Evaluation of the Allen's relations for two Possibilistic Time Periods.

Abstract.

1 Introduction

In this work we propose the evaluation of the Allen's relations for time intervals for two given possibilistic time periods.

2 Implementation of the Allen's Relations

In the following we will consider a possibilistic time interval (PTI) be $I = [S_i, E_i]$. Where S_i and E_i are possibilistic time points representing the starting and the ending boundaries of the time interval.

Each possibilistic time point is given by $P = [D, a, b]$. This notation represents that:

- D is the central main point.
- $D - a$ is the left point.
- $D + b$ is the right point.

In the following, a time interval I will be noted as:

$$I = [S_i, E_i] \tag{1}$$
$$S_i = [D_{S_i}, a_{S_i}, b_{S_i}] \tag{2}$$
$$E_i = [D_{E_i}, a_{E_i}, b_{E_i}] \tag{3}$$

A relation between two crisp time points m, n is given by the following expression.

$$(n \; R \; m) \tag{4}$$

Where R is one of the following: $\{<, >, \leq, \geq, =\}$. The result of the expression is a boolean value indicating whether the point n is in the relation (R, m).

$$n \in A : (a, m) \in R \tag{5}$$

2.1 Translation to the possibilistic case

In order to implement the Allen's relations for two possibilistic valid-time periods, we need to translate the expression in (4) to the possibilistic case. In this case, we will note (4) as the following:

$$(P_n \ R \ P_m) \tag{6}$$

Where:

- P_n is either the starting or ending point of the possibilistic time interval given by $I = [S_i, E_i]$.
- P_m is either the starting or ending point of the possibilistic time interval given by $J = [S_j, E_j]$.
- $R \in \{<, >, \leq, \geq, =\}$.

The evaluation of the expression (6) is equivalent to the following ill-known constraints:

$$\begin{aligned} (P_n \ R \ P_m) &\triangleq \lambda\left(C_1, C_2\right) \\ C_1 &= (=, P_n) \\ C_2 &= (R, P_m) \end{aligned} \tag{7}$$

Consider that $a \in A \subseteq U$, then:

$$\begin{aligned} \mathrm{Pos}\left(C_1\left(A\right)\right) &= \min_{a \in A}\left(\pi_{P_n}\left(a\right)\right) \\ \mathrm{Nec}\left(C_1\left(A\right)\right) &= \min_{a \in A}\left(1 - \pi_{P_n}\left(a\right)\right) \\ \mathrm{Pos}\left(C_2\left(A\right)\right) &= \min_{a \in A}\left(\sup_{\{a,\omega\} \in R} \pi_{P_m}\left(\omega\right)\right) \\ \mathrm{Nec}\left(C_2\left(A\right)\right) &= \min_{a \in A}\left(\inf_{\{a,\omega\} \notin R} 1 - \pi_{P_m}\left(\omega\right)\right) \end{aligned} \tag{8}$$

With $\lambda = (\wedge, (C_1, C_2))$, we obtain:

$$\begin{aligned} \mathrm{Pos}\left(\lambda\left(A\right)\right) &= \max \min_{a \in A}\left(\mathrm{Pos}\left(C_1\left(A\right)\right), \mathrm{Pos}\left(C_2\left(A\right)\right)\right) \\ \mathrm{Nec}\left(\lambda\left(A\right)\right) &= \max \min_{a \in A}\left(\mathrm{Nec}\left(C_1\left(A\right)\right), \mathrm{Nec}\left(C_2\left(A\right)\right)\right) \end{aligned} \tag{9}$$

2.2 Before

The crisp version for the before operator for two crisp intervals $i = [s_i, e_i]\,, j = [s_j, e_j]$.

$$i \text{ Before } j \triangleq (e_i < s_j) \tag{10}$$

This is translated to the possibilistic case: $I = [S_i, E_i]\,, J = [S_j, E_j]$.

$$
\begin{aligned}
I \text{ Before } J &\triangleq \lambda\left(C_1, C_2\right) \\
C_1 &\triangleq (=, E_i) \\
C_2 &\triangleq (<, S_j)
\end{aligned} \tag{11}
$$

The possibility and the necessity of the Before relation is computed as follows:

$$
\begin{aligned}
\operatorname{Pos}(\lambda(A)) &= \max \min_{a \in A} \left(\operatorname{Pos}\left(C_1(A)\right), \operatorname{Pos}\left(C_2(A)\right)\right) \\
\operatorname{Nec}(\lambda(A)) &= \min \min_{a \in A} \left(\operatorname{Nec}\left(C_1(A)\right), \operatorname{Nec}\left(C_2(A)\right)\right)
\end{aligned} \tag{12}
$$

Example 1. Consider the following possibilistic time periods:

$$
\begin{aligned}
I &= \left([1, 2, 2], [6, 2, 2]\right) \\
J &= \left([6, 1, 1], [9, 1, 1]\right)
\end{aligned} \tag{13}
$$

We want to know the possibility of I Before J. As a result of triangle's line intersection, we get:

$$
\begin{aligned}
\operatorname{Pos}(\lambda(A)) &= -\frac{-D_{S_j} + D_{E_i} - a_{E_i}}{a_{S_j} + a_{E_i}} \\
&= -\frac{-6 + 6 - 2}{2 + 1} = \frac{2}{3}
\end{aligned} \tag{14}
$$

$$
\operatorname{Nec}(\lambda(A)) = 0 \tag{15}
$$

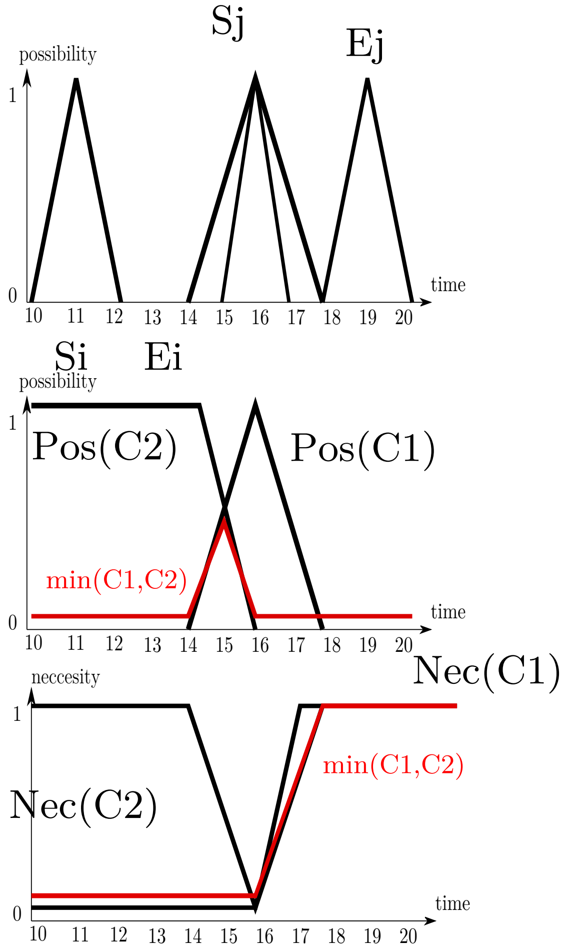
3 Evaluation

In this section we will explain the evaluation of the Allen's relations in the possibilistic case defined by sets of ill-known constraints. First of all, we will consider that the representation for both starting and ending points is a triangular membership function. Therefore, the possibilistic interval is defined by two triangular membership functions. The membership function for a time point $X = [D, a, b]$ is given by the following equation:

$$
\mu(x) = \begin{cases}
0, & \text{if } x \leq (D - a) \vee x \geq (D + b) \\
1, & \text{if } x = D \\
\frac{x - (D - a)}{a}, & \text{if } x > (D - a) \wedge x < D \\
\frac{D + b - x}{b}, & \text{if } x > D \wedge x < (D + b)
\end{cases} \tag{16}
$$

The last two cases are based on the equation for the line and a point. The following equations are the membership functions for the ill-known constraints:

Fig. 1: Illustration of the before relationship.



– $(<, X)$:

$$\mu\left(x\right)=\begin{cases}1, & \text{if } x \leq D-a \\ 0, & \text{if } x \geq D \\ \frac{x-D}{a}, & \text{if } x > (D-a) \wedge x < D\end{cases} \tag{17}$$

– $(\leq, X)$

$$\mu\left(x\right)=\begin{cases}1, & \text{if } x \leq D \\ 0, & \text{if } x \geq D+b \\ \frac{D+b-x}{b}, & \text{if } x > D \wedge x < (D+b)\end{cases} \tag{18}$$

– $(>, X)$

$$\mu(x) = \begin{cases} 0, & \text{if } x \leq D \\ 1, & \text{if } x \geq D + b \\ \frac{x-D}{b}, & \text{if } x > D \wedge x < (D + b) \end{cases} \tag{19}$$

– $(\geq, X)$

$$\mu(x) = \begin{cases} 0, & \text{if } x \leq (D - a) \\ 1, & \text{if } x \geq D \\ \frac{x-(D-a)}{a}, & \text{if } x > (D - a) \wedge x < D \end{cases} \tag{20}$$

– $(=, X)$, see equation (16).

– $(\neq, X)$

$$\mu(x) = \begin{cases} 1, & \text{if } x \leq (D - a) \wedge x \geq (D + b) \\ \frac{D-x}{a}, & \text{if } x > (D - a) \wedge x < D \\ \frac{x-D}{b}, & \text{if } x > D \wedge x < (D + b) \end{cases} \tag{21}$$

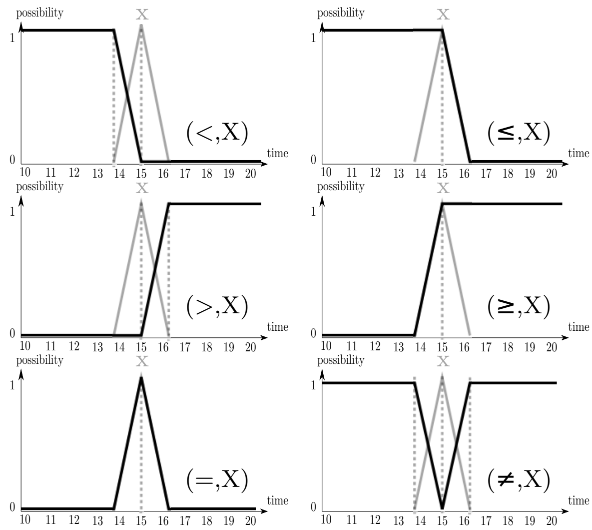


Fig. 2: Representation for the membership functions for each ill-known constraint.

The algorithm to compute the intersections between two ill-known constraints is given by the following algorithm:

For a quick reference, the following table 1 presents the results for the function solve(l1,l2).

3.1 evaluation for allen's relations

In this subsection we are going to implement the allen's relationships for two ill-known time intervals.

input : Two ill-known values X, Y with triangular membership functions and two ill-known constraints in the form $C_1 \triangleq (\theta, Y)$, $C_2 \triangleq (=, X)$

output: The possibility degree for $C_1 \wedge C_2$

Data: ;
$X = [D_x, a_x, b_x]$;
$Y = [D_y, a_y, b_y]$;
$C_1 \triangleq (\theta, Y), \theta \in (<, \leq, >, \geq)$;
$C_2 \triangleq (=, X)$

Result: $\mathrm{Pos}\,(C_1 \wedge C_2)$

`/* Based on the standard equation of a line passing through 2 points: */`
gen:$\leftarrow \frac{x-x_1}{x_2-x_1} = \frac{y-y_1}{y_2-y_1}$;
`/* The definition of the left and the right lines for the ill-known constraint` $(=, X)$`: */`
li:$\leftarrow$gen $\{x_1 = (D_x - a_x), x_2 = D_x, y_1 = 0, y_2 = 1\}$;
ld:$\leftarrow$gen $\{x_1 = (D_x + b_x), x_2 = D_x, y_1 = 0, y_2 = 1\}$;
$sol \leftarrow \emptyset$; `/* The set that contains the candidate solutions */`
switch θ **do**
  case $<$
    ld1:$\leftarrow$gen $\{x_1 = (D_y + a_y), x_2 = D_y, y_1 = 1, y_2 = 0\}$;
  end
  case $\leq$
    ld1:$\leftarrow$gen $\{x_1 = (D_y + b_y), x_2 = D_y, y_1 = 0, y_2 = 1\}$;
  end
  case $>$
    li1:$\leftarrow$gen $\{x_1 = D_y, x_2 = (D_y + b_y), y_1 = 0, y_2 = 1\}$;
  end
  case $\geq$
    li1:$\leftarrow$gen $\{x_1 = D_y, x_2 = (D_y - a_y), y_1 = 1, y_2 = 0\}$;
  end
  case $=$
    ld1:$\leftarrow$gen $\{x_1 = (D_y + b_y), x_2 = D_y, y_1 = 0, y_2 = 1\}$;
    li1:$\leftarrow$gen $\{x_1 = D_y, x_2 = (D_y - a_y), y_1 = 1, y_2 = 0\}$;
  end
  case $\neq$
    ld1:$\leftarrow$gen $\{x_1 = (D_y + a_y), x_2 = D_y, y_1 = 1, y_2 = 0\}$;
    li1:$\leftarrow$gen $\{x_1 = D_y, x_2 = (D_y + b_y), y_1 = 0, y_2 = 1\}$;
  end
end
`/* Now compute the intersections between each pair of lines: */`
$sol_1 \leftarrow$ solve (li,ld1);
$sol_2 \leftarrow$ solve (ld,ld1);
$sol_3 \leftarrow$ solve (li,li1);
$sol_4 \leftarrow$ solve (ld,li1);
`/* Finally get the result value: */`
max $\leftarrow 0$;
foreach $sol[i] \in sol$ **do** **if** $sol[i] \in [0, 1] \wedge sol[i] >$ max **then**
  max $= sol[i]$;
return max;

Algorithm 1: Computation of the possibility for two ill-known constraints.

Table 1: Results for the function solve

Operator	Results
$<$	$sol_1 = -\left(-D_y + D_x - a_x\right) / \left(a_x + a_y\right)$ $sol_2 = \left(-D_y + D_x + b_x\right) / \left(b_x + a_y\right)$
$\leq$	$sol_1 = -\left(-D_y + D_x - b_y - a_x\right) / \left(a_x + b_y\right)$ $sol_2 = \left(-D_y + D_x - b_y + b_x\right) / \left(b_x - b_y\right)$
$>$	$sol_1 = -\left(-D_y + D_x - a_x\right) / \left(a_x - b_y\right)$ $sol_2 = \left(-D_y + D_x + b_x\right) / \left(b_x + b_y\right)$
$\geq$	$sol_1 = -\left(-D_y + D_x + a_y - a_x\right) / \left(a_x - a_y\right)$ $sol_2 = \left(-D_y + D_x + b_x + a_y\right) / \left(b_x + a_y\right)$
$=$	$sol_1 = -\left(-D_y + D_x - a_x\right) / \left(a_x + a_y\right)$ $sol_2 = \left(-D_y + D_x + b_x\right) / \left(b_x + a_y\right)$ $sol_3 = -\left(-D_y + D_x + a_y - a_x\right) / \left(a_x - a_y\right)$ $sol_4 = \left(-D_y + D_x + b_x + a_y\right) / \left(b_x + a_y\right)$
$\neq$	$sol_1 = -\left(-D_y + D_x - b_y - a_x\right) / \left(a_x + b_y\right)$ $sol_2 = \left(-D_y + D_x - b_y + b_x\right) / \left(b_x - b_y\right)$ $sol_3 = -\left(-D_y + D_x - a_x\right) / \left(a_x - b_y\right)$ $sol_4 = \left(-D_y + D_x + b_x\right) / \left(b_x + b_y\right)$

Table 2: Allen's relations for two ill-known values

Relation	Crisp relation	Ill-known constraints	Combination
Before	$e_i < s_j$	$C_1 \triangleq (<, S_j)$ $C_2 \triangleq (=, E_i)$	$C_1 \wedge C_2$
Equals	$e_i = e_j \wedge$ $s_i = s_j$	$C_1 \triangleq (=, E_i)$ $C_2 \triangleq (=, E_j)$ $C_3 \triangleq (=, S_i)$ $C_4 \triangleq (=, S_j)$	$C_1 \wedge C_2$ $\wedge$ $C_3 \wedge C_4$
Overlaps	$s_i < s_j \wedge$ $e_i > s_j \wedge$ $e_i < e_j$	$C_1 \triangleq (<, S_j)$ $C_2 \triangleq (=, S_i)$ $C_3 \triangleq (>, S_j)$ $C_4 \triangleq (=, E_i)$ $C_5 \triangleq (<, E_j)$	$C_1 \wedge C_2$ $\wedge$ $C_3 \wedge C_4$ $\wedge$ $C_5 \wedge C_4$
Meets	$e_i = s_j$	$C_1 \triangleq (=, S_j)$ $C_2 \triangleq (=, E_i)$	$C_1 \wedge C_2$
During	$(s_i < s_j \wedge$ $e_i \leq e_j) \vee$ $(s_i \geq s_j \wedge$ $e_i < e_j)$	$C_1 \triangleq (<, S_j)$ $C_2 \triangleq (=, S_i)$ $C_3 \triangleq (\leq, E_j)$ $C_4 \triangleq (=, E_i)$ $C_5 \triangleq (\geq, S_j)$ $C_6 \triangleq (<, E_j)$	$(C_1 \wedge C_2$ $\wedge$ $C_3 \wedge C_4)$ $\vee$ $(C_5 \wedge C_2$ $C_6 \wedge C_4)$
Starts	$s_i = s_j$	$C_1 \triangleq (=, S_j)$ $C_2 \triangleq (=, S_i)$	$C_1 \wedge C_2$
Finishes	$e_i = e_j$	$C_1 \triangleq (=, E_j)$ $C_2 \triangleq (=, E_i)$	$C_1 \wedge C_2$

Algorithm: Implementation of the Before allen's relationship.

input : Two ill-known values I, J with triangular membership functions.
output: The possibility degree for I Before J
Data:

$I = [S_i, E_i]$;
$J = [S_j, E_j]$;
`/* In this implementation we will only use` E_i `and` S_j `*/`
$E_i = [D_{E_i}, a_{E_i}, b_{E_i}]$;
$S_j = \left[D_{S_j}, a_{S_j}, b_{S_j}\right]$;
`/* Initialization of the result variable: */`
pos $\leftarrow 0$;
Result: Pos (I Before J)
if $D_{E_i} - a_{E_i} > D_{S_j}$ **then**
  pos $\leftarrow 0$;
else if $D_{E_i} + b_{E_i} < D_{S_j}$ **then**
  pos $\leftarrow 1$;
else
  pos $\leftarrow$ Intersects(C_1, C_2);
end

Algorithm 2: Implementation for the Before operator.

Algorithm: Implementation of the Equals allen's relationship.

input : Two ill-known values I, J with triangular membership functions.
output: The possibility degree for I Equals J
Data:

$I = [S_i, E_i]$;
$J = [S_j, E_j]$;
`/* In this implementation we will use` S_i, E_i, S_j, E_j `*/`
$S_i = [D_{S_i}, a_{S_i}, b_{S_i}]$;
$E_i = [D_{E_i}, a_{E_i}, b_{E_i}]$;
$S_j = \left[D_{S_j}, a_{S_j}, b_{S_j}\right]$;
$E_j = \left[D_{E_j}, a_{E_j}, b_{E_j}\right]$;
`/* Initialization of the result variable: */`
pos $\leftarrow 0$;
Result: Pos (I Equals J)
if $D_{S_i} = D_{S_j} \wedge a_{S_i} = a_{S_j} \wedge b_{S_i} = b_{S_j} \wedge D_{E_i} = D_{E_j} \wedge a_{E_i} = a_{E_j} \wedge b_{E_i} = b_{E_j}$ **then**
  pos $\leftarrow 1$;
else
  pos $\leftarrow$ max (Intersects (C_1, C_2) , Intersects (C_3, C_4));
end

Algorithm 3: Implementation for the Equals operator.

Algorithm: Implementation of the Overlaps allen's relationship.

input : Two ill-known values I, J with triangular membership functions.
output: The possibility degree for I Meets J
Data:

$I = [S_i, E_i]$;
$J = [S_j, E_j]$;
`/* In this implementation we will use` S_i, E_i, S_j, E_j `*/`
$S_i = [D_{S_i}, a_{S_i}, b_{S_i}]$;
$E_i = [D_{E_i}, a_{E_i}, b_{E_i}]$;
$S_j = \left[D_{S_j}, a_{S_j}, b_{S_j}\right]$;
$E_j = \left[D_{E_j}, a_{E_j}, b_{E_j}\right]$;
`/* Initialization of the result variable: */`
pos $\leftarrow 0$;
Result: $\text{Pos}\,(I \text{ Overlaps } J)$
if $D_{S_i} + b_{S_i} < D_{S_j} \wedge D_{E_i} - a_{E_i} > D_{S_j} \wedge D_{E_i} + b_{E_i} < D_{E_j}$ **then**
  pos $\leftarrow 1$;
else
  pos $\leftarrow \max\left(\text{Intersects}\,(C_1, C_2)\,, \text{Intersects}\,(C_3, C_4)\,, \text{Intersects}\,(C_5, C_4)\right)$;
end

Algorithm 4: Implementation for the Overlaps operator.

Algorithm: Implementation of the Meets allen's relationship.

input : Two ill-known values I, J with triangular membership functions.
output: The possibility degree for I Meets J
Data:

$I = [S_i, E_i]$;
$J = [S_j, E_j]$;
`/* In this implementation we will use` E_i, S_j `*/`
$E_i = [D_{E_i}, a_{E_i}, b_{E_i}]$;
$S_j = \left[D_{S_j}, a_{S_j}, b_{S_j}\right]$;
`/* Initialization of the result variable: */`
pos $\leftarrow 0$;
Result: $\text{Pos}\,(I \text{ Meets } J)$
if $D_{E_i} = D_{S_j} \wedge a_{E_i} = a_{S_j} \wedge b_{E_i} = b_{S_j}$ **then**
  pos $\leftarrow 1$;
else
  pos $\leftarrow \text{Intersects}(C_1, C_2)$;
end

Algorithm 5: Implementation for the Meets operator.

Algorithm: Implementation of the During allen's relationship.

input : Two ill-known values I, J with triangular membership functions.
output: The possibility degree for I During J
Data:

`/* In this implementation we will use` S_i, E_i, S_j, E_j `*/`

$S_i = [D_{S_i}, a_{S_i}, b_{S_i}]$;
$E_i = [D_{E_i}, a_{E_i}, b_{E_i}]$;
$S_j = \left[D_{S_j}, a_{S_j}, b_{S_j}\right]$;
$E_j = \left[D_{E_j}, a_{E_j}, b_{E_j}\right]$;
`/* Initialization of the result variable: */`
pos $\leftarrow 0$;
Result: $\text{Pos}\,(I \text{ During } J)$
if $\left(D_{S_i} + b_{S_i} < D_{S_j} \wedge D_{E_i} + b_{E_i} \leq D_{E_j} - a_{E_j}\right) \vee$
$\left(D_{S_i} - a_{S_i} \geq D_{S_j} + b_{S_j} \wedge D_{E_i} + b_{E_i} < D_{E_j}\right)$ **then**
 pos $\leftarrow 1$;
else
 pos $\leftarrow \min\left(\max\left(\text{Intersects}\left(C_1, C_2\right), \text{Intersects}\left(C_3, C_4\right)\right),\right.$
 $\left.\max\left(\text{Intersects}\left(C_5, C_2\right), \text{Intersects}\left(C_6, C_4\right)\right)\right)$;
end

Algorithm 6: Implementation for the During operator.

Algorithm: Implementation of the Starts allen's relationship.

input : Two ill-known values I, J with triangular membership functions.
output: The possibility degree for I Starts J
Data:

`/* In this implementation we will use` $S_i, , S_j$ `*/`

$S_i = [D_{S_i}, a_{S_i}, b_{S_i}]$;
$S_j = \left[D_{S_j}, a_{S_j}, b_{S_j}\right]$;
`/* Initialization of the result variable: */`
pos $\leftarrow 0$;
Result: $\text{Pos}\,(I \text{ Starts } J)$
if $D_{S_i} = D_{S_j} \wedge a_{S_i} = a_{S_j} \wedge b_{S_i} = b_{S_j}$ **then**
 pos $\leftarrow 1$;
else
 pos $\leftarrow \text{Intersects}\left(C_1, C_2\right)$;
end

Algorithm 7: Implementation for the Starts operator.

Algorithm: Implementation of the Finishes allen's relationship.

input : Two ill-known values I, J with triangular membership functions.
output: The possibility degree for I Finishes J
Data:

`/* In this implementation we will use` $S_i, , S_j$ `*/`
$S_i = [D_{S_i}, a_{S_i}, b_{S_i}]$;
$S_j = \left[D_{S_j}, a_{S_j}, b_{S_j}\right]$;
`/* Initialization of the result variable: */`
pos $\leftarrow 0$;
Result: Pos $(I$ Finishes $J)$
if $D_{E_i} = D_{E_j} \wedge a_{E_i} = a_{E_j} \wedge b_{E_i} = b_{E_j}$ **then**
  pos $\leftarrow 1$;
else
  pos $\leftarrow$ Intersects (C_1, C_2) ;
end

Algorithm 8: Implementation for the Finishes operator.

Acknowledgements

The researchers are supported by the grant BES-2009-013805 within the research project TIN2008-02066: *Fuzzy Temporal Information treatment in relational DBMS*.