

Un método lingüístico para comparar de sistemas dinámicos

Juan Moreno-Garcia

Dept.de Informática

E.U.I. Técnica Industrial de Toledo

Universidad de Castilla-LaMancha

Juan.Moreno@uclm.es

Ramon Manjavacas

Dept.de Informática

E.S. Informática de Ciudad Real

Universidad de Castilla-LaMancha

Ramon.Manjavacas@uclm.es

Carlos González

Dept.de Informática

E.S. Informática de Ciudad Real

Universidad de Castilla-LaMancha

Carlos.Gonzalez@uclm.es

Resumen

Este trabajo presenta un nuevo método para comparar la evolución temporal de dos sistemas dinámicos. Para ello hace uso de los modelos difusos temporales. Nuestro método genera primeramente los modelos difusos temporales de la evolución temporal de dos sistemas dinámicos, es decir, se obtienen dos modelos. Finalmente, compara los modelos obtenidos representativos de los sistemas dinámicos, obteniendo como salida, de forma lingüística, el "parecido" de estos modelos.

1. Introducción

Un sistema dinámico (SD) puede ser definido como un sistema que cambia en el tiempo. En un SD existen un conjunto de relaciones entre las variables de entrada y salida que representan las modificaciones de las variables de salida cuando las variables de entrada son modificadas. En los SDs, también se necesita conocer la evolución temporal de estas variables porque los valores de las variables en un tiempo dependen de los valores de las variables en los instantes anteriores. Los SDs con magnitudes físicas continuas verifican que la evolución de las variables es continua en el tiempo, esto es, en el instante $t + 1$ el valor de la variable v_{t+1} es muy parecido al valor de la variable v_t en el instante t . Esta hipótesis se supone cuando se definen los modelos difusos temporales (MDTs) [6].

Los SDs se modelan utilizando diferentes técnicas tradicionales [9, 5]. Se modelan pa-

ra explicar y predecir su comportamiento, y para analizar sus propiedades dinámicas. Se pueden modelar utilizando de la lógica difusa [8, 1, 2, 3]. Los MDTs se utilizan para representar la evolución del SD. Esta estructura simula el comportamiento de un sistema en función de la entrada. Informalmente, los MDTs son una secuencia ordenada de reglas difusas, donde cada regla representa una zona temporal.

Una diferencia entre nuestro método y otros es que nosotros modelamos los SDs mediante sistemas ordenados de reglas, de esta forma, cada regla representa instantes consecutivos con la misma etiqueta de salida. Otra diferencia es que utilizamos etiquetas lingüísticas [12] a lo largo de todo el proceso de inducción, así, los modelos obtenidos son más descriptivos. La última diferencia es que nuestro método agrupa utilizando la variable de salida tomando los ejemplos consecutivos que tienen la misma salida.

En este trabajo utilizaremos los modelos temporales para comparar el "dinamismo" de un SD con otro SD. Para ello, se generará el MDT de la evolución temporal de cada SD, y posteriormente, se compararán dichos modelos. Por ello, el objetivo de este trabajo es presentar un método para comparar MDTs, es decir, podremos comparar la evolución temporal de dos SDs. Para que la comparación obtenida sea lo más comprensible posible se utilizarán etiquetas lingüísticas para mostrar lo parecido que es un MDT a otro en una serie consecutivas de instantes.

En la sección 2 se presenta una introducción a los MDTs. Después, se presenta el mé-

todo propuesto. En la sección 4 se muestra un ejemplo de cómo funcionan los algoritmos, y se finaliza mostrando las conclusiones.

2. Introducción a los MDTs

Se mostrará la estructura de nuestro método de modelado, los MDTs. En las subsecciones 2.1 y 2.2 se ofrece una descripción detallada de las dos entradas del algoritmo. La sección 2.3 define los MDTs.

2.1. CONJUNTO DE EJEMPLOS E

Una serie de tiempo es una secuencia de datos que es ordenada en el tiempo [10]. Un SD se puede representar por medio de una serie temporal. Cada ejemplo de la serie contiene los valores de las variables en un instante. Esta sección muestra formalmente la estructura de la serie de tiempo que se representa mediante un conjunto de ejemplos E .

Cada ejemplo de E tiene un conjunto de m variables reales de entrada definidas en el dominio $X = X_1 \times \dots \times X_m \subset R^m$, y una variable real de salida $S \subset R$. $E = \{e_1 \dots e_n\}$ siendo $e_i = (x_1^i \dots x_m^i, s^i, t_i)$, donde $x_j^i \in X_j$, $s^i \in S$ y t_i es el tiempo en que ocurre e_i .

2.2. CTOS ORDENADOS DE ETIQUETAS

Los MDTs trabajan con variables lingüísticas que pueden tomar valores de un conjunto ordenado de etiquetas lingüísticas con algunas restricciones sobre su dominio. Estas variables se denominan *variables lingüísticas continuas* (desde ahora variables). Las etiquetas lingüísticas (desde ahora etiquetas) de las variables se definen antes de obtener el MDT.

Se define un conjunto ordenado de etiquetas para cada variable de entrada X_j . Su estructura es $SA_j = \{SA_j^1 \dots SA_j^{i_j}\}$. Para la variable de salida S se define un conjunto ordenado de etiquetas $SC = \{SC^1 \dots SC^{i_y}\}$. En estos conjuntos el superíndice i es la posición de la etiqueta, $i_j = |SA_j|$ e $i_y = |SC|$.

El orden de las etiquetas en los conjuntos de etiquetas se basa en el método de defuzzificación llamado *media de máximos* (MOM) [4]. El

MOM de una etiqueta SA_j^a lo representamos como $MOM(SA_j^a)$. Una etiqueta SA_j^a es menor que SA_j^b si ambas están definidas para X_j y el MOM de SA_j^a es menor que el MOM de SA_j^b . De igual forma se puede establecer cuando una etiqueta es menor o igual que otra.

X_j es una variable lingüística continua si tiene asociada una semántica a cada SA_j^i definida sobre ella que verifica: (1) Cada etiqueta SA_j^i se define para un dominio ordenado; (2) Las etiquetas definidas sobre X_j están ordenadas mediante el MOM; (3) La suma del grado de pertenencia de un valor x a todas las etiquetas definidas en SA_j debe ser 1 ($\sum_{SA_j^i \in SA_j} \mu_{SA_j^i}(x) = 1$); (4) Cada x_j en el dominio de X_j tiene 1 o 2 etiquetas con un grado de pertenencia mayor que 0. Así, una de estas expresiones se verifica: ($\mu_{SA_j^i}(x_j) = 1$) ó ($\mu_{SA_j^i}(x_j) = a$ y $\mu_{SA_j^{i+1}}(x_j) = 1-a$). Si el grado de pertenencia de x_j a SA_j^i se incrementa o decrementa entonces el grado de pertenencia de x_j a SA_j^{i+1} se decrementará o incrementará en el mismo grado.

2.3. MODELOS DIFUSOS TEMPORALES

Los SDs estudiados tienen una variable de salida. Nuestro método de inducción agrupa utilizando la variable de salida, así, todos los ejemplos consecutivos con la misma etiqueta de salida (la etiqueta con más grado de pertenencia al valor de salida s^i del ejemplos e_i) se representan con la misma regla.

Un MDT está compuesto por **Reglas Difusas Temporales** (*RDTs*) que tienen dos componentes: un antecedente y una etiqueta de salida. El antecedente es una conjunción de disyunciones de expresiones X_j is SA_j^i . El antecedente tiene asociado un conjunto de ejemplos consecutivos $E_{p,u}$. Un **conjunto de ejemplos consecutivos** $E_{p,u}$ es un conjunto de ejemplos que verifica: (1) $p \leq u$ y $E_{p,u} \subset E$; (2) $\forall k \in [p, u], \exists e_k \in E_{p,u}$; (3) $\forall e_a, e_b \in E_{p,u}, a \neq b$, donde E es el conjunto de ejemplos; p, u, k, a y b son los índices a los ejemplos de E .

Se utilizan los intervalos lingüísticos para representar las disyunciones, y el conjunto de m intervalos lingüísticos para representar las con-

junciones de disyunciones de una RDT. Un **intervalo lingüístico de longitud** c es una secuencia de c etiquetas consecutivas definidas en SA_j que comienzan en la etiqueta p . Se representa como $LI_{j,p}^c = \{SA_j^p \dots SA_j^{p+(c-1)}\}$, donde $SA_j = \{SA_j^1 \dots SA_j^j\}$ es un conjunto ordenado de j etiquetas en X_j , p es la posición en SA_j de la primera etiqueta del intervalo lingüístico y c es el número de etiquetas del intervalo. Por ejemplo, el intervalo $LI_{j,3}^4$ representa la disyunción $((x_j \text{ is } FN) \text{ OR } (x_j \text{ is } NR) \text{ OR } (x_j \text{ is } FP) \text{ OR } (x_j \text{ is } P))$ si se utiliza el conjunto de etiquetas $ETIS = \{VN, N, FN, NR, FP, P, VP\}$.

Un **conjunto ordenado de m intervalos** SLI_m es una secuencia de m intervalos definida sobre m variables. Se representa como $SLI_m = \{LI_{1,p_1}^{c_1} \dots LI_{m,p_m}^{c_m}\}$ donde p_j es la posición en SA_j de la primera etiqueta del intervalo j y c_j es el número de etiquetas del intervalo j . Los SLI_m se utilizan para representar lingüísticamente el rango de valores de m variables de entrada. Por ejemplo, si para cada variable de entrada se utiliza un conjunto ordenado de 7 etiquetas representadas en la figura ??, entonces, $SLI_3 = \{LI_{1,5}^3, LI_{2,1}^2, LI_{3,2}^4\} = \{\{FP, P, VP\}, \{VN, N\}, \{N, FN, NR, FP\}\}$ representa la conjunción de disyunciones $((x_1 \text{ is } FP) \text{ OR } (x_1 \text{ is } P) \text{ OR } (x_1 \text{ is } VP)) \text{ AND } ((x_2 \text{ is } VN) \text{ OR } (x_2 \text{ is } FN)) \text{ AND } ((x_3 \text{ is } N) \text{ OR } (x_3 \text{ is } FN) \text{ OR } (x_3 \text{ is } NR) \text{ OR } (x_3 \text{ is } FP))$.

Así, una **regla difusa temporal** $R_{p,u}$ es una regla difusa que verifica: (1) Su antecedente es un SLI_m ; (2) Su consecuente es una etiqueta $SC^w \in SC$; (3) Su SLI_m tiene asociado un conjunto de ejemplos consecutivos $E_{p,u}$. $E_{p,u}$ indica los instantes que una RDT representa. Luego una RDT $R_{p,u}$ es una regla difusa que representa los instantes del p al u en un SD. Se pueden comparar dos RDTs, así, $R_{k,l}$ está incluida en $R_{i,j}$ si el índice i es menor o igual que el índice k y el índice j es mayor o igual que l . Una regla $R_{i,j}$ es menor que $R_{k,l}$ si $R_{k,l}$ no está incluida en $R_{i,j}$ y $i < k$. De la misma forma se define cuando una RDT es igual a otra RDT, o cuando una RDT es mayor que otra. Utilizando estos conceptos se pueden ordenar las RDTs. Como puede verse, el orden

del cada regla viene dado por el conjunto de ejemplos asociado a cada SLI_m .

Para finalizar esta sección, el algoritmo 1 ofrece el método de inferencia de los MDTs. Para comprender su comportamiento se estudiará la función de pertenencia de $LI_{j,p}^c$. La ecuación 1 define la función de pertenencia de un $LI_{j,p}^c$.

$$\mu_{LI_{j,p}^c}(a_j) = \sum_{SA_j^z \in LI_{j,p}^c} \mu_{SA_j^z}(a_j) \quad (1)$$

donde a_j es un valor discreto definido en el dominio de la variable continua X_j , $z \in [p..c-1]$.

La ecuación 2 define la función de pertenencia de un SLI_m .

$$\mu_{SLI_m}(e_i) = *(\mu_{LI_{j,p_j}^{c_j}}(x_j)) \quad (2)$$

donde e_i es un ejemplo del conjunto E , $j \in [1..m]$ y $*$ es una t-norma.

Algoritmo 1 Método de Inferencia

```

j ← 1
for i = 1 to |E| do
  if μSLImj(ei) > μSLImj+1(ei) then
    s ← Cons(Rj)
  else
    s ← Cons(Rj+1)
    j ← j + 1
  end if
end for

```

El algoritmo de inferencia necesita como entrada un conjunto de ejemplos. El ejemplo y regla actual se indican mediante los índices i y j respectivamente. El bucle *for* se utiliza para recorrer el conjunto de ejemplos E . Si el grado de pertenencia de e_i al SLI_m de R_j (SLI_m^j) es mayor que el grado de pertenencia de e_i al SLI_m de R_{j+1} (SLI_m^{j+1}) la salida es la etiqueta de salida de R_j ($s \leftarrow \text{Cons}(R_j)$), en otro caso, la salida es la etiqueta de salida de R_{j+1} ($s \leftarrow \text{Cons}(R_{j+1})$) y la nueva regla actual es R_{j+1} ($j \leftarrow j + 1$). Se necesita un proceso de defuzzification [4, 11] para obtener un valor discreto de salida.

3. Método propuesto

El método propuesto consiste en la comparación de la evolución temporal mediante la creación y la posterior comparación de MDTs. En [7] se muestra cómo inducir un MDTs a partir de un serie temporal. Una vez obtenidos los modelos, se compararán mediante la propuesta realizada en este trabajo.

Algoritmo 2 Comparación de MDTs

```

MDT1 ← InducirMDT(E1, CtoEti)
MDT2 ← InducirMDT(E2, CtoEti)
T1 ← ObtenerTendencias(MDT1)
T2 ← ObtenerTendencias(MDT2)
[Tra1, Tra2] ← ObtenerTramos(T1, T2)
L ← CompararTramos(Tra1, Tra2)

```

El algoritmo 2 muestra el método propuesto para comparar MDTs. Las dos primeras sentencias obtienen los MDTs aplicando el algoritmo propuesto en [7]. Una vez obtenidos los MDTs MDT_1 y MDT_2 , se procede a la comparación de modelos. Para ello, se utilizarán las líneas del algoritmo de la 3 a la 6.

Las siguientes subsecciones explican cómo obtener tendencias (sección 3.1), cómo obtener los tramos de cada tendencia (sección 3.2) y cómo comparar los tramos de cada tendencia (sección 3.3).

3.1. Obtener tendencias

Las líneas 3 y 4 del algoritmo 2 obtienen lo que denominamos *tendencias*. Una tendencia es un conjunto de reglas consecutivas de un MDT que representan una tendencia común respecto a la variable de salida. Esta tendencia será un incremento o un decremento. Así, las tendencias anteriores y posteriores a una tendencia marcarán un "sentido" diferente a dicha tendencia, es decir, si una tendencia marca un incremento, la tendencia anterior y posterior marcarán un decremento. Por ejemplo, si las etiqueta de salida de la reglas de un MDT son las siguientes:

[1,2,4,3,2,4,5,6], donde cada número n representa la etiqueta del conjunto

ordenado de etiquetas en la posición n .

Las tendencias obtenidas serán las siguientes:

[[1,2,4],[3,2],[5,6]], donde la primera tendencia, que marca un incremento, será [1,2,4]; la segunda sería [3,2], luego marca un decremento; y finalmente [5,6], que será la tendencia tercera y marca un incremento.

Algoritmo 3 Obtención de tendencias

```

j ← 0
T ← θ
Tend ← θ
if Cons(R1) = Cons(R2) then
    sentido ← '+'
else
    sentido ← '-'
end if
for i = 1 to |MDT1| do
    if sentido = '+' then
        if Cons(R1) < Cons(R2) then
            Tend ← Tend + Ri
        else
            sentido ← '-'
            T ← T + Tend
            Tend ← θ
        end if
    end if
    if sentido = '-' then
        if Cons(R1) > Cons(R2) then
            Tend ← Tend + Ri
        else
            sentido ← '+'
            T ← T + Tend
            Tend ← θ
        end if
    end if
end for

```

El algoritmo 3 obtiene las tendencias de un MDT. A continuación se explicará su funcionamiento brevemente. Inicialmente el conjunto de tendencias está vacío ($T \leftarrow \theta$) y la tendencia que va a ser calculada se inicializa a vacío ($Tend \leftarrow \theta$). Posteriormente se detecta si la primera tendencia marca un incremento o un decremento, para ello, se utiliza el *if*

que está fuera del bucle. Este condicional utiliza la variable *sentido* para indicar si la tendencia marca incremento (*sentido* \leftarrow '+') o decremento (*sentido* \leftarrow '-'). Posteriormente se recorre el modelo (*MDT*₁) comparando las etiquetas de salida de reglas consecutivas. Si la tendencia es positiva, se van agrupando las reglas consecutivas que la etiqueta de salida de la siguiente reglas sea mayor que la etiqueta de salida de la regla anterior (*if* $Cons(R_1) < Cons(R_2)$). Esto se realiza hasta que ocurre un cambio y se realiza el *else*, que cambia el sentido de la tendencia (que será la siguiente) mediante la sentencia *sentido* \leftarrow '-'. También se añade la sentencia calculada a *T* y se inicializa a vacío (*Tend* \leftarrow θ) la tendencia para calcular la siguiente. Cuando la tendencia es negativa el proceso es análogo y se utiliza el segundo *if* del bucle *for*. Como puede verse, es muy similar al del caso de cálculo de tendencias positivas.

3.2. Obtener los tramos de tendencias

A continuación, se obtienen lo que hemos denominado "tramos" (línea 5 del algoritmo 2). Un tramo es un conjunto de reglas consecutivas de una tendencia que van a ser comparados. Para la obtención de los tramos se va comparando, regla a regla, la etiqueta de salida de las tendencias *T*₁ y *T*₂, y se van agrupando.

Se mostrará un ejemplo utilizando dos MDTs con las siguientes etiquetas de salida: *Tend*₁=[[0, 1, 3, 4, 5, 6], [5, 3, 2, 1, 0], [1], [0]] y *Tend*₂=[[0, 2, 3, 5, 6], [5, 4, 3, 2, 1, 0], [1], [0]], donde *Tend*₁ y *Tend*₂ son las tendencias con las etiqueta de salidas de las reglas de las tendencias obtenidas de dos MDTs (sólo se ponen las etiquetas de salidas numeradas según su orden para simplificar).

Ahora se agrupa la tendencia *i* (*T*_{*i*}) de *Tend*₁ y *Tend*₂, es decir, se comparan las tendencias de la siguiente forma:

- *T*₁ =[0, 1, 3, 4, 5, 6] con *T*₁ =[0, 2, 3, 5, 6].
- *T*₂ =[5, 3, 2, 1, 0] con *T*₂ =[5, 4, 3, 2, 1, 0].
- *T*₃ =[1] con *T*₃ =[1].

- *T*₄ =[0] con *T*₄ =[0].

Para comparar cada una de ellas, se utiliza el algoritmo 4, obteniendo los tramos [[0], [1, 3], [4, 5], [6], [5], [3], [2], [1], [0], [1], [0]] y [[0], [2, 3], [5], [6], [5], [4, 3], [2], [1], [0], [1], [0]] para ambas tendencias:

[0 , [1, 3], [4, 5]] y [[0], [2, 3], [5]] para la primera tendencia.

[6 , [5], [3], [2], [1]] y [[5], [4, 3], [2], [1], [0]] para la segunda tendencia.

[1] y [[1]] para la tercera tendencia.

[0] y [[0]] para la cuarta tendencia.

Para aclarar el proceso vamos a detallar cómo se han realizado las agrupaciones para obtener los tramos mediante un ejemplo. Se aplicará el algoritmo a la primera tendencia de las anteriores. Inicialmente *Tra*₁, *Tra*₂, *A*₁ y *A*₂ se inicializan a θ ; y *j*₁ y *j*₂ a 1 (apuntan al primer elemento de la tendencia). A continuación se calcula el sentido de las primeras tendencias, para ello se compara el consecuente de la primera regla de *T*₁ y el consecuente de la segunda regla de *T*₁ (la etiqueta 0 con la etiqueta 1), luego el sentido es creciente ('+'). A continuación, se procede a crear las agrupaciones recorriendo *T*₁ mediante el índice *j*₁ y *T*₂ mediante el índice *j*₂. Este recorrido se realiza en el algoritmo mediante el bucle *while*. En la primera iteración, como el sentido es '+' se realiza el primer *if*. Se compara el consecuente de la primera regla de *T*₁ (*R*_{1,*j*₁}) con el consecuente de la primera regla de *T*₂ (*R*_{2,*j*₂}), es decir, 0 con 0, como son iguales se cumple el *if* $Cons(R_{1,j_1}) = Cons(R_{2,j_2})$ (*R*_{1,*j*₁} representa la regla *j*₁ de *T*₁), luego se realizan las acciones asociadas a él:

- *Tra*₁ \leftarrow *Tra*₁ + *R*_{1,*j*₁}: que añade a *Tra*₁ la primera regla de *T*₁.
- *A*₁ \leftarrow *A*₁ + *Tra*₁: como ya se a terminado el primer tramo, se añade el tramo calculado (*Tra*₁) al conjunto de tramos, luego *A*₁ = [0].
- *j*₁ \leftarrow *j*₁ + 1: se incrementa *j*₁ que pasa a apuntar a la segunda regla.

Algoritmo 4 Obtención de tramos en tendencias

```

Tra1 ← θ
A1 ← θ
j1 ← 1
Tra2 ← θ
A2 ← θ
j2 ← 1
if Cons(T1,1) < Cons(T1,2) then
  sentido ← '+'
else
  sentido ← '-'
end if
while j1 < |T1| and j2 < |T2| do
  if sentido = '+' then
    if Cons(R1,j1) = Cons(R2,j2) then
      Tra1 ← Tra1 + R1,j1
      A1 ← A1 + Tra1
      j1 ← j1 + 1
      Tra1 ← θ
      Tra2 ← Tra2 + R2,j2
      A2 ← A2 + Tra2
      j1 ← j1 + 1
      Tra2 ← θ
    else
      if Cons(R1,j1) < Cons(R2,j2) then
        Tra1 ← Tra1 + R1,j1
        j1 ← j1 + 1
      else
        Tra2 ← Tra2 + R2,j2
        j2 ← j2 + 1
      end if
    end if
  end if
  if sentido = '-' then
    if Cons(R1,j1) = Cons(R2,j2) then
      Tra1 ← Tra1 + R1,j1
      A1 ← A1 + Tra1
      j1 ← j1 + 1
      Tra2 ← Tra2 + R2,j2
      A2 ← A2 + Tra2
      j1 ← j1 + 1
    else
      if Cons(R1,j1) > Cons(R2,j2) then
        Tra1 ← Tra1 + R1,j1
        j1 ← j1 + 1
      else
        Tra2 ← Tra2 + R2,j2
        j2 ← j2 + 1
      end if
    end if
  end if
end while
if j1 < |T1| then
  Tra1 ← Tra1 + R1,j1
end if
if Tra1 ≠ θ then
  A1 ← A1 + Tra1
end if
if j2 < |T2| then
  Tra2 ← Tra2 + R2,j2
end if
if Tra2 ≠ θ then
  A2 ← A2 + Tra2
end if

```

- $Tra_1 \leftarrow \theta$: se asigna Tra_1 a θ para el cálculo del siguiente tramo.
- $Tra_2 \leftarrow Tra_2 + R_{2,j_2}$: que añade a Tra_2 la primera regla de T_2 .
- $A_2 \leftarrow A_2 + Tra_2$: se añade al conjunto de tramos el tramo calculado (Tra_2), luego $A_2 = [0]$.
- $j_2 \leftarrow j_2 + 1$: se incrementa j_2 que pasa a apuntar a la segunda regla.
- $Tra_2 \leftarrow \theta$: se asigna Tra_2 a θ para el cálculo del siguiente tramo.

A continuación se realiza la segunda iteración del bucle. Nuevamente se realiza la comparación de los consecuentes de las reglas apuntadas por los índices j_1 y j_2 , es decir, se compara la etiqueta 1 ($Cons(R_{1,j_1})$) con la etiqueta 2 ($Cons(R_{2,j_2})$). Como no son iguales se comprueba la sentencia condicional *if* $Cons(R_{1,j_1}) < Cons(R_{2,j_2})$, que se cumple, ya que $1 < 2$, luego se realizan las dos sentencias asociadas al *if*: Tra_1 toma el valor [1] ($Tra_1 \leftarrow Tra_1 + R_{1,j_1}$) y se incrementa j_1 ($j_1 \leftarrow j_1 + 1$) tomando el valor 3.

Se realiza la tercera iteración del bucle, comparando el consecuente de la tercera regla de T_1 con el consecuente de la segunda de regla de T_2 , es decir, se compara 3 con 2, luego se ejecutan las sentencias del *else* del segundo *if*. La primera sentencia ($Tra_2 \leftarrow Tra_2 + R_{2,j_2}$) hace $Tra_2 = [2]$ y la segunda hace que j_2 tome el valor 3.

A continuación se realiza la cuarta iteración del bucle. Se realiza la comparación de los consecuentes de las reglas apuntadas por los índices j_1 y j_2 , es decir, se compara la etiqueta 3 con la etiqueta 3. Como son iguales se completan los segundos tramos de ambas tendencias, pasando A_1 a valer $[[0], [1, 3]]$, A_2 a $[[0], [2, 3]]$, j_1 toma el valor 4 y j_2 el valor 4.

Posteriormente se realiza la iteración 5 del bucle, comparando el consecuente de la regla 4 de la primera tendencia con el consecuente de la regla 4 de la segunda tendencia, es decir, se compara 4 con 5, luego se ejecuta el segundo *if* ya que $4 < 5$: Tra_1 toma el valor [4] y se incrementa j_1 pasando a valer 5.

A continuación se realiza la iteración 6 del bucle. Se comparan los consecuentes de las reglas j_1 y j_2 , es decir, se compara la etiqueta 5 con la etiqueta 5. Como son iguales se completan los segundos tramos de ambas tendencias, pasando A_1 a valer $[[0], [1, 3], [4, 5]]$, A_2 a $[[0], [2, 3], [5]]$, j_1 toma el valor 6 y j_2 el valor 5.

Finalmente se realiza la iteración 7 del bucle. Se compara la etiqueta 6 con la etiqueta 6. Ya que son iguales, A_1 pasa a valer $[[0], [1, 3], [4, 5], [6]]$, A_2 a $[[0], [2, 3], [5], [6]]$, j_1 toma el valor 7 y j_2 el valor 7. Estos valores provocan la salida del bucle *while*. De los cuatro condicionales que quedan por ejecutar no se realiza ninguno debido a que no se cumple ninguna de las condiciones. El resto de tendencias se realizan con el mismo algoritmo.

3.3. Comparando tramos

La última línea del algoritmo 2 realiza la comparación de cada tramo de las tendencias. El proceso de comparación de los tramos se realiza en dos fases:

1. Se realiza la unión de las reglas de cada tramo, obteniendo una estructura con dos componentes. El primer componente será un SLI_m obtenido como unión de los SLI_m de las reglas a unir, y el segundo, un conjunto de etiquetas lingüísticas obtenido como la unión de la etiquetas salida de cada una de las reglas a unir. La representaremos mediante lo que hemos denominado *UNION*: $U_i = \langle Ant_i, CtoEti_i \rangle$, donde Ant_i es un SLI_m y $CtoEti_i$ es un conjunto de etiquetas lingüísticas.
2. Comparación de la *UNION* U_1 y U_2 . Para ello se compara la Ant_1 con Ant_2 y $CtoEti_1$ con $CtoEti_2$ y se realiza una t-norma entre los valores obtenidos (ecuación 3). Para realizar estas comparaciones se presentarán posteriormente las funciones de similitud.

Para realizar la unión de dos SLI_m llamados SLI_1 y SLI_2 se realiza la unión de los m intervalos de SLI_1 y SLI_2 uno a uno [6]. Por

ejemplo, si tenemos las siguientes reglas difusas temporales:

- IF ((VP) and (N) and (NR or FP)) THEN VN
- IF ((P) and (N) and (FP)) THEN FN

Se deben unir los $SLI_3 = [[VP], [N], [NR, FP]]$ y $SLI_3 = [[P], [N], [FP]]$, luego su unión será el $SLI_3 = [[P, VP], [N], [NR, FP]]$. Con respecto al conjunto de etiquetas tenemos que $CtoEti = [VN] \cup [FN] = [VN, FN]$. Luego para estas dos reglas la *UNION* creada sería $U_i = \langle Ant_i = [[P, VP], [N], [NR, FP]], CtoEti_i = [VN, FN] \rangle$

La ecuación 3 muestra cómo se calcula la similitud entre dos estructuras U_1 y U_2 .

$$*(S(Ant_1, Ant_2), R(CtoEti_1, CtoEti_2)) \quad (3)$$

donde $S(Ant_1, Ant_2)$ se rige por la ecuación 4, y es la utilizada para comparar dos SLI_m .

$$S(Ant_1, Ant_2) = 1 - |MOM(Ant_1) - MOM(Ant_2)| \quad (4)$$

donde $MOM(Ant_1)$ es el valor central del intervalo (*media de máximos* (MOM) [4]).

Para comparar los conjuntos de etiquetas se utilizará la función $R(CtoEti_a, CtoEti_b)$ (Ecuación 5).

$$\frac{N^\circ \text{ etiquetas comunes a } CtoEti_a \text{ y } CtoEti_b}{N^\circ \text{ etiquetas del intervalo representado}} \quad (5)$$

Como puede verse es una fracción entre el número de etiquetas que pertenecen a $CtoEti_a$ y $CtoEti_b$, entre el número de etiquetas totales en el intervalo entre la mínima etiqueta de $CtoEti_a$ o $CtoEti_b$ y la máxima etiqueta de $CtoEti_a$ o $CtoEti_b$ (soporte total que representa $CtoEti_a \cup CtoEti_b$).

Como consecuencia del cálculo de la similitud de los tramos se obtiene un conjunto ordenado de medidas de similitud llamado *simil* $= \langle s_1, s_2 \dots s_n \rangle$ donde n es el número de tramos (una similitud por tramo).

Estamos muy interesados en obtener la información de la similitud de dos MDTs de forma lingüística, por ello, se ha definido un conjunto ordenado de etiquetas lingüísticas P que definirá *lo que se parecen* los tramos de los modelos. Lo que haremos será calcular la pertenencia de la medida de similitud (s_i de *simil*) a las etiquetas de P , seleccionando la etiqueta de mayor pertenencia.

4. Un ejemplo de aplicación

Para probar el algoritmo se han utilizado dos series temporales que representan dos pasos humanos. Normalmente, los MDTs de dos pasos humanos son prácticamente iguales ya que todos caminamos de forma muy parecida. Por este motivo se han suprimido algunos ejemplos de los dos pasos a comparar para que existan diferencias entre los MDTs obtenidos.

Hemos utilizado tres variables de entrada y una de salida. Las variables de entrada son el ángulo de flexión y extensión de la cadera derecha (RH), rodilla derecha (RK) y tobillo derecho (RD); la variable de salida es el ángulo de flexión y extensión de la rodilla izquierda (LK).

Para obtener los MDTs se necesitan conjuntos ordenados de etiquetas para las variables de entrada y la variable de salida. Para cada variable se utiliza un conjunto ordenado de 7 etiquetas con la siguientes etiquetas: MN en Muy Negativa, N es Negativo, PN es Poco Negativo, NR es Normal, PP es Poco Positivo, P es Positivo y MP es Muy Positivo.

Aplicando el algoritmo de inducción de MDTs [6] se obtienen los MDTs de las tablas 1 y 2.

El algoritmo 3 se utiliza para obtener las tendencias, obteniendo las siguientes:

- $T_1 = [[MN, N, NR, PP, P, MP], [P, NR, PN, N, MN], [N], [MN]]$
- $T_2 = [[MN, PN, NR, P, MP], [P, PP, NR, PN, N, MN], [N], [MN]]$

A continuación se aplica el algoritmo 4 para calcular los tramos dentro de las tendencias, obteniendo los siguientes tramos:

Cuadro 1: MDT Obtenido.

| RH | RK | RD | LK |
|----------|----------|----------|----|
| {MP} | {MN} | {PP, P} | MN |
| {MP} | {N} | {PP} | N |
| {P} | {N} | {PP} | NR |
| {P} | {N} | {PP, P} | PP |
| {P} | {N} | {P} | P |
| {NR, PP} | {MN, N} | {P, MP} | MP |
| {PN, NR} | {MN} | {MP} | P |
| {N, PN} | {MN} | {MP} | NR |
| {N} | {MN} | {MP} | PN |
| {MN, N} | {MN} | {MP} | N |
| {MN} | {MN} | {P, MP} | MN |
| {MN..PP} | {MN..MP} | {MN..PP} | N |
| {P, MP} | {MN..MP} | {NR..P} | MN |

- $A_1 = [[MN], [N, NR], [PP, P], [MP], [P], [NR], [PN], [N], [MN], [N], [MN]]$
- $A_2 = [[MN], [PN, NR], [P], [MP], [P], [PP, NR], [PN], [N], [MN], [N], [MN]]$

Posteriormente se deben comparar tramo a tramo obteniendo una secuencia de similitudes llamada *simil*, que es como sigue: [0.93, 0.33, 0.5, 0.935, 0.93, 0.5, 1.0, 1.0, 0.93, 0.93, 0.93].

Finalmente, se calcula la etiqueta para cada valor de similitud de un tramo. Para esto se ha definido un conjunto ordenado de etiquetas llamado *parecido* con 5 etiquetas lingüísticas. Sus etiquetas son trapezoidales y vienen definidas por los valores de la Tabla 3. Al calcular estas etiquetas se obtiene la lista de etiquetas $LABELS = ['IGUAL', 'POCO', 'MEDIO', 'IGUAL', 'IGUAL', 'MEDIO', 'IGUAL', 'IGUAL', 'IGUAL', 'IGUAL', 'IGUAL']$.

Como fin de esta sección comentar que utilizando los tramos obtenidos (A_1 y A_2) y la lista $LABELS$ se puede describir lingüísticamente las "similitudes de dos MDTs" de la siguiente forma:

Los pasos del humano 1 (MDT_1) y del humano 2 (MDT_2) son IGUALES en el tramo $[MN]$, se parecen POCO en el tramo N a NR ,

Cuadro 2: MDT Obtenido.

| RH | RK | RD | LK |
|-----------|----------|----------|----|
| {MP} | {MN} | {P} | MN |
| {MP} | {N} | {NR, PP} | PN |
| {MP} | {N} | {PP} | NR |
| {P} | {N} | {P} | P |
| {PN...PP} | {MN, N} | {P, MP} | MP |
| {PN} | {MN} | {MP} | P |
| {N, PN} | {MN} | {MP} | PP |
| {N} | {MN} | {MP} | NR |
| {N} | {MN} | {MP} | PN |
| {MN, N} | {MN} | {MP} | N |
| {MN} | {MN, N} | {P, MP} | MN |
| {MN..PP} | {MN..MP} | {MN..PP} | N |
| {P, MP} | {MN..MP} | {NR..P} | MN |

Cuadro 3: MDT Obtenido.

| Nombre | a | b | c | d |
|--------|-------|-------|-------|-------|
| NADA | 0 | 0 | 0, 10 | 0, 15 |
| POCO | 0, 10 | 0, 15 | 0, 35 | 0, 40 |
| MEDIO | 0, 35 | 0, 40 | 0, 60 | 0, 65 |
| MUCHO | 0, 60 | 0, 65 | 0, 85 | 0, 90 |
| IGUAL | 0, 85 | 0, 90 | 1, 00 | 1, 00 |

tienen un parecido MEDIO en el tramo del *PP* a *P* y son IGUALES en el tramo *MP* (último de la primera tendencia).

De la misma forma se podrían describir los tramos del resto de tendencias.

5. Conclusiones

Se ha presentado un método de comparación de MDTs. Este método puede ser utilizado para comparar sistemas dinámicos. Además, esta comparación se puede ofrecer de manera lingüística, como se ha mostrado en la sección 4.

Creemos que es una buena herramienta para comparar lingüísticamente sistemas dinámicos del mismo tipo. El método presentado puede ser utilizado, por ejemplo, en entrenadores automáticos, o en detección de patologías (por

ejemplo, en la marcha humana). También podría ser utilizado en otros campos como en sistemas económicos.

Como trabajos futuros tenemos pensado la utilización de nuestro método en un ámbito de aplicación concreto para comprobar su eficiencia y su posibles mejoras.

AGRADECIMIENTOS

Este trabajo ha sido financiado por el Ministerio de Ciencia y Tecnología y la Junta de Comunidades de Castilla-La Mancha mediante los proyectos DIMOCLUST TIC2003-08807-C02-02 y PREDACOM PBC-03-004.

Referencias

- [1] Castro, J.L., Castro-Schez, J.J., Zurita, J.M. *Learning maximal structure rules in fuzzy logic for knowledge acquisition in expert systems*, Fuzzy Sets and Systems, vol. 101, pag. 331-342, 1999.
- [2] Delgado, M, Gonzalez, A. *An inductive learning procedure to identify fuzzy system*, Fuzzy Sets and Systems, vol. 55, pag. 121-132, 1993.
- [3] Gonzalez, A. *A learning methodology in uncertain and imprecise environments*, International Journal of Intelligence Systems, vol 10, pag. 357-371, 1995.
- [4] Leekwijck, W. V., Kerre, E. E.. *Defuzzification: criteria and classification*, Fuzzy Sets and Systems, vol. 108, n.2, pag. 159-178, 1999.
- [5] Luenberger, D.G. *Introduction to Dynamic Systems: Theory, Models and Applications*, John Wiley and sons, 1979.
- [6] Moreno-Garcia, J., Jimenez, L., Castro-Schez, J.J., Rodriguez, L., *A linguistic modelling approach for dynamic systems by using Temporal Fuzzy Models*. Enviado a Int. Journal of Approximate Reasoning, 2004.
- [7] Moreno-Garcia, J., Jimenez, L., Castro-Schez, J.J., Rodriguez, L., *Induction of Temporal Fuzzy Models*, ICEIS 2003 Proceedings, 2003.

- [8] Moreno-Garcia, J., Jimenez, L., Castro-Schez, J.J., Rodriguez, L.. *A direct linguistic induction method for systems*. Fuzzy Sets and Systems, Vol 146, 79-96, 2004.
- [9] K. Ogata. Ingenieria de Control Moderna. *Prentice-Hall HispanoAmericana SA*. 1998.
- [10] Paudit, S.M., Wu, S.M. , *Time Series and Systems Analysis, with applications*, New York, Wiley, 1983.
- [11] Tanaka, K. *An introduction to fuzzy logic for practical applications*. Springer, 1998.
- [12] Zadeh, L., *The concept of a linguistic variable and its applications to Approximate reasoning. Part I,II,III*, Information Science, 1975.