igual que en el método anterior hay que suponer una política de acción, con ésta el agente inicia la interacción con el medio, generando recompensas hasta terminar el episodio, por cada estado que pasó el agente se generó una señal de recompensa, estas señales que se generan en promedio convergerán al valor real de la función de valor. Después de estimar la función de valor se genera una política de acción basada en la estimación y siguiéndola se repite el proceso.

Inicializar Q(s,a) y π(s) arbitrariamente.
Recompensa(s,a) vacío.
Repetir siempre:
  Generar un episodio usando π.
  Por cada par s,a que aparezca en el episodio
    R← recompensa seguida de la primera ocurrencia de s,a.
    Agregar R a Resultado Recompensa(s,a)
    Q(s,a)← promedio(Recompensa(s,a))
  Para cada s en el episodio
    π(s)← arg $\max_a$ Q(s,a)

Tabla 4. Algoritmo de Monte Carlo.

1.4.3. Aprendizaje por Diferencia Temporal

Un método muy importante en el aprendizaje por refuerzo es el aprendizaje por diferencia temporal (TD), éste es una combinación entre las ideas de Monte Carlo y la programación dinámica, se puede desarrollar un aprendizaje a partir de la experiencia y basar los estimativos de la función de valor en otros realizados anteriormente. Monte Carlo estima el valor del estado hasta finalizar el episodio, hallando la recompensa recibida al alcanzar la meta, mientras que la programación lineal utiliza un estimativo del valor del estado futuro y la recompensa inmediata.

TD ajusta el valor del estado de la misma forma como lo hace el algoritmo de Monte Carlo, pero toma el estimativo del estado como lo hace el algoritmo de programación dinámica utilizando la recompensa por la acción y el valor de estado futuro, haciendo el ajuste del valor de estado para cada acción sin tener que esperar hasta el fin del episodio.

La política de acción se elige en función del valor que se le asocia a cada estado, puede ser una política en la que varía la elección de la acción en forma probabilística como e-greedy o simplemente una que elija la acción que recibe máxima recompensa.

Cuando se utiliza la función de valor estado-acción, se pueden generar dos variantes SARSA o Q-learning la diferencia entre estas dos es muy simple, lo único que varía es la aproximación de la recompensa. Q-learning utiliza la máxima recompensa que se puede tener en ese estado, mientras que SARSA toma el valor de la recompensa dada por la acción tomada siguiendo la política de acción.

SARSA	Q-learning
Inicializar Q(s,a) arbitrariamente. Repetir para cada episodio: Inicializar s. Repetir por cada paso del episodio: Elegir a de s con la política obtenida de Q (e.g. e-greedy) Repetir para todos s Tomar la acción a, observar r y s^f. Elegir a^f de s^f con la política obtenida de Q $Q(s,a) \leftarrow Q(s,a)+\alpha[r+\gamma Q(s^f,a^f)-Q(s,a)]$ $s \leftarrow s^f$ $a \leftarrow a^f$ Hasta s terminal	Inicializar Q(s,a) arbitrariamente. Repetir para cada episodio: Inicializar s. Repetir por cada paso del episodio: Elegir a de s con la política obtenida de Q (e.g. e-greedy) Repetir para todos s Tomar la acción a, observar r y s^f. $Q(s,a) \leftarrow Q(s,a)+\alpha[r+\gamma \max_a{}^f Q(s^f,a^f)-Q(s,a)]$ $s \leftarrow s^f$ Hasta s terminal

Tabla 5. SARSA y Q-learning

1.4.4. Tasas de Elegibilidad

Las tasas de elegibilidad son utilizadas para intentar acercar el algoritmo TD a Monte Carlo, con estas tasas de elegibilidad se intenta afectar la función de valor en los estados por los que el agente ya ha pasado. Para implementar este algoritmo se necesita de una nueva variable: la elegibilidad que indica hace cuanto el agente paso por ese estado.

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1} & s \neq s_t \\ \gamma\lambda e_{t-1} + 1 & s = s_t \end{cases}$$

Ecuación 1

Donde γ es el parámetro de decaimiento que hace que la variable de elegibilidad decaiga con el tiempo mientras que el estado no es visitado por el agente. En este nuevo algoritmo la estimación del valor de la función se hará sobre todos los estados ponderando el factor de aproximación por la elegibilidad, de esta forma se tendrá en cuenta los estados que recientemente se hallan visitado. En el momento en que el parámetro de decaimiento sea igual a uno, se estará llevando a cabo el algoritmo de Monte Carlo. Este mecanismo puede ser introducido también en los algoritmos planteados para la función de valor estado-acción.

Inicializar V(s) arbitrariamente, y e(s) para todo s.
Repetir para cada episodio:
  Inicializar s.
  Repetir por cada paso del episodio:
    a ← acción tomada por π en s
    tomar la acción a, observar la recompensa r y el estado futuro s^f
    δ ← r+γV(s^f)-V(s)
    e(s)←1+e(s)
    Repetir para todos s
      V(s)←V(s)+$\delta\gamma$e(s)
      e(s)←$\gamma\lambda$e(s)
    s ← s^f
  Hasta s terminal

Tabla 6. Algoritmo TD(λ)

2. SISTEMAS DIFUSOS

Un sistema de inferencia difuso [3] [4] es un sistema basado en reglas, que utiliza lógica difusa para evaluar la información en lugar de la lógica booleana. La lógica difusa es una herramienta matemática que permite manipular información de la forma como comunican y realizan los procesos de razonamiento los seres humanos. Esta basada en que un elemento, puede ser parcialmente verdadero o falso. Por ejemplo la expresión: “estoy cerca de la nevera”, donde el valor difuso, cerca, está aplicado a la variable difusa distancia, además de ser imprecisa, la expresión esta sujeta a interpretación. Los fundamentos de la lógica difusa fueron establecidos por Lotfi Zadeh en 1965.

2.1. NOCIONES BÁSICAS DE SISTEMAS DIFUSOS

2.1.1. Conjuntos Difusos

Un conjunto concreto A en un universo de discurso U, puede ser definido enumerando cada uno de sus elementos o definiendo las características que los identifican $x \in A$ (i.e., A={x | x cumple con cierta condición}). La función característica o función de pertenencia asociada con A asigna cada uno de los

elementos del universo, $\mu_A \in . U \rightarrow \{0,1\}$, a cada uno de los elementos que pertenecen al conjunto lo asocia con uno, los otros elementos se asocian con cero.

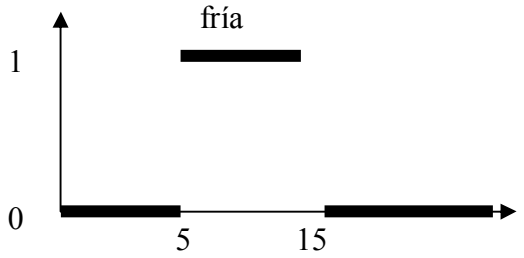


Ilustración 2. Ejemplo de conjunto concreto. El conjunto concreto esta definido por fría ={x | 5<x<15}

Los conjuntos difusos son una generalización de los conjuntos concretos. Un conjunto difuso F definido en un universo de discurso U esta caracterizado por una función de pertenencia $\mu_F(x)$, la cual toma valores en el intervalo cero a uno. La función de pertenencia tiene más sentido en el contexto difuso ya que $\mu_F(x)$ representa el grado en que x es miembro de F. A la operación que asigna un grado de pertenencia a x se le conoce como fusificación.

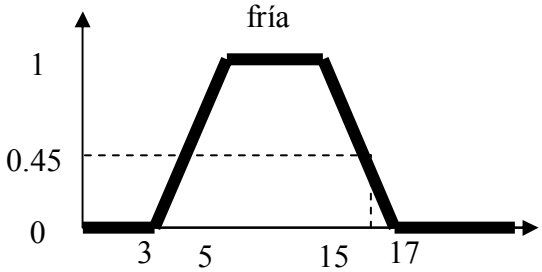
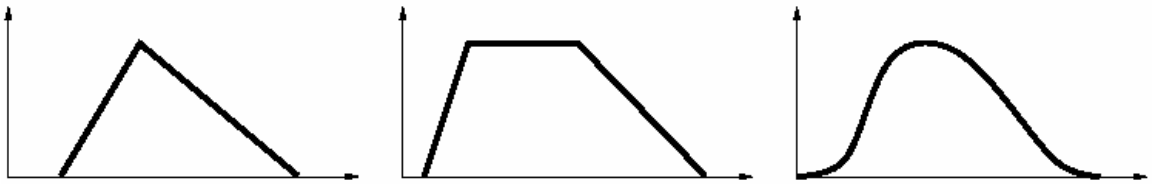


Ilustración 3. Ejemplo de un conjunto difuso. El conjunto difuso esta definido por fría = {x | x esta entre 5 y 15} como ejemplo se muestra $\mu_F(16)=0.45$

Las funciones de pertenencia pueden tener cualquier forma, sin embargo generalmente son suaves y monotónicas. Esto se debe a que los universos de discurso son generalmente usados para representar variables lingüísticas descritas en el contexto de un universo de discurso coherente (características similares son representadas por elementos cercanos, como es el caso de variables físicas), las formas más comunes son triangulares, trapezoidales y en forma de campana.



(1) (2) (3)

Ilustración 4. Funcione s de pertenencia (1) triangular, (2) trapezoidal, (3) campana

2.1.2. Lógica Difusa

Los sistemas de inferencia difusa están construidos sobre reglas que están expresadas como implicaciones lógicas. Para desarrollar de una manera formal la lógica difusa se puede tomar ventaja de que “está bien establecido que la lógica preposicional es isomórfica de la teoría de conjuntos bajo la correcta correspondencia entre los componentes de estos dos sistemas matemáticos. Aún más, estos dos sistemas son isomórficos al álgebra booleana”.

Conjuntos	*Lógica*	*Álgebra*
Pertenencia	Verdad	Valor
Miembro	Verdadero	1
No-miembro	Falso	0
Unión	Or	Suma
Intersección	And	Producto
Complemento	Not	Complemento

Tabla 7. Equivalencias entre los dominios isomórficos

Esta equivalencia se puede ver fácilmente si se expresa la relación $x \in A$ utilizando la preposición "x es A", por ejemplo, el agua esta fría es una forma de expresar agua $\in$fría donde fría ={x | x está entre 5 y 15 grados centígrados}, la diferencia entre la interpretación concreta o difusa de esta preposición está determinada por la definición de la función de pertenencia μ_{FRIA}. Asumiendo que el agua se encuentra a 16 grados, el valor de verdad de "el agua esta fría" observando la figura 1 y 2 se encuentra que el valor es 0 y 0.45 respectivamente. Así como los conjuntos difusos son una generalización de los conjuntos concretos la lógica difusa es una generalización de la lógica booleana.

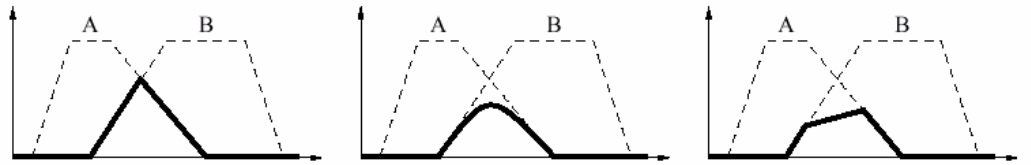
2.1.3 Operaciones

En esta sección se muestran las generalizaciones de las operaciones realizadas en los conjuntos discretos, sin embargo estas generalizaciones no son únicas, se han realizado varios operadores que pretenden realizar la misma operación en conjuntos concretos. Las operaciones más generales se presentan a continuación.

Operadores para intersección ($\mu_{A\cap B}(x)= \mu_A(x) \wedge \mu_B(x)$). También es conocida como t-norma los más conocidos son el mínimo, el producto y el producto limitado.

Mínimo	*Producto*	*Producto limitado*
$\text{Min}(\mu_A(x), \mu_B(x))$	$\mu_A(x) \bullet \mu_B(x)$	$\max(0, \mu_A(x)+ \mu_B(x)-1)$

Tabla 8 Operadores para intersección



(1) (2) (3)

Ilustración 5. t-normas comunes, (1) mínimo, (2) producto, (3) producto limitado.

Operadores para Unión ($\mu_{A\cup B}(x)= \mu_A(x) \vee \mu_B(x)$). También son conocidos como t-conorma, los operadores más conocidos son el máximo la suma probabilística y la suma limitada.

Máximo	*Suma probabilística*	*Suma limitado*
$\max(\mu_A(x), \mu_B(x))$	$\mu_A(x)+ \mu_B(x)- \mu_A(x) \bullet \mu_B(x)$	$\min(1, \mu_A(x)+ \mu_B(x))$

Tabla 9. Operadores para la Unión

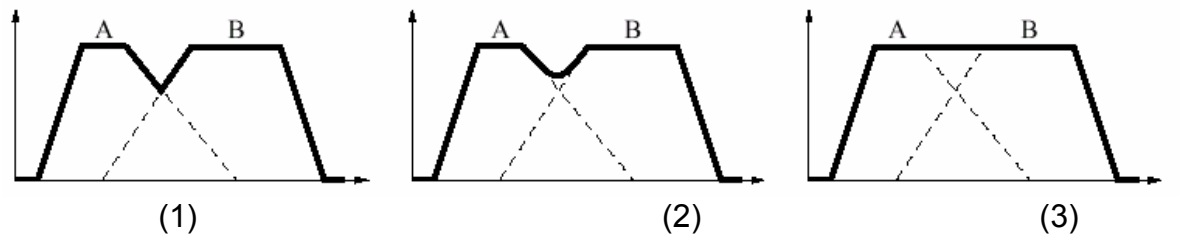


Ilustración 6. t-conormas comunes, (1) máximo, (2) suma probabilística, (3) suma limitada

Complemento ($\mu_{A'}(x) = \mu_{\sim A}(x)$). Esta operación a diferencia de las otras solo tiene una representación que se utiliza, generalmente conocida como complemento difuso.

Complemento difuso
$1-\mu_A(x)$

Tabla 10. Operador complemento

2.2. RAZONAMIENTO DIFUSO

2.2.1. Variables Lingüísticas

En el ejemplo utilizado para explicar la lógica difusa se utilizó un conjunto difuso para representar el concepto frió, se pueden plantear otros conjuntos difusos que

representen los conceptos de tibio y caliente. Todos estos conjuntos son valores lingüísticos que pertenecen a la variable lingüística temperatura.

Una variable lingüística también es llamada variable difusa, esta determinada por su nombre, un conjunto de valores lingüísticos y sus respectivas funciones de pertenencia.

2.2.2. Condiciones Difusas

Una preposición como “el agua esta fría” asigna un valor de verdad determinado por la función de pertenencia asociada al valor lingüístico fría. Esta preposición se conoce con el nombre de condición difusa, sin embargo, los sistemas de inferencia difusa dependen de mas de una variable, preposiciones como “el agua esta fría y la presión es baja”, son las que se utilizan generalmente. Una condición difusa doble, recibe el valor de verdad a la operación de unión de las dos condiciones difusas sencillas, “el agua esta fría” y “la presión es baja”, esto corresponde al producto cartesiano de los dos conjuntos difusos frió y baja, definidos en las variables lingüísticas de temperatura y presión.

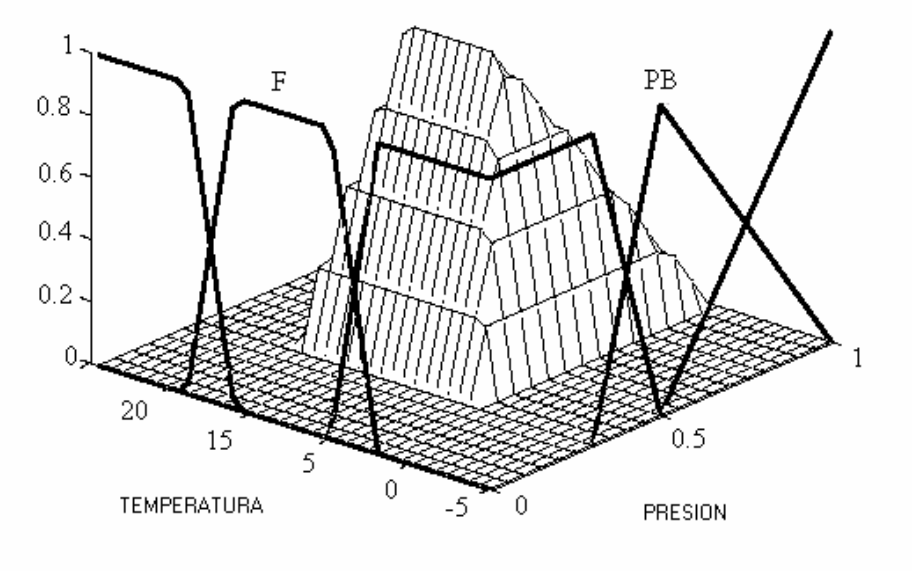


Ilustración 7. Función bidimensional de pertenencia. Se obtuvo al aplicar el mínimo sobre los valores difusos fría(F) y presión baja (PB) de las variables temperatura y presión.

2.2.3. Inferencia Difusa

Inferir esta definido como “concluir de hechos y razonamientos”. Esta definición presenta tres elementos del proceso de inferencia, los hechos son la información sin procesar, el proceso de razonamiento es el motor de inferencia y la conclusión es el resultado de este proceso. Estos pasos son tenidos en cuenta en el proceso de inferencia difusa.

Implicación: En un sistema de inferencia difusa las reglas tiene la forma de:

Si (condición difusa de entrada), entonces (asignación difusa de salida)

La asignación difusa de salida es el resultado de aplicar el operador de implicación causal, también conocido como motor de implicación, a los conjuntos difusos de entrada y salida. Las condiciones de entrada y de salida son llamadas antecedentes y consecuentes respectivamente.

Por ejemplo dada la regla

Si x es F entonces b es M

Donde $x \in X$ y $b \in B$, (X y B son los conjuntos de discurso de entrada y salida respectivamente). La operación de implicación $F \rightarrow M$ para un valor de entrada x está determinada por el conjunto difuso $M'=\mu_{F\rightarrow M}(x)$, definido en el conjunto de salida M, Los operadores mas comunes de implicación son el mínimo y el producto.

Mínimo	Producto
$M'=\min(\mu_F(x),M)$	$M'=\mu_F(x)\bullet M$

Tabla 11. Operadores de implicación

Cuando el antecedente no es cierto el valor de pertenencia será cero por lo tanto esta regla no se aportará en el proceso de inferencia del sistema.

Agregación: Los sistemas de inferencia difusa tienen mas de una regla y cuando se desarrolla el proceso de inferencia, varias de estas se van a activar proponiendo un conjunto difuso de salida con un cierto valor de verdad, lo ideal es que un único conjunto difuso sea seleccionado como salida, por esto se utiliza el proceso de agregación para generar un solo conjunto difuso a partir de los

conjuntos que generan las reglas. Generalmente las t-conormas son utilizadas para realizar esta operación.

Defusificación: La salida de un sistema de inferencia difusa es un valor difuso, éste contiene demasiada información cualitativa sin embargo en la mayoría de casos lo único que se necesita es un valor concreto, el proceso que permite obtener un valor concreto de un conjunto difuso se le conoce como defusificación. Hay muchos métodos por los cuales se puede obtener este valor sin embargo, los más utilizados son: centroide y el promedio de los máximos.

Centroide	*Promedio de los máximos*
$y_c = \frac{\sum \mu_Y(v) \cdot v}{\sum \mu_Y(v)}$	$y_{\inf} = \min(y \mid \mu_Y(y) = \max(\mu_Y(v))$ $y_{\sup} = \max(y \mid \mu_Y(y) = \max(\mu_Y(v))$ $y_{MOM} = \frac{y_{\inf} + y_{\sup}}{2}$

Tabla 12. Métodos de defusificación

2.3. SISTEMAS DE INFERENCIA DIFUSA

Hay dos grandes grupos de sistemas de inferencia difusa los cuales se diferencian en el tipo de consecuente que utilizan, éstos son el sistema de inferencia tipo Mamdani y el sistema de inferencia tipo Takagi Sugeno.

2.3.1. Sistema de Inferencia Tipo Mamdani

Este tipo de sistema de inferencia difuso tiene como consecuentes conjuntos difusos, recibe este nombre porque fueron propuestos por Mamdani. Su ventaja es la gran interpretabilidad, debido a que cada conjunto difuso está asociado a un valor de una variable lingüística, sin embargo tiene como desventaja su alto costo computacional al llevar a cabo operaciones como la defusificación y su baja precisión.

2.3.2. Sistemas de Inferencia Tipo TKS

Este tipo de sistema de inferencia difuso obtuvo su nombre de Takagi, Sugeno y Kang quienes propusieron esta alternativa de sistemas difusos, en la cual el consecuente no es un conjunto difuso sino que es un función (generalmente lineal) de las variables de entrada. Estos sistemas son mas precisos que el descrito anteriormente, sin embargo se pierde la interpretabilidad debido a que las reglas ya no representan conceptos lingüísticos. Por otra parte si no hay límite en el número de reglas en un sistema TKS de primer orden, es decir cuando la función del consecuente es una constante tiene ilimitados poderes de aproximación, puede aproximar cualquier función no lineal.

Existen dos alternativas bastante probadas para sintonizar los parámetros de los sistemas de inferencia TS. La primera, utiliza como única herramienta el gradiente

descendente para ajustar cada uno de los parámetros, este método es un poco lento y no garantiza que el sistema realice la mejor aproximación. La segunda, parte el problema en dos, la sintonización de los parámetros lineales y la de los parámetros no lineales. Esta perspectiva utiliza entonces no solo el gradiente descendente, que es la herramienta para ajustar parámetros no lineales, sino que también utiliza mínimos cuadrados recursivos para ajustar los parámetros lineales, este método permite a los sistemas de inferencia difusa, aproximar mucho mejor la función objetivo con un número menor de iteraciones a las utilizadas por los métodos basados únicamente en gradiente descendente.

3. SISTEMA TKS COMO ESTIMADOR DE FUNCIÓN EN AR

Las dos técnicas que se han presentado en capítulos anteriores, hacen parte de las metodologías de aprendizaje de máquina [4]. Las fortalezas que se han desarrollado en estas técnicas pueden ser aprovechadas aún más si se mezclan entre si para poder dar solución a un problema y mejorar el rendimiento de los algoritmos planteados hasta el momento.

Un inconveniente que se puede ver en el planteamiento de los algoritmos utilizados para llevar acabo el aprendizaje por refuerzo, es la necesidad de discretizar el espacio en que se mueve el agente, así como también las acciones que debe realizar. Este aspecto de discretización se convierte en un punto muy importante que determinará el grado de aprendizaje que desarrolle el agente.

En cada uno de los algoritmos que se han explorado en AR se asume de antemano que se conocen todos los estados en los que el agente se va a encontrar, si no se tuviera acceso a esta información ninguno de los algoritmos se podría llevar a cabo. Sin embargo en la mayoría de los casos esta información no es sencilla de determinar o peor aun depende del tipo de resolución con la que el agente percibe el ambiente o de la resolución de la acción del mismo. Esta imprecisión no es tenida en cuenta en ninguno de los algoritmos anteriores, lo que

es muy riesgoso, ya que si por algún motivo, el estado al que llega no esta definido anteriormente tampoco estará definida la función de valor y por ende no podrá tomar una acción en ese momento. Poder trabajar con un espacio continuo al igual que con una acción continua enriquecería la búsqueda de la solución por parte del agente, mejorando sin duda el desempeño de éste al alcanzar sus metas.

Como se vio en el capítulo dos, los sistemas de inferencia difusa Takagi-Sugeno son aproximadores universales. Por este motivo es de esperarse que estos sean una buena alternativa para aproximar la función de valor y construir un algoritmo que aproveche las ventajas de convergencia que tiene los algoritmos híbridos (de ajuste de parámetros) aplicados a este tipo de sistemas de inferencia.

Cada uno de los canales de información por los cuales el agente percibe el ambiente serán tomados como una variable lingüística y los estados posibles del agente serán las reglas del sistema de inferencia difuso. Sin embargo estas reglas del sistema de inferencia no van a estar encargadas directamente de indicarle al agente como actuar, sino por el contrario, se encargarán de estimar la función de valor que utiliza la técnica de aprendizaje por refuerzo, para encontrar las acciones a medida que interactúa con el ambiente.

3.1. PLANTEAMIENTO DEL ALGORITMO

A diferencia de los algoritmos utilizados para sintonizar los sistemas de inferencia difusa, en este caso no conocemos el valor real de la función que se esta aproximando, solo conocemos una medida del error debido a la definición de la función de valor. El algoritmo se desarrollará a partir de la deducción de esta relación en espacio continuo [2].

Consideramos el sistema derterminístico continuo:

$$\dot{x} = f(x(t), u(t))$$
$$x \in X \subset R^n$$
$$u \in U \subset R^m$$

Ecuación 2. Definición del problema

Donde x es el estado y u es la acción. Se denota la función de recompensa inmediata para el estado y la acción como:

$$r(t) = r(x(t), u(t))$$

Ecuación 3. Recompensa

El objetivo es encontrar una política de acción:

$$u(t) = \mu(x(t))$$

Ecuación 4. Política

Que maximice la acumulación de recompensas futuras, para un estado inicial x(t).

$$V^{\mu}(x(t)) = \int_{t}^{\infty} e^{-\frac{s-t}{\tau}} r(x(s),u(s))ds$$

Ecuación 5. Función de valor

Donde x(s) y u(s) siguen la dinámica y la política establecidas. A V^{μ} (x) se le conoce como la función de valor del estado x y a τ se le llama constate de descuento de futuras recompensas.

La función de valor para la política optima esta definida como:

$$V^{*}(x(t)) = \max_{u(t,\infty]} \int_{t}^{\infty} e^{-\frac{s-t}{\tau}} r(x(s),u(s))ds$$

Ecuación 6. Función de valor para la política óptima

Donde u(t,∞] denota el curso de u(s) para (t<s<∞). De acuerdo con el principio de optimalidad, la condición para encontrar la función de valor óptima para el tiempo t es:

$$\frac{1}{\tau}V^{*}(x(t)) = \max_{u(t) \in U} [r(x(s),u(s)) + \frac{\partial V^{*}(x(t))}{\partial x} f(x(t),u(t)]$$

Ecuación 7. Condición de la función de valor óptima

Que es una versión con descuento de la ecuación de Hamilton-Jacobi-Bellman, la política esta determinada por la selección de la acción que maximiza el lado derecho de la ecuación.

Se va a utilizar un sistema de inferencia TKS para estimar la función de valor, este sistema esta descrito brevemente como se muestra a continuación. En el se puede ver, como cada una de las reglas, es un plano construido por la combinación lineal de las variables de estado. Los valores lingüísticos de cada una de las variables de estado están determinados por funciones de pertenencia gaussianas y el nivel de activación de cada una de las reglas esta determinado por el producto de las pertenencias de cada una de las variables de estado a un valor lingüístico dado.

$$V(x,\mu,A)=\sum_{j=1}^{total}\frac{\mu_j\left(A_{0j}+A_{j1}x_1+\ldots+A_{nj}x_n\right)}{\sum_{i=1}^{totla}\mu_i}$$

$$\mu_a=\exp\left[-0.5\left(\frac{x_1-c_{1k}}{\sigma_{1k}}\right)^{1/2}\right]\cdot\exp\left[-0.5\left(\frac{x_2-c_{2m}}{\sigma_{2n}}\right)^{1/2}\right]\ldots\exp\left[-0.5\left(\frac{x_n-c_{ns}}{\sigma_{ns}}\right)^{1/2}\right].$$

$$V^{\mu}(x(t))\cong V(x(t),w,A)$$

Ecuación 8. Función de aproximación

Donde w son los parámetros de la función aproximadora que serán ajustados mediante el método de gradiente descendente, en este caso son los parámetros c y σ. En el marco de la técnica de estimación TD, el ajuste de la función de valor es

llevado a cabo por medio de la condición que se obtiene al diferenciar la ecuación 5.

$$\dot{V}^{\mu}(x(t)) = \frac{1}{\tau} V^{\mu}(x(t)) - r(t)$$
$$\delta(t) = r(t) - \frac{1}{\tau} V^{\mu}(x(t)) + \dot{V}^{\mu}(x(t))$$

Ecuación 9. Definición continua del error TD

Esta es la definición continua del error TD. Para disminuir el error TD hacia cero se puede sintonizar el nivel de la función de valor, su derivada o ambos. Considerando la función objetivo:

$$E(t) = \frac{1}{2}\left|\delta(t)\right|^2$$

Ecuación 10. Error TD al cuadrado

Si aplicamos el gradiente, se obtiene:

$$\frac{\partial E(t)}{\partial w_i} = \delta(t)\left[-\frac{1}{\tau}\frac{\partial V(x;w)}{\partial w_i} + \frac{\partial}{\partial w_i}\left(\frac{\partial V(x;w)}{\partial x}\right)\dot{x}(t)\right]$$

Ecuación 11. Gradiente del error al cuadrado

De donde podemos ver que la regla para la sintonización de los parámetros por medio del gradiente descendente es:

$$w_i = w_{i-1} - \eta \frac{\partial E(t)}{\partial w_i}$$

Ecuación 12. Ecuación de ajuste

Lo que puede ser un buen mecanismo de ajuste, sin embargo resulta mejor pensar en ajustar los parámetros sin afectar los estimativos futuros, por este motivo se considera un método para implementar el "back-up" del error TD, para esto se utiliza la aproximación de Euler por atrás de la derivada con respecto al tiempo de la función de valor. De aquí obtenemos que el error TD es:

$$\delta(t) = r(t) + \frac{1}{\Delta t}\left[\left(1 - \frac{\Delta t}{\tau}\right)V(t) - V(t - \Delta t)\right]$$

Ecuación 13. Error TD aproximando la derivada

Esta forma de ver el error TD es equivalente al error TD discreto. Utilizando la esta nueva representación se puede encontrar que el gradiente del cuadrado del mismo con respecto al parámetro w_i es:

$$\frac{\partial E(t)}{\partial w_i} = \delta(t)\frac{1}{\Delta t}\left[(1 - \frac{\Delta t}{\tau})\frac{\partial V(x(t); w)}{\partial w_i} - \frac{\partial V(x(t - \Delta t); w)}{\partial w_i}\right]$$

Ecuación 14. Gradiente del error al cuadrado utilizando TD aproximado.

Sin embargo, una manera alternativa es sintonizar únicamente V(t-Δt) sin cambiar de forma explícita V(t), de donde podemos plantear el gradiente descendente como:

$$w_i = w_{i-1} + \eta\delta(t)\frac{1}{\Delta t}\left[\frac{\partial V(x(t-\Delta t);w)}{\partial w_i}\right]$$

Ecuación 15. Ajuste de parámetros a partir del gradiente descendente.

El gradiente descendente se pude utilizar para sintonizar todos los parámetros de la función aproximadora, pero como se esta empleando un sistema de inferencia TKS, tomar algoritmos híbridos de sintonización, aumentaría la velocidad de convergencia y aprovecharía el hecho de tener funciones lineales como consecuentes, por tal razón el método de gradiente descendente será utilizado únicamente con los parámetros no lineales del sistema mientras que los parámetros lineales serán sintonizados utilizando una variación del método de mínimos cuadrados recursivos, debido a que no se conoce el valor real de la función a la que se esta aproximando, sino una medida del error.

Se puede plantear de manera general el problema de mínimos cuadrados de la siguiente forma [3]:

$$V(x,\mu,A)=\sum_{j=1}^{total}\frac{\mu_j\left(A_{0j}+A_{j1}x_1+...+A_{nj}x_n\right)}{\sum_{i=1}^{totla}\mu_i}$$

$$V(x,\mu,A)=\frac{\mu_1}{\sum_{i=1}^{totla}\mu_i}A_{01}+A_{11}\frac{\mu_1 x_1}{\sum_{i=1}^{totla}\mu_i}+...+A_{n1}\frac{\mu_1 x_n}{\sum_{i=1}^{totla}\mu_i}+...+\frac{\mu_h}{\sum_{i=1}^{totla}\mu_i}A_{0h}+A_{1h}\frac{\mu_h x_1}{\sum_{i=1}^{totla}\mu_i}+...+A_{nh}\frac{\mu_h x_n}{\sum_{i=1}^{totla}\mu_i}$$

$$V(x)=\theta_1 g_1(x,u)+\theta_2 g_2(x,u)+...+\theta_n g_n(x,u)$$

Ecuación 16. Planteamiento de mínimos cuadrados recursivos

Al tomar diferentes puntos sobre el espacio de estado, se puede construir la matriz de diseño.

$$A\theta=Y$$

Ecuación 17. Matriz de diseño

Debido a que se va a tener más puntos de prueba que coeficientes para hallar, se tendrán más datos que incógnitas, por lo tanto encontrar una solución que satisfaga todas estas ecuaciones no es posible, debido a esto se asume un error, ruido aleatorio para modelarlo.

$$Y=e+A\theta$$

Ecuación 18. Modelo de aproximación

En lugar de encontrar una solución exacta, se desea encontrar una que minimice la suma del error al cuadrado definida como:

$$E(\theta) = \sum_{i=1}^{m} \left(y_i - a_i^t \theta\right)^2 = e^t e = \left(Y - A\theta\right)^t \left(Y - A\theta\right)$$

Ecuación 19. Error

Para llegar al valor que minimice esta ecuación se deriva el error obteniendo:

$$\frac{\partial E(\theta)}{\partial \theta} = 2A^t A\theta - 2A^t Y$$

Ecuación 20. Derivada del error

Haciendo la derivada igual a cero:

$$A^t A\theta = A^t Y$$
$$\theta = (A^t A)^{-1} A^t Y$$

Ecuación 21. Parámetros óptimos

Ahora bien, si se tiene acceso a nuevos pares de datos, en lugar de recalcular el factor θ, se puede obtener ventaja del valor calculado para obtener uno que tenga en cuenta la nueva información. Este problema es conocido en la literatura como mínimos cuadrados recursivos.

$$\theta_{k+1} = \left(\begin{bmatrix} A \\ a^t \end{bmatrix}^t \begin{bmatrix} A \\ a^t \end{bmatrix}\right)^{-1} \begin{bmatrix} A \\ a^t \end{bmatrix}^t \begin{bmatrix} Y \\ y_k \end{bmatrix}$$
$$P_k = \left(A^t A\right)^{-1}$$

$$P_{k+1} = \left(\begin{bmatrix} A \\ a^t \end{bmatrix}^t \begin{bmatrix} A \\ a^t \end{bmatrix} \right)^{-1}$$
$$P_{k+1} = (A^t A + aa^t)^{-1}$$
$$P_k^{-1} = P_{k+1}^{-1} - aa^t$$

$$\theta_k = P_k A^t Y$$
$$\theta_{k+1} = P_{k+1}(A^t Y + a y_k)$$

Ecuación 22. Parámetros futuros en función de los pasados

Reemplazando θ_{k+1} en términos de θ_k se tiene:

$$\theta_{k+1} = P_{k+1}(P_k^{-1}\theta_k + a y_k)$$
$$\theta_{k+1} = \theta_k + P_{k+1} a (y_k - a^t \theta_k)$$

Por lo tanto el parámetro puede ser representado en función de la nueva información y la estimación anterior. Calcular P_{k+1} es computacionalmente pesado debido a que se tiene que realizar la inversión de una matriz, aun así se puede llegar a una relación incremental para P_{k+1}.

$$P_{k+1} = P_k - P_k a (I + a^t P_k a)^{-1} a^t P_k$$
$$P_{k+1} = P_k - \frac{P_k a a^t P_k}{1 + a^t P_k a}$$

Ecuación 23. Relación de mínimos cuadrados recursivos

Ahora bien, como no conocemos el valor real de la función de valor, sino que se conoce una estimación del error, el algoritmo de mínimos cuadrados recursivos se debe ajustar a esta situación. Esto se hace disminuyendo su velocidad de convergencia, haciendo converger el algoritmo no a un valor sino a una dirección.

$$P_{k+1} = P_k - \frac{P_k a a^t P_k}{1 + a^t P_k a}$$
$$\theta_{k+1} = \theta_k + P_{k+1} a \alpha (y_k - a^t \theta_k)$$
$$donde \qquad 0 < \alpha < 1$$

Ecuación 24. Ajuste para asegurar convergencia al aplicar mínimos cuadrados recursivos.

De esta forma se puede utilizar estos dos mecanismos simultáneamente para sintonizar los parámetros del sistema de inferencia.

El algoritmo que se utilizó para encontrar la función de valor y la política de acción fue:

Inicializar c, σ arbitrariamente, y los coeficientes A_{ij} en cero.
Inicializar P_0=kI
Repetir para cada episodio:
  Inicializar s.
  Repetir por cada paso del episodio:
    acc ← acción tomada por π en s (eg. e-greedy)
    tomar la acción acc, observar la recompensa r y el estado futuro s^f
    δ ← r+γV(s^f)-V(s)

$$P_{k+1} = P_k - \frac{P_k a a^t P_k}{1 + a^t P_k a}$$
$$\theta_{k+1} = \theta_k + P_{k+1} a \alpha (\delta)$$
$$donde \qquad 0 < \alpha < 1$$

    Repetir para todos los c_{ik} y σ_{ik}

$$c_{ik} = c_{ik-1} + \eta \delta(t) \left[\frac{\partial V(s)}{\partial c_{ik}} \right]$$
$$\sigma_{ik} = \sigma_{ik-1} + \eta \delta(t) \left[\frac{\partial V(s)}{\partial \sigma_{ik}} \right]$$

    Hasta terminar c_{ik} y σ_{ik}
    s ← s^f
  Hasta s terminal

3.2. EXPERIMENTO DE NAVEGACIÓN

Como primer intento para desarrollar un algoritmo que utilice el principio del AR y aproxime la función de valor con un sistema de inferencia TKS, se planteo un problema de navegación altamente conocido, que fue solucionado con los algoritmos de AR; el cual busca que un agente llegue en el menor tiempo posible a cualquiera de dos esquinas opuestas. Para esto es necesario definir todos los aspectos del aprendizaje por refuerzo sobre los cuales el agente construirá el conocimiento.

Como primera medida la señal de estado debe cumplir con la condición de Markov, le debe permitir al agente saber su estado y prever cual será su estado siguiente si ejecuta una acción, en este problema es muy claro que la señal de estado no va más allá que la posición del agente en la cuadrícula. Las acciones posibles estarán limitadas a los movimientos que lleven al agente hacia el norte, sur, oriente u occidente.

Otro aspecto muy importante es la recompensa, pues del planteamiento de esta dependen aspectos tan importantes como la velocidad de convergencia, debe representar claramente la funcionalidad que el agente tiene que desarrollar, por este motivo la mejor recompensa que se adaptó al problema fue cero en todos los estados y uno en las metas.

En esta situación podemos conocer fácilmente el estado futuro a partir de la acción, se puede trabajar con la función de valor y obtener de ésta la política de acción. El sistema de inferencia TKS que aproximará la función de valor, debe tener dos variables lingüísticas, las posiciones en *X* y en *Y* del agente sobre la cuadrícula (la señal de estado).

3.2.1. Resultados

La política de acción utilizada para entrenar al agente fue e-greedy, en la cual la selección de la acción golosa era elegida con una probabilidad de 0.9. El parámetro paso del gradiente descendente, al igual que el parámetro ajuste de los mínimos cuadrados recursivos se ajustó en 0.01.

Después de implementar los parámetros mencionados con anterioridad, se encontró que el agente aprendió a alcanzar las metas fijadas, utilizando una cuadrícula de 3 por 3, éste lograba llegar a una generalización para un especio continuo. El algoritmo planteado en el numeral anterior, permitió llegar a una solución aceptable en no más de cien iteraciones. Para probar la capacidad de generalización que ofrece la función TKS, se añadió un requerimiento de navegación al problema; el agente debía evitar situarse en un estado donde se localizaba un muro, después de esta prueba se encontró que el agente conseguía aprender correctamente, los estados en los que podía circular y cumplir la meta principal de la mejor forma.

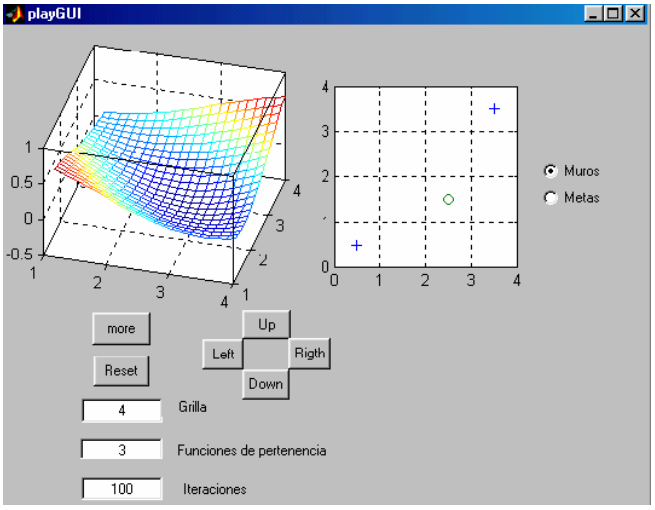


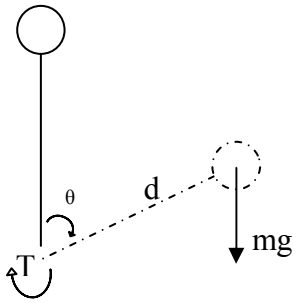
Ilustración 8. Interfase gráfica de programa playGUI.m.

Con el deseo de representar este ejemplo, se generó un programa con interfase gráfica sobre MATLAB de nombre playGUI.m, en el cual se puede definir el tamaño de la cuadrícula; la posición de las metas; la ubicación de un obstáculo (muro); la cantidad de funciones de pertenencia en cada lado del espacio; y también se puede variar el número de iteraciones que debe correr el algoritmo para generar la función de valor. Gracias a esto se encontró el potencial de utilizar la mezcla de sistemas de inferencia difusa y los métodos de aprendizaje por refuerzo, aprovechando los algoritmos híbridos de esta primera corriente de inteligencia artificial.

3.3. EXPERIMENTO DEL PÉNDULO

Otro problema que se utilizó para probar el desempeño del algoritmo, fue levantar un péndulo de la posición vertical inferior hasta la posición vertical superior, con un torque menor al mayor torque que puede hacer el péndulo por efecto de su peso. Por lo tanto este problema no es trivial. Esto se resolvió utilizando una aproximación de la función de valor estado acción.

El sistema se planteo de la siguiente manera:



$$\sum M = I\ddot{\theta}$$

$$T + mgd\sin(\theta) - \mu\dot{\theta} = md^2\ddot{\theta}$$

Ilustración 9. Modelo del péndulo.

De las ecuaciones que representan la dinámica del sistema se pude ver que el mismo esta determinado por dos variables: la posición y la velocidad, además del momento aplicado; con estas se puede determinar que pasará con el péndulo. Por lo que se eligieron como variables de estado, ya que cumplen con la propiedad de Markov.

El agente recibe siempre menos uno hasta que llega a la meta (posición superior), en donde la recompensa es de uno, además para ayudarle a seleccionar la mayor cantidad de acciones correctas se utiliza una política golosa, ya que para que

cumpla con la tarea debe efectuar una gran cantidad de acciones correctas. La meta se definió como un intervalo de ángulos y velocidades angulares posibles. Este rango esta determinado por el número de funciones de pertenencia, cuando utilizamos el sistema de inferencia como predictor de la función de valor o por el tamaño de la rejilla, cuando se discretiza el espacio de estado. Si estos dos intervalos no son lo suficientemente pequeños o si la función de aproximación no logra especificar correctamente estos espacios, el problema no podrá ser solucionado de la forma deseada.

Para entrenar el agente se utilizó el algoritmo descrito anteriormente, sin embargo fue más sencillo utilizar la función de valor estado acción ya que no era claro la transición de los estados debido a la acción, utilizando una política de control bang-of-bang en la que se podía elegir solo tres posibles acciones: torque positivo, negativo o cero.

Se dividió cada una de las dos variables de estado con 15 funciones de pertenencia y a cada regla se le asigno un plano, por lo tanto el sistema debe utilizar el proceso de mínimos cuadrados recursivos para sintonizar todos los parámetros de los planos que termina siendo 15X15X3X3 (2025) por lo cual se necesita una matriz de 2025X2025. Este fue un inconveniente que impidió la prueba del algoritmo híbrido de sintonización del ejemplo. Por este motivo solo se utilizó el método de gradiente descendente para sintonizar los parámetros de los planos y se agrego tasas de elegibilidad para disminuir el tiempo de convergencia del algoritmo.

Empleando estas características de entrenamiento y de configuración del sistema de inferencia difuso, se llega a encontrar una función de valor mucho mas precisa, que permite controlar el péndulo de tal manera que este logra alcanzar su meta. Por otro lado, si se discretiza el espacio, dividiéndolo uniformemente (cada variable en treinta partes), y utilizando las mismas características de aprendizaje, como lo son la política y la recompensa, bajo el algoritmo de entrenamiento de aprendizaje por refuerzo SARSA, se logra encontrar una solución bastante cercana a la que se obtiene con la expresión continua, aunque esta no alcanza a controlar el movimiento del péndulo como se desea desde un principio. Con esta solución, el péndulo no logra mantenerse en equilibrio en el la parte superior, por lo contrario oscila indefinidamente, debido a que el espacio no es lo suficientemente pequeño.

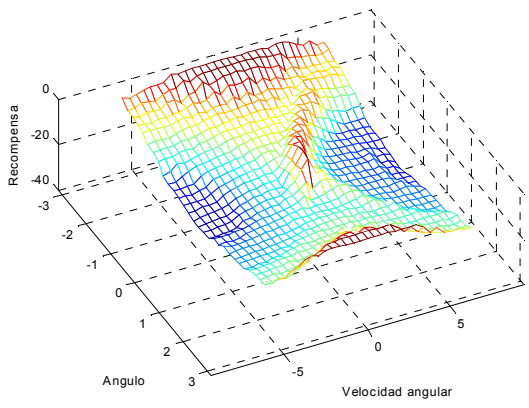
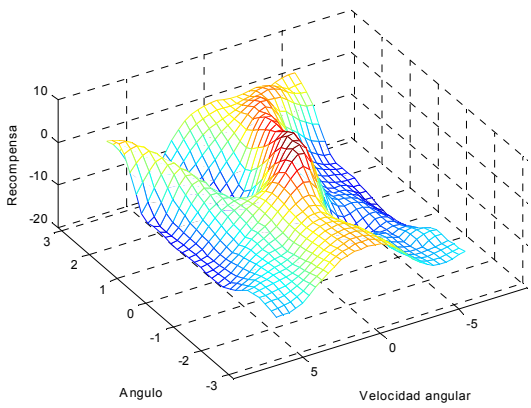
Función de recompensa, rejilla de 30X30	Función de recompensa, 15X15 funciones de pertenencia y una función
	

Ilustración 10. Comparación de funciones de valor discretas y continuas

Aprovechando el planteamiento del algoritmo como función de valor estado acción, se colocaron funciones de pertenencia sobre todo el intervalo de torque, esto aumento el numero de acciones posibles que el agente podía efectuar, por lo tanto como era de esperarse, se encontró que el aprendizaje es mucho mas pesado, aunque el agente logra generar una función de estado acción de la forma esperada, el control del péndulo es menos preciso que el que se efectúa después de entrenar el sistema utilizando únicamente tres torques, por lo cual, el agente llega a la meta establecida, en un mayor tiempo. Esto ocurre gracias a que la política de acción es golosa, debido a esto, el agente no continua explorando sobre todo el espacio de acción sino por lo contrario explota el conocimiento adquirido en cada uno de los episodios.

La velocidad de convergencia de cada una de estas aproximaciones de la función de valor estado acción, no se diferencian en gran medida, ya que por un número muy similar de iteraciones se puede construir cualquiera de ellas. No es igual si se evalúan con respecto al proceso computacional ya que cuando se sintoniza el sistema para generar una función de valor en estado continuo o una función de valor estado-acción continuo, la cantidad de procesos necesarios para sintonizar los parámetros se eleva, por lo que el sistema puede demorase casi 250 veces mas.

Por otra lado los parámetros de sintonización como lo es el parámetros paso en el algoritmo SARSA o el paso del gradiente no son iguales, es más, cuando se utilizan mas acciones que las utilizadas con la política de acción bang-of-bang el

paso del gradiente es afectado para que el algoritmo logre realizar un mejor aprendizaje en un tiempo relativamente corto. La sintonización de este parámetro se hace eurísticamente, debido a esto, el conocimiento del sistema es indispensable para lograr entrenarlo de la mejor forma posible.

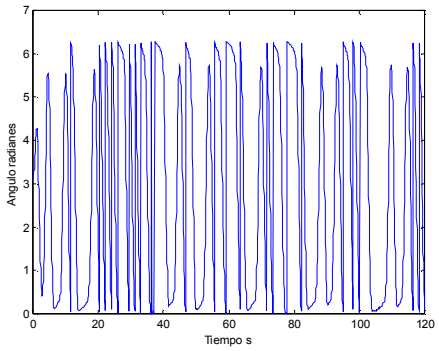
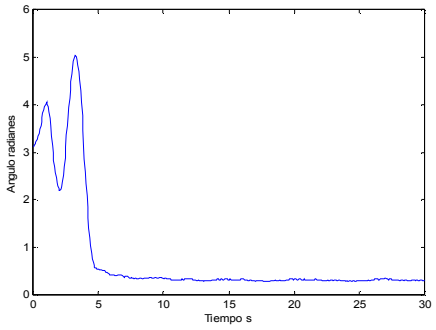
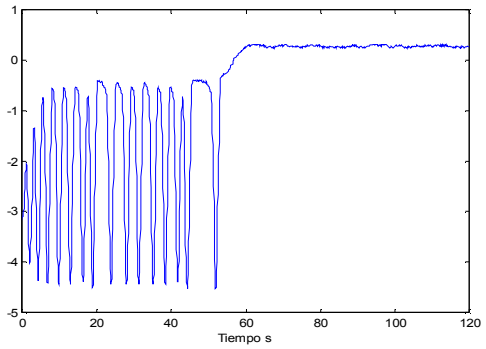
Esta gráfica muestra como el péndulo cambia su posición angular con respecto al tiempo, aunque pasa por la posición de equilibrio en la parte superior no logra estabilizarse en este punto.	
Esta gráfica muestra la estabilización del péndulo cerca al punto superior deseado, en un tiempo muy corto, utilizando bang-of- bang con estados continuos.	
Esta gráfica muestra la estabilización del péndulo utilizando una selección continua de acción y estados continuos.	

Ilustración 11. Comportamiento del péndulo para las respuestas encontradas.

Los parámetros de paso son una pieza fundamental en el proceso de aprendizaje, ya que controlan dos variables muy importantes en el proceso de entrenamiento del agente, la primera variable es el tiempo, medido como la cantidad de ensayos o pruebas que se tiene que hacer para poder tomar la información necesaria para estimar de forma adecuada la función de valor, y la segunda, es la calidad de la aproximación de la función de valor. Esto significa que entre mas pequeño es el paso, se necesita un mayor número de experimentos para el proceso de entrenamiento, pero la función de valor se aproxima mejor.

4. CONCLUSIONES Y TRABAJO FUTURO

A diferencia de las aproximaciones discretas de la función de valor cuando se utiliza un sistema TKS como función de aproximación, esta no ajusta únicamente el estado por el cual pasa el agente, sino que también ajusta los estados vecinos; a la larga este hecho representa que todos los estados tienen tanto un valor de recompensa esperado como una acción asociada.

La definición de la política, se puede hacer de igual forma que en los algoritmos discretos TD o SARSA, con la ventaja que los sistemas TKS permiten tomar la acción como otra variable lingüística, por lo cual se puede construir un espacio de decisión con una acción continua.

Aunque estas dos cualidades del sistema son muy atractivas, no hay que olvidar que para que la aproximación logre desarrollar un desenvolvimiento eficiente del agente, se debe hacer un entrenamiento que permita recorrer todas los estados posibles un gran número de veces, es por esta razón que cuando se empieza a trabajar con estos predictores continuos, el número de ejemplos o de eventos a ser ejecutados por el agente aumenta, además la política de acción tiene que tener en cuenta este gran número de posibles selecciones, por lo tanto utilizar una política poco explorativa puede reducir el potencial del sistema TKS.

Una de las motivaciones mas fuertes que impulsaron el desarrollo de esta investigación fue la posibilidad de utilizar los métodos híbridos de sintonización de los sistemas TKS, desafortunadamente se encontró que las ventajas en tiempo y en calidad de aproximación, se pagan con los requerimientos computacionales, por ejemplo, en problemas que están definidos por varias variables de estado y que son altamente no lineales, como se vio en el problema del péndulo, el tamaño de las matrices necesarias para sintonizar los parámetros de las reglas del sistema, utilizando mínimos cuadrados recursivos, son tan grandes que un ordenador convencional no puede trabajar con ellas, debido a esto el gradiente descendente es la herramienta seleccionada para la sintonización de parámetros.

Las pruebas realizadas con los ejemplos, mostraron que no todas las variables son igual de sensibles a la hora de hacer la sintonización. Los parámetros que determinan las funciones de pertenencia de los valores lingüísticos, revelaron ser muy sensibles al gradiente descendente, estos parámetros debían ser sintonizados con un valor de paso mucho menor al utilizado para ajustar los coeficientes del plano asignado a cada una de las reglas.

Aunque obtener una representación continua es muy ventajoso, esto incrementa en gran manera los algoritmos de aprendizaje, pero el resultado final es mucho mejor y la información no es considerablemente más grande.

Uno de los pasos siguientes que se puede dar en esta investigación, es aumentar la velocidad de aprendizaje del agente dividiendo el aprendizaje en dos etapas:

una en el que el agente aprendiera a identificar cualquier estado futuro a partir de una acción realizada y la otra en el que se aprendiera la deseabilidad de cada estado.

Si lo anterior se realiza, se aprovecharía cada exploración del agente en el medio y se almacenaría la información de cada episodio. Se podría así, variar la meta del agente sin necesidad de volver a recorrer todos los eventos.

La exploración del agente marca de manera significativa la forma de aprendizaje, por esto, probar diferentes políticas de acción (continua), enriquecería la concepción de como este aprendizaje debe ser realizado, por esto, la política es otro aspecto que se debe evaluar en un trabajo futuro (variar la política e-greedy o simplemente golosa, por cualquiera de las otras que se mencionaron anteriormente).

La aplicación de esta metodología (gradiente descendente y mínimos cuadrados recursivos) para la solución de problemas altamente no lineales con múltiples variables de estado, no puede ser tratado con facilidad utilizando estados continuos debido a la gran memoria que necesita el algoritmo, sin embargo, el planteamiento discreto no necesita la misma cantidad de memoria en el algoritmo, esto se paga en la resolución del control que aprende el agente, un paso intermedio entre estos dos es utilizar únicamente gradiente descendente, este mecanismo permite utilizar estados continuos y no aumentar demasiado la memoria necesaria del algoritmo. El aspecto de memoria es un aspecto limitante

en las posibilidades de esta metodología ya sea utilizando estados continuos o discretos.

5. REFERENCIAS

[1] Sutton, Richard S., Barto, Andrew G.. Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, 1998.

[2] Doya, Kenji. Reinforcement Learning In Continuous Time and Space. Neural Computation, vol. 12. pp. 219-245.

[3] Jang, J.-S.R., Sun, C.T., Mizutani E. Neuro-Fuzzy and Soft Computing. *Regression and Optimization*, pages 95-126, *Neuro-Fuzzy Modeling,* pages 333-363. Prentice-Hall, Inc. 1997

[4] Peña R., Carlos Andrés. Coevolutionary Fuzzy Modeling. Lausanne, EPFL, 2002.

[5] Gu, Dongbing; Hu Housheng. Reinforcement Learning of Fuzzy Controllers for Quadruped Walking Robots, 15th Trienal World Congress, Barcelona, Spain 2002.