Notes de cours

Raisonnement Temporel
La logique de ALLEN

- Module Intelligence Artificielle -

P. De Loor - 2001/2002 – deloor@enib.fr

➢ **Raisonnement temporel :**

Avant, Après, Pendant
Juste avant, longtemps après …
n secondes avant, durant m heures

T1 (Allen) :

Il met ses chaussures avant de marcher

Il marche avant d'arriver à l'école

⇒ Il met ses chaussures avant d'arriver à l'école

Problèmes :

- que peut-on exprimer ?
- comment raisonner ?

☞ **Il fume avant de travailler**

⇒ **lien temporel entre conduit et travailler ?**

Il fume pendant qu'il conduit

☞ **Il court ⇒ il aura couru (représentation logique du passé du futur ? ?)**

☞ ***Si A est juste avant B,***

est-ce que C peut être après A et avant B ?

☞ ***Plusieurs futurs possibles ? plusieurs passés possibles ?***

- **Nombreux domaines intéressés :**

Validation : exprimer ce qui doit arriver…

Diagnostic médical (scénario d'évolution)

Compréhension d'histoire (représentation du passé)

Planification (représentation du futur)

Linguistique …

- **Voies explorées (1/2) :**

  ♦ **Logiques temporelles modales (Prior) voire annexe :**

  Quoi ? : *Mise en place d'un système formel cohérent avec la notion de temps (grammaire, axiomes, règles d'inférence ...) : formaliser le temps.*

  Notion de futurs, de granularité, de possibilité …

  Pourquoi *: Raisonner sur le temps : vers un Prolog temporel…*

  Logique :

  predicat1 :- predicat2, prédicat3

  Temporelle modale :

  □ *(Predicat1 P predicat2).* □ *(P : précède, □ : nécessaire)*

  ◊ *(Predicat1 U predicat2).* *(U : jusqu'à ce que, ◊ : possible)*

  Casse Tête théorique

- **Voies explorées (2/2):**

♦ **Systèmes temporels (Allen, Mc Dermott) :**

Quoi ? : *Proposition d'une représentation temporelle concrète associée à des règles de déduction.*

Pourquoi ? *: Représenter des scénarios et assurer leur cohérence.*

♦ **Logiques temporelles « réifiées » (Allen, Mc Dermott):**

Quoi ? *Proposer une logique support d'un système temporel. Introduire des opérateurs permettant de passer du comment vers le quand.*

Pourquoi ? : *Interfacer les concepts du langage naturel avec la logique temporelle*

➢ Plan ……………..

- 1.Introduction
- 2. Système temporel de Allen
- 3. Algorithme de Allen
- Annexes :
    - logiques temporelles modales
    - logique de Mac Dermot
    - logique réifiée de Allen

➢ Système temporel de Allen

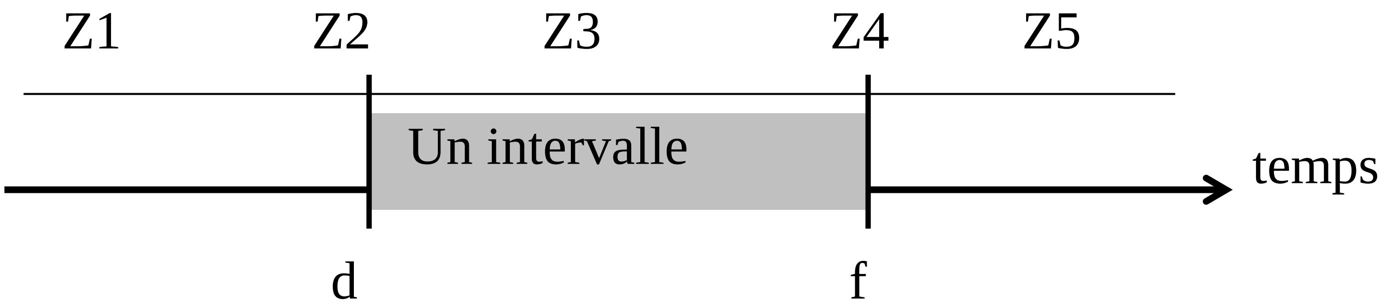
♦ Raisonnement sur les intervalles de temps :

- Possède un temps de début **d** et un temps de fin **f** , tels que **d** < **f**

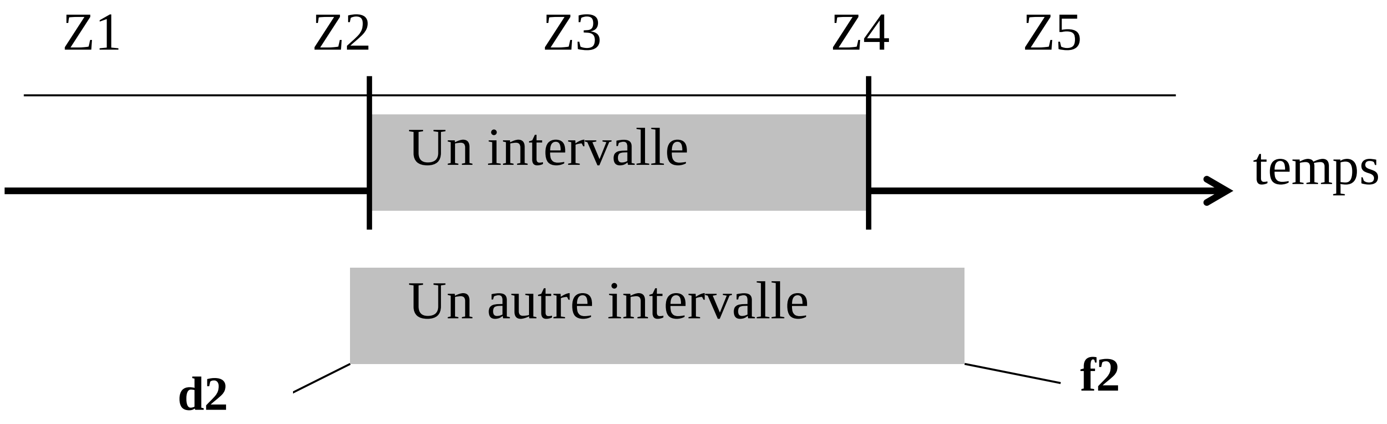
Exemple :

```
intervalle(ilFaitChaud).
intervalle(jaiSoif).
intervalle(jeBoisUneBiere).
```

- Un intervalle défini 5 zones dans le temps :

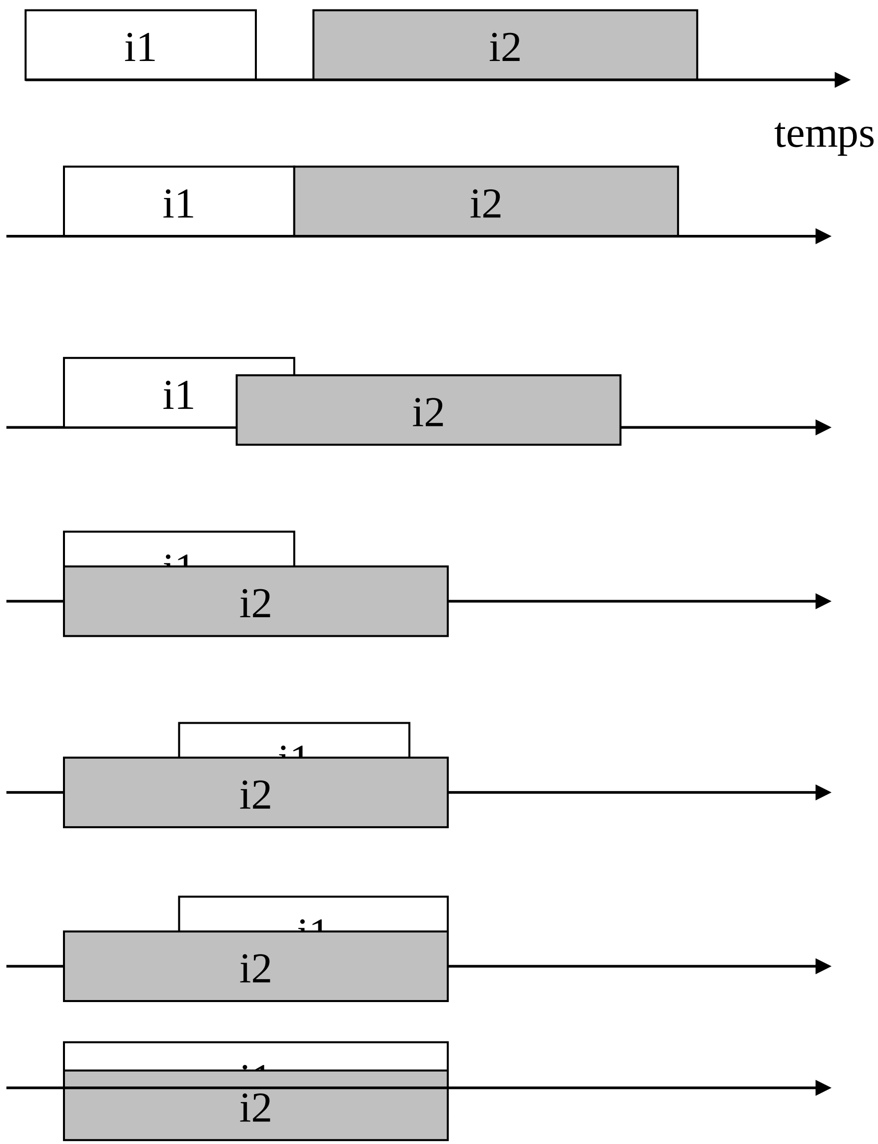


- Deux intervalles peuvent se positionner selon 13 configurations :



d2	**f2**	**Relations d'Allen**
Z1	Z1 ;Z2 ;Z3 ;Z4 ;Z5	< ;m ;o ;et ;dt
Z2	Z3 ;Z4 ;Z5	s ;= ;st
Z3	Z3 ;Z4 ;Z5	d ;e ;ot
Z4	Z5	mt
Z5	Z5	>

➢ Les 13 relations d'Allen :

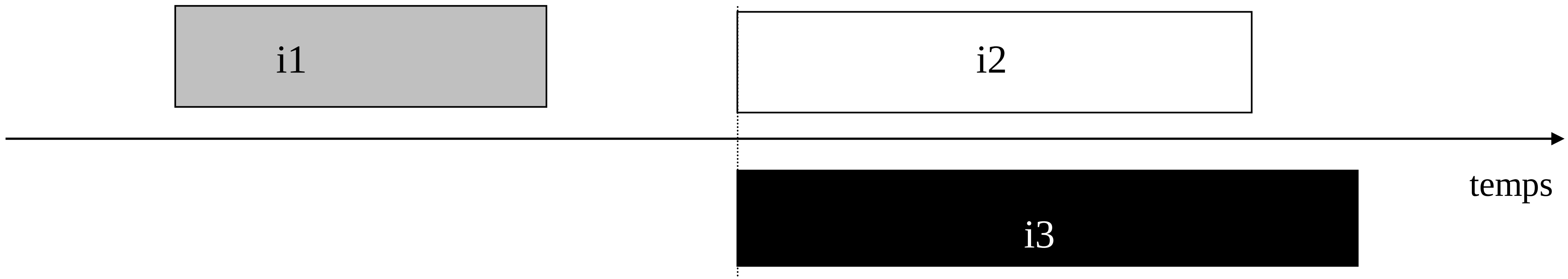


	Relations	Transposées (i1↔i2)
i1 < i2	i1précède i2	Suit (>)
i1 m i2	i1 rencontre i2 (m : meets)	rencontré par (m^t)
i1 o i2	i1 chevauche i2 (o : overlaps)	recouvert par (o^t)
i1 s i2	i1 début i2 (s : starts)	débuté par (s^t)
i1 d i2	i1 dans i2 (d : during)	contient (d^t)
i1 e i2	i1 termine i2 (e : ends)	terminé par (e^t)
i1 = i2	égalité	égalité

➢ Composition des relations d’Allen

• exemple de « **<** **○** **s** » (composition de la relation < avec la relation s)

(i1<i2) *et* ***(i2 s i3)***



i1 < i3

➢ Composition : Les incertitudes

- **Exemple :**

Marc est passé avant le spectacle (« *Présence Marc* » < « *Spectacle* »)
Céline a vu tout le spectacle (« *Spectacle* » d « *Présence Céline* »)

Donc :

Soit Marc est passé avant Céline (<), *il ne l'a pas vu*.
Soit Marc partait quand Céline arrivait (m), *il l'a croisée*.
Soit Marc est parti après que Céline soit arrivée (o), *ils ont pu parler*.
Soit Marc est arrivé avec Céline (s), *ils se connaissaient*.
Soit Céline était déjà là lorsque Marc est passé (d)

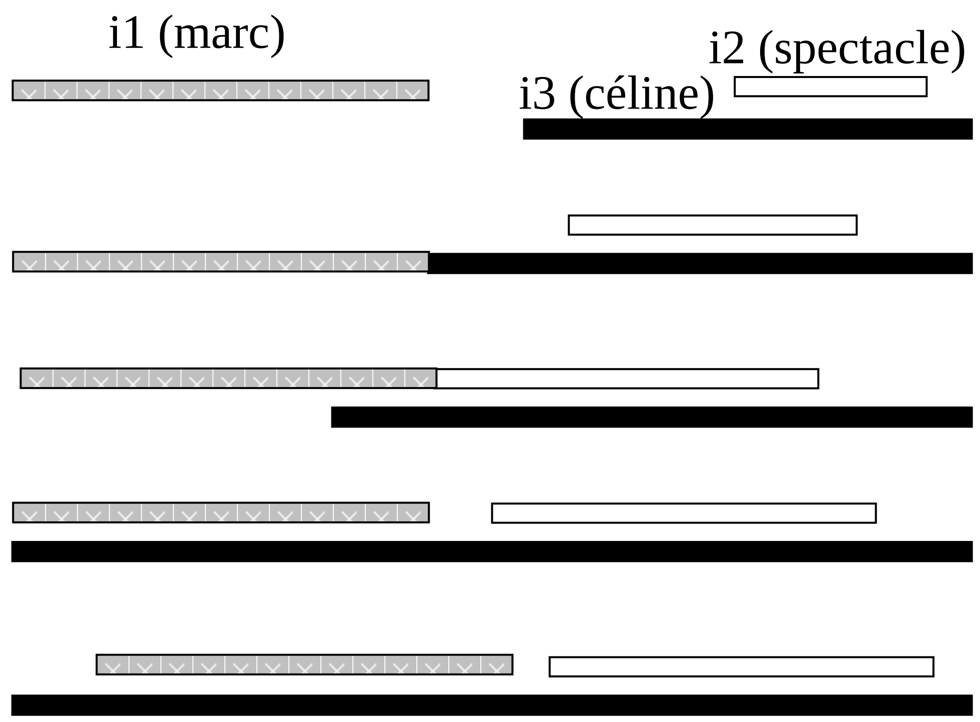
D'où l'incertitude : $(< \circ\, \boldsymbol{d}) \;=\; \boldsymbol{\{ <\, ;\, m\, ;\, o\, ;\, s\, ;\, d \}}$

➢ Composition des relations d'Allen

« **<** $\circ$ **d** » (composition de la relation < avec la relation d)

i1 : présence de marc, i2 : spectacle, i3 : présence de Céline

(i1<i2) *et* ***(i2 d i3)***



i1 < i3 (i3 > i1)

i1 m i3 (i3 m^t i1)

i1 o i3 (i3 o^t i1)

i1 s i3 (i3 s^t i1)

i1 d i3 (i3 d^t i1)

➢ Composition des relations d'Allen : Complexité

13 relations : 13*13 = 169 compositions !
Certaines compositions débouchent sur 9 relations possibles !

- **Simplification :**

La transposition :

Soient A et B deux relations d'Allen,

$$(A \circ B)^t = B^t \circ A^t, \text{ d'où } A \circ B = (B^t \circ A^t)^t$$

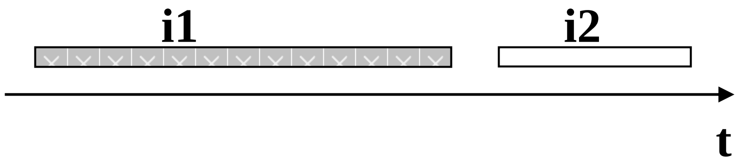
La symétrie :

Toute relation d'Allen A possède une symétrie temporelle A^s :

$$A \circ B = (A^s \circ B^s)^s$$

➢ Transposé, exemples :

$i1<i2$

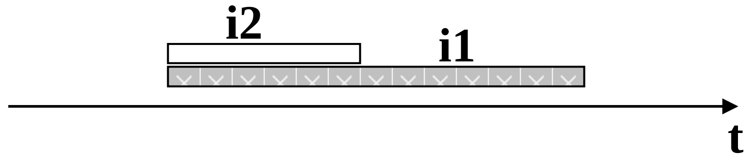


transposition

$i1>i2$

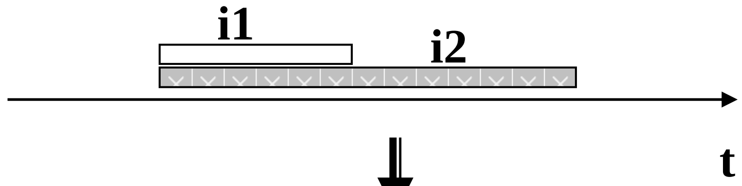
$$<^t \quad = \quad >$$

i2 s i1



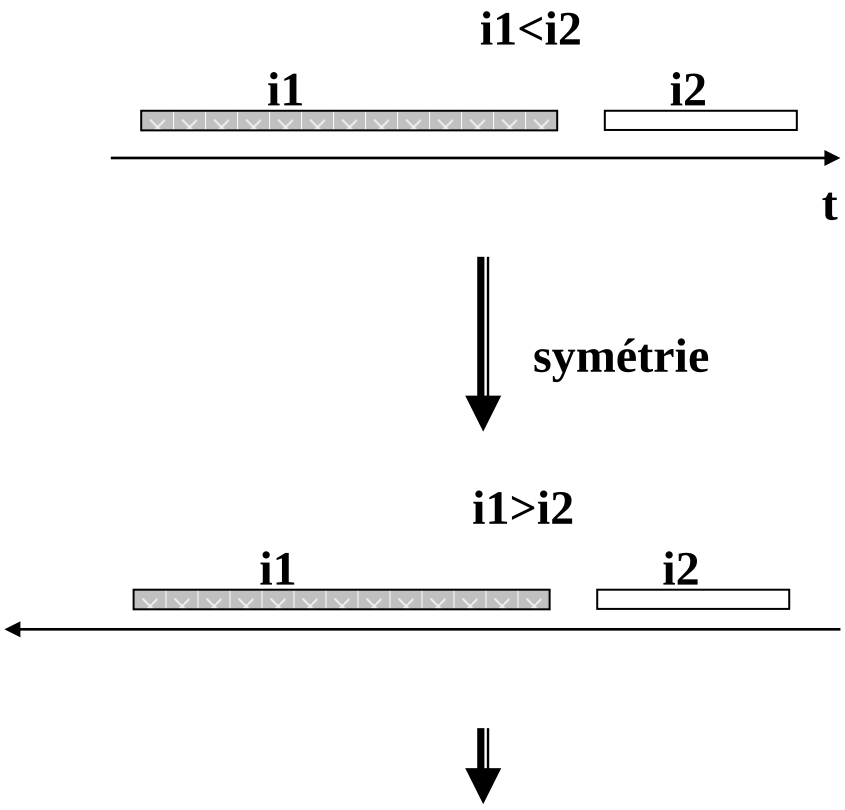
transposition

i2 s^t i1

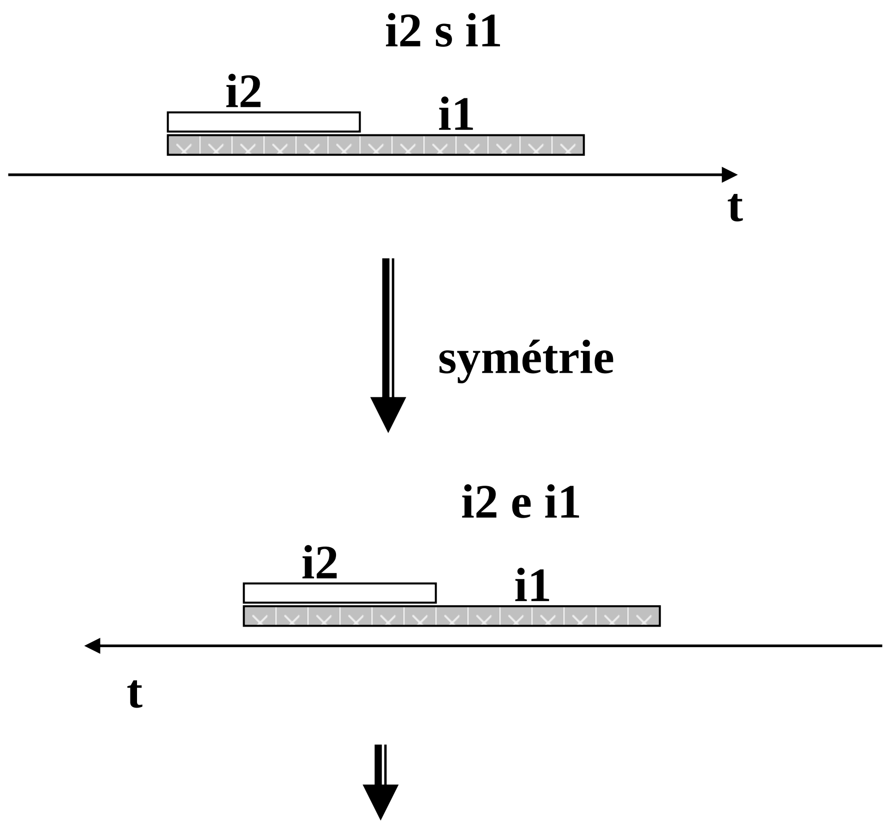


$$s^t \quad = \quad s^t$$

Symétrie, exemples :



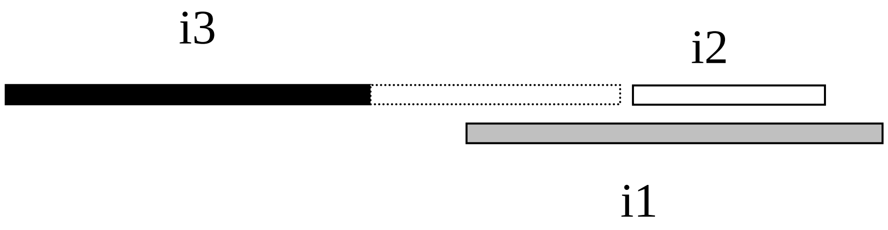
$$<^{s} \;=\; >$$



$$s^{s} \;=\; e$$

Simplification : exemple de la transposée

Soit i1 d^t i2 et i2 > i3



→ i1> i3 ; i1 o^t i3 ; i1 d^t i3 ; i1 m^t i3 ; i1 s^t i3.

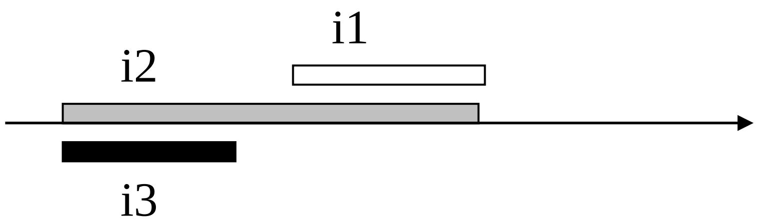
donc $(d^t \circ >) = > ; o^t ; d^t ; m^t ; s^t$

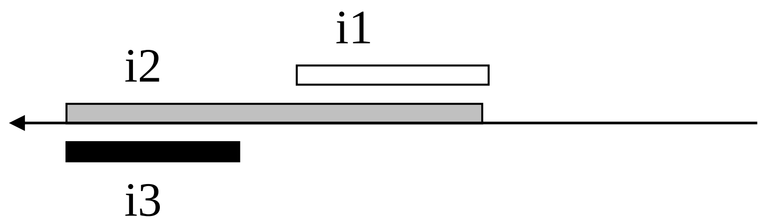
Calcul à l'aide des transposées :

$$(d^t \circ >) = (>^t \circ d^{tt})^t = (< \circ d)^t = (< ; o ; d, m; s)^t = (> ; o^t ; d^t ; m^t ; s^t)$$

Simplification : exemple de la symétrie

Relation | **Symétrie**





(i1 e i2) ; (i2 s^t i3) $\quad e^s = s \; ; \; (s^t)^s = e^t \quad$ (i1 s i2) ; (i2 e^t i3)

$$(s \circ e^t) = (<, m, o) \rightarrow (e \circ s^t) = (<, m, o)^s = (>, m^t, e^t)$$

Combinaison de la transposée et de la symétrie :

$$\mathbf{(A \circ B) = (B^{st} \circ A^{st})^{ts}}$$

➢ Compositions de base des relations d'Allen : 43 suffisent

	<	m	o	e^t	s	d	d^t	e	s^t	o^t	m^t	>
<	<	<	<	<	<	<mo sd	<	<mo sd	<	<mo sd	<mo sd	tout
m		<	<	<	m	osd	<	osd	m	osd	ee^t=	
o			<mo	<mo	o	osd	<mo e^td^t	osd	oe^td^t	oo^te e^tdd^t s^ts=		
s				<mo	s	d	<mo e^td^t	d	ss^t=			
e^t					o	osd	d^t	ee^t=				
d						d	tout					
d^t						oo^te e^tdd^t s^ts=						

➢ en Prolog

Génération des 169 relations sous la forme de prédicats

```
  :
compose(<, d, d).
compose(<, d, s).
compose(<, d, o).
  :
```

Génération d'un fichier prolog

43 relations de Allen de base (136 prédicats):

```
    :
compositionBase(<,<,<).
compositionBase(<,m,<).
    :
relations de symétrie
    :
sym(<,>).
sym(s,e).
    :
transposées
    :
transpo(<,>).
transpo(s,st).
    :
```

169 relations de Allen : 409 prédicats :

```
    :
compose(<, >, =).
compose(<, >, >).
compose(<, >, mt).
compose(<, >, ot).
compose(<, >, st).
compose(<, >, e).
compose(<, >, dt).
compose(<, >, d).
compose(<, >, s).
    :
```

Obtention de la liste des relations de composition de deux opérateurs :

```
ensCompo(OP1, OP2, L).
```

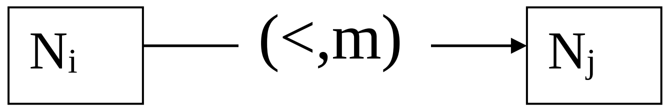
exemple :

```
1 ?- ensCompo(s,dt,L).

L = [dt, et, o, m, <].
```

Graphe Temporel

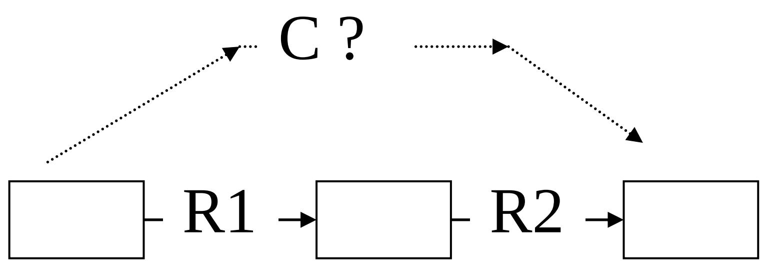
Un intervalle = 1 nœud.
Un arc = l'ensemble des relations de Allen liant deux intervalles.

Exemple : (i < j) ou (i m j) :



♦ **Composition = calcul de l'arc reliant deux nœuds, séparés par deux arcs :**

```
Contraintes (R1, R2)
C←{} ;
Pour tout r1 dans R1
        Pour tout r2 dans R2
                C ←C ∪(r1 o r2) ;
                Retourner C ;
```



➢ En Prolog :

- **Création du prédicat `contrainte (R1,R2,L).`**

  - **R1 et R2 sont deux listes de relations de Allen.**
  - **L est la liste calculée résultante.**

- **Exemple d'utilisation :**

```
7 ?- contrainte([o,s,mt,<],[dt,st],L).

L = [dt, et, o, m, =, st, s, >, <]

Yes
```

➢ Le prédicat `contrainte`

Procédure « Pour Tout » en prolog :

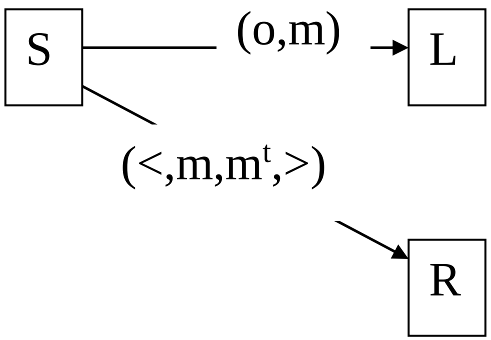
Contraintes (R1, R2)
C ← {} ;
Pour tout r1 dans R1
    Pour tout r2 dans R2
        C ← C ∪ (r1 o r2) ;
        Retourner C ;

```
contrainte([],_,[]).

contrainte([T|Q], L, R) :-
   contrainteElt(T,L,R1),
   contrainte(Q,L,R2),
   union(R1,R2,R).

contrainteElt(_,[],[]).
contrainteElt(E,[T|Q],R) :-
    ..
    ..
```

➢ **Construction d'un graphe temporel :**

Toute introduction d'un arc nécessite de recalculer l'ensemble des autres arcs.

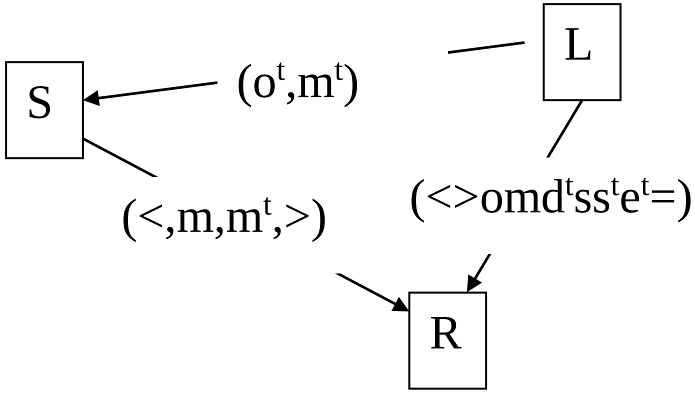
Exemple : S = instant où je touche l'interrupteur
L = temps durant lequel la lampe est allumée
R = temps durant lequel John est dans la pièce.



La lampe s'est allumée pendant ou juste après que j'ai appuyé sur l'interrupteur.
John n'était pas dans la pièce lorsque j'ai appuyé sur l'interrupteur.

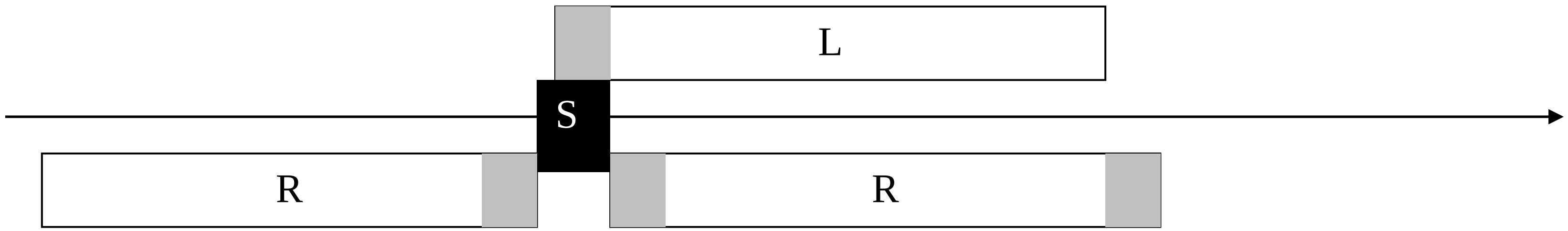
➢ Construction d'un graphe temporel :

♦ Calcul de l'arc entre L et R par l'algo Contraintes



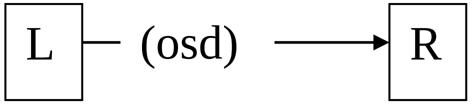
John n'était pas là au moment où j'ai appuyé, donc,

- soit il est venu avant que j'appuie
  - parti avant que la lampe s'allume (<)
- soit il est venu après que j'appuie
  - en même temps que la lampe s'allume (s,et, st,=)
  - après que la lampe s'allume (o,dt)
  - après que la lampe soit éteinte (>,m)



➢ Construction d'un graphe temporel :

- Nouvelle contrainte :

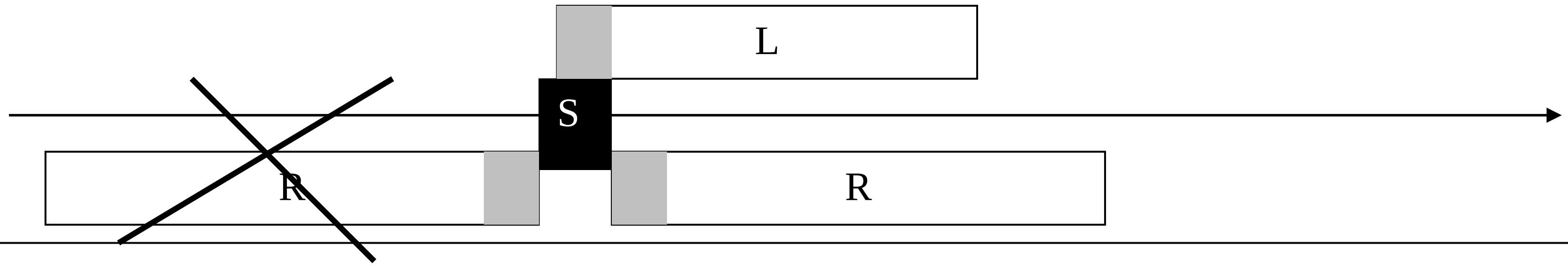


John était dans la pièce, lorsque la lumière a été éteinte.

- Nouvel arc entre L et R = conjonction des relations :

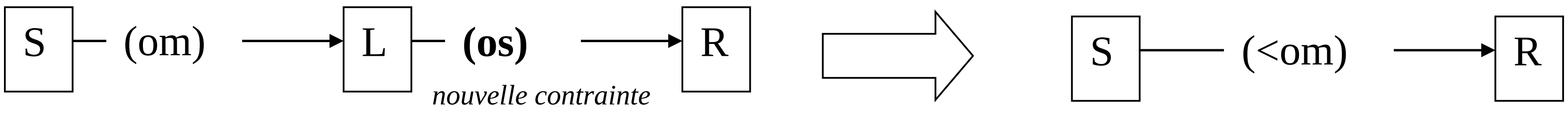
$$\text{Conjonction } ((<>oo^t m d^t s s^t e^t =),\ (osd)) = os$$

John était dans la pièce lorsque la lumière s'est éteinte, mais il n'y était pas lorsqu'elle s'est allumée.



➢ Construction d'un graphe temporel

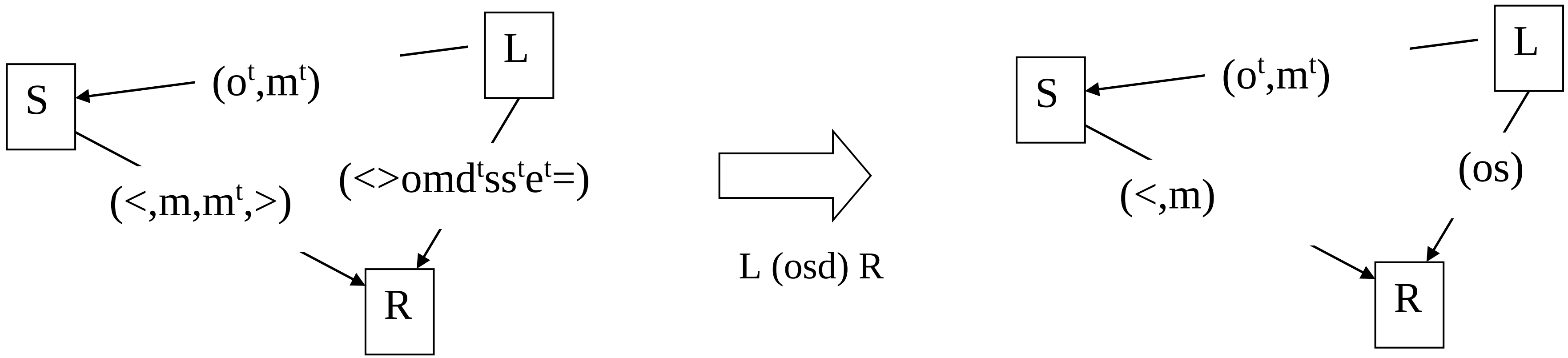
♦ Propagation de cette nouvelle contrainte sur le graphe :



♦ Conjonction avec l'ancienne valeur :

Conjonction (<om), (**>,**mt**,** m, <) = <m

John est arrivé dans la pièce après que j'ai touché l'interrupteur



➢ Problème de complexité :

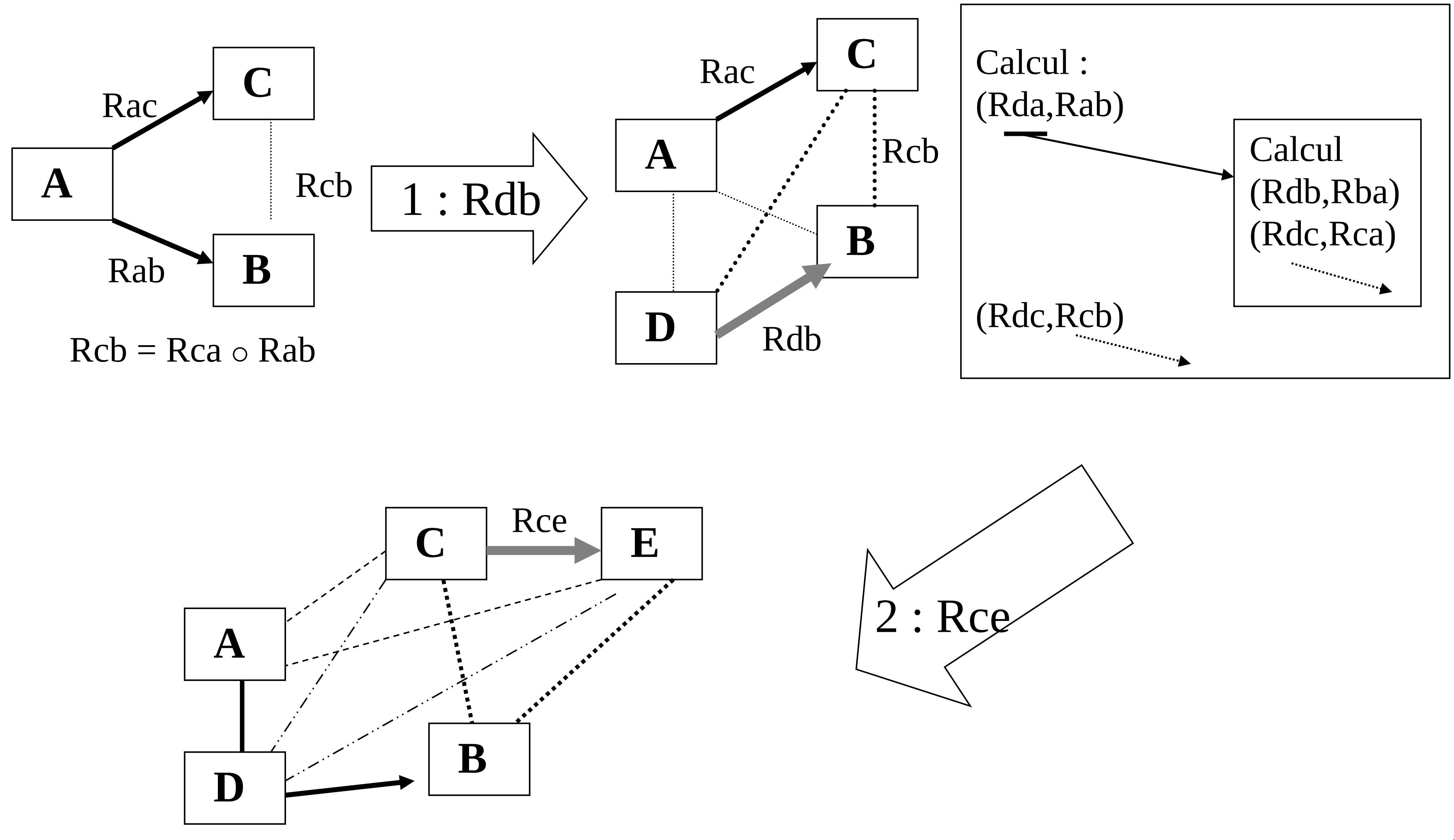
Le calcul de la cohérence d'un graphe temporel est un problème NP complet

- **Algorithme de Allen :** *complexité polynomiale*

Rab est une contrainte temporelle entre les nœuds a et b, que l'on veut ajouter à un graphe temporel de n nœuds :

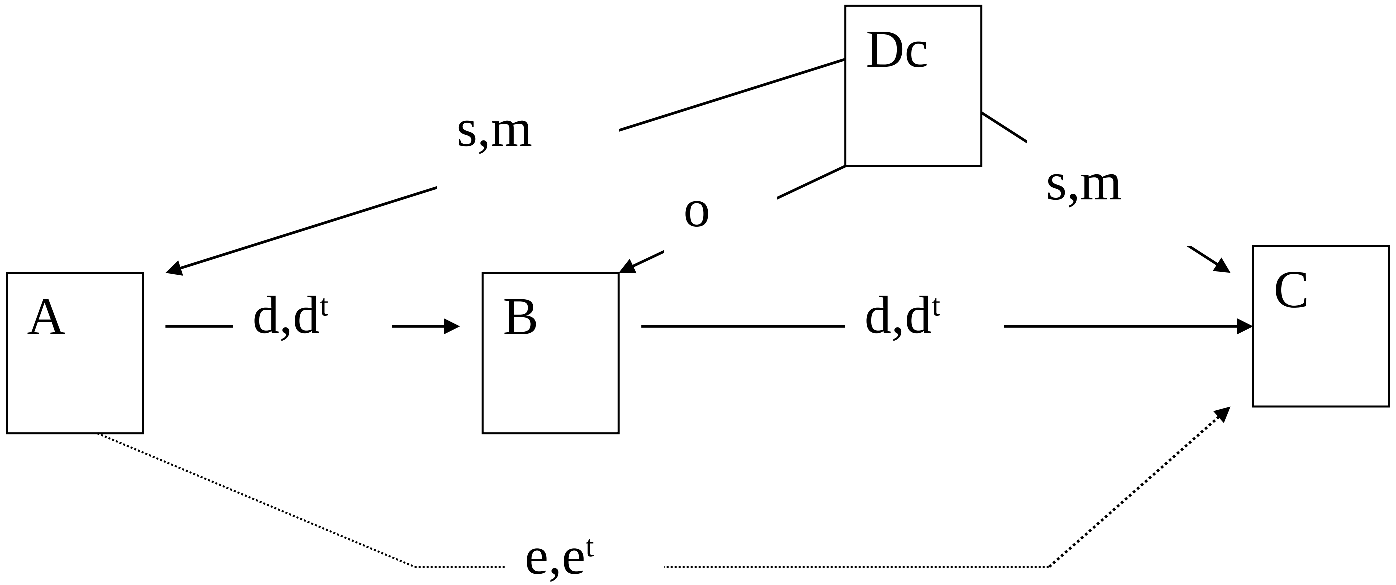
```
Propager (Rab)
    Empiler(Rab) ;
    Tant que la pile n'est pas vide faire
        Dépiler(Rij) ;
        Pour tout k dans [1,n], k≠i et k≠j faire
            Nouv-Rik ←conjonction(Rik, contrainte(Rij,Rjk)) ;
            Nouv-Rkj ←conjonction(Rkj, contrainte(Rki,Rij)) ;
            Si Nouv-Rik = ∅ ou Nouv- Rkj = ∅ Alors : Contradiction temporelle (Arret) ;
            Si Nouv-Rik ≠ Rik Alors
                Rik ← Nouv-Rik ;
                Empiler Rik ;
            Si Nouv-Rkj ≠ Rkj alors
                Rkj ← Nouv-Rkj ;
                Empiler(Rkj) ;
```

➢ Algorithme de Allen :



➢ Algorithme de Allen

- N'introduit pas d'inconsistance mais ne vérifie la consistance que sur des chemins de 3 nœuds



(A→B→C) ……… OK
(A→D→C) ………OK
(A→B→D→C)……inconsistance, non détectée pour « e ».

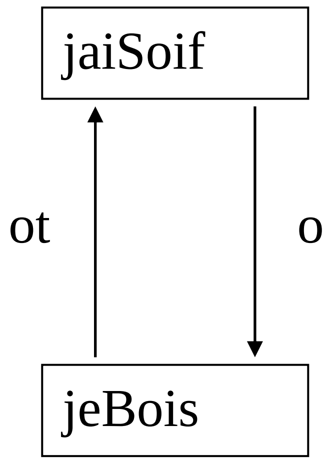
➢ Complexité

- Ajout d’une contrainte temporelle → simplification(arc i)
- simplification sur (arc i) → Ajout d’une contrainte temporelle
- 13 contraintes Maximums par couple d’intervalles
- 13 pas Maximum d’algorithme pour simplifier un arc
- Pour N nœud, l’introduction d’une contrainte peut prendre

  13*(N-1)(N-2)/2 itérations.
- L’emplacement mémoire nécessaire est proportionnel à 2^N.

➢ Notes sur le TP

Le Graphe temporel sera représenté à l'aide de deux prédicats




```
intervalle(I).
liste_ct(I1,I2,R).



exemple pour ce graphe :
?- intervalle(X).

X=jaiSoif.
Yes ;
X=jeBois ;
No

?- liste_ct(jeBois, jaiSoif, L).
L=[ot]

Yes
```

➢Notes sur le TP

Objectif :

```
?- ajouter(jaiSoif).

Yes
?- ajouter(jeBoisUneBiere).

Yes
?- ajouter(maCopineMeSurveille).

Yes
?- restreindre(jaiSoif, jeBoisUneBiere, [o]).

Yes
?- restreindre(jeBoisUneBiere, maCopineMeSurveille, [<,>]).

Yes
?- liste_ct(jaiSoif, maCopineMeSurveille,L).

L = [<,>,mt, ot, dt, st]
```

Exemple : Absence de s et de o : Il est impossible que j’ai soif au moment où elle arrive et que je n’ai plus soif avant qu’elle reparte (ma copine).

➢ **Notes sur le TP**

Le prédicat : `ajouter(I)`

- fixe les relations entre I et tous les intervalles déjà existants : par défaut, les 13 relations d'Allen :

```
ajouter(I) :- \+ clause(intervalle(I),_).
      majAllen([<,>,=,m,mt,o,ot,e,et,s,st,d,dt], I),
      assert(intervalle(I)).

      majAllen(R,I) :- findall(Q, intervalle(Q), L),
                          majAllen(R,I,L).

      majAllen(_,_,[]).
      majAllen(R,I,[T|Q]) :-     assert(liste_ct(I,T,R)),
                                 assert(liste_ct(T,I,R)),
                                 majAllen(R,I,Q).
```

Le Prédicat allen(I1,I2)

```
Propager (Rab)
    Empiler(Rab) ;
    Tant que la pile n'est pas vide faire
            Dépiler(Rij) ;
            Pour tout k dans [1,n], k≠i et k≠j faire
                    Nouv-Rik ←conjonction(Rik, contrainte(Rij,Rjk)) ;
                    Nouv-Rkj ←conjonction(Rkj, contrainte(Rki,Rij)) ;
                    Si Nouv-Rik = ∅ ou Nouv- Rkj = ∅ Alors :
Contradiction temporelle (Arret) ;
                    Si Nouv-Rik ≠ Rik Alors
                            Rik ← Nouv-Rik ;
                            Empiler Rik ;
                    Si Nouv-Rkj ≠ Rkj alors
                            Rkj ← Nouv-Rkj ;

                            Empiler(Rkj) ;
```

```
allen(A,B) :-
        razPile,
        liste_ct(A,B,Rab),
        empiler(liste_ct(A,B,Rab)),
        repeat,
                depiler(liste_ct(I,J,Rij)),
                findAllNodes(I,J),
                liste_ct(J,K,Rjk),
                liste_ct(K,I,Rki),
                contrainte(Rij,Rjk,ContRik),
                (
                        (
                                rstListeCt(I,K,ContRik,keep)
                        ) ;
                        (
                                rstListeCt(I,K,ContRik,replace),
                                liste_ct(I,K,NewRik),
                                empiler(liste_ct(I,K,NewRik))
                        )
                ),
                contrainte(Rki, Rij, ContRkj),
                (
                        (
                                rstListeCt(K,J,ContRkj,keep)
                        ) ;
                        (
                                rstListeCt(K,J,ContRkj,replace),
                                liste_ct(K,J,NewRkj),
                                empiler(liste_ct(K,J,NewRkj))
                        )
                ),
                testedAllNodes(K),
        pileVide, !.
```

repeat, write('un'), repeat, write('deux'), fail.

<u>Imbrication de deux boucles :</u>

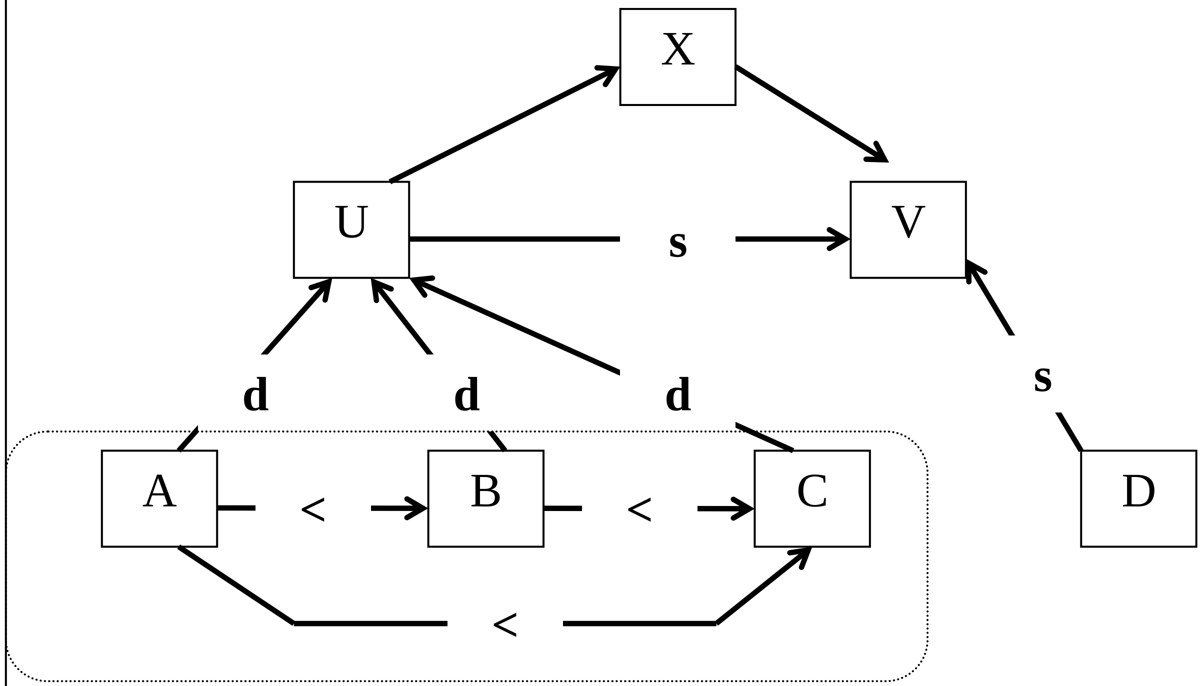
findAllNodes(I,J) : fait une assertion dynamique d'un prédicat nodes(LK)

LK étant la liste des intervalles K devant être traités dans la boucle « pour tout »

TestedAllNodes(K) retire K de LK et échoue tant que LK n'est pas vide.

Les intervalles de référence :

- Quoi : Limiter le temps de calcul de la cohérence du graphe
- Comment : En regroupant des intervalles sous une même référence temporelle : ici U avec (A d U), (B d U) et (C d U) et V avec (D s V).



Avantages :

- 10 relations au lieu de 20 :
- Toutes les relations entre les intervalles de U sont calculées indépendemment de U, V et D.

Difficultés :

- Définition des intervalles de références
- Recherche des relations entre intervalles d' « intervalles de références différents »

Logiques temporelles modales : Un aperçu

➢ Qu'est-ce qu'une logique modale ?

Logique qui admet un contexte, un univers ! ! !
Pallier aux paradoxes de la logique classique, défiant l'intuition.
Exemple : en logique classique, on a les théorèmes suivants :

$$B \rightarrow (A \rightarrow B)$$

$$(\text{car } A \rightarrow B \Leftrightarrow \neg A \vee B)$$

$$\neg A \rightarrow (A \rightarrow B)$$

A : la bière sans bulle existe.
B : la bière sans bulle est bonne.

Si la bière sans bulles n'existe pas, l'affirmation selon laquelle elle est bonne est " vraie " ! ! !

Manque la prise en compte d'un contexte (connaissance, vérité relative)

➢ Qu’est-ce qu’une logique modale ?

Une logique modale réfute l’implication logique pure et la remplace par l’implication stricte : « > »

$$(A>B) \Leftrightarrow \neg\Diamond(A\wedge\neg B)$$

(alors que $A \rightarrow B \Leftrightarrow \neg A \vee B$*)*

B : La bière sans bulle est bonne.
A : La bière sans bulle existe.

La bière sans bulle est bonne si il est impossible qu’elle existe sans être bonne.

En plus clair : ***On ne peut affirmer que la bière sans bulle est bonne que si elle existe et qu’en plus on la trouve bonne ! ! !***

➢ Qu'est-ce qu'une logique modale ?

Notions de base :	*Opérateurs*
Possibilité	◊
Nécessité	□

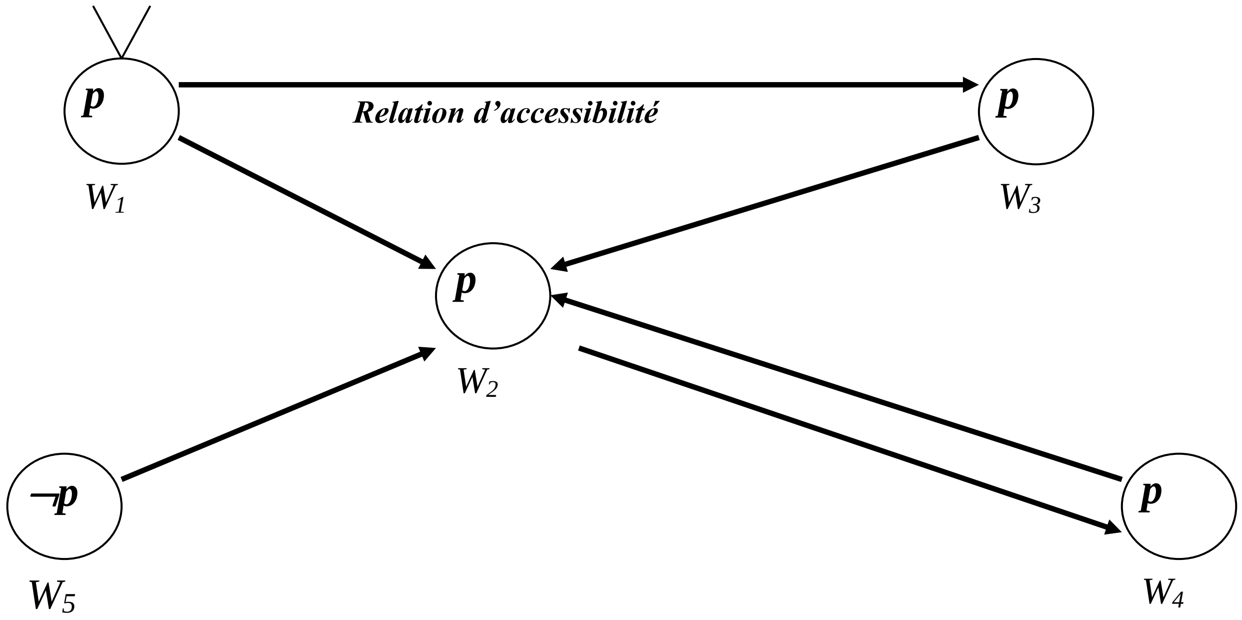
□A ⇔ ¬◊¬A

Les logiciens ont " re-fabriqué " des systèmes formels (alphabet, règles de formations, axiomes, règles d'inférences et sémantique)

¬A>(A>B) et A>(B>A) ne sont pas (toujours) des théorèmes

➢ Qu’est-ce qu’une logique modale ?

La notion de possibilité implique que la valeur de vérité d’un énoncé puisse évoluer. On parle d’évolutions dans des “ mondes possibles ”…



□ p est vraie en w_1,

◊¬p est faux (car w_5 n’est pas accessible)

Comprendre la différence ? ?

Logique classique	***Une logique modale (S4)***
Axiomes :	***Axiomes :***
(1) $A \rightarrow (B \rightarrow A)$	(K) $\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B)$
(2) $((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$	(T) $\Box A \rightarrow A$
(3) $(\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$	(1) $\Diamond A \rightarrow \Box \Diamond A$
Règles d'Inférences : *Modus Ponens*	***Règles d'Inférences :***
$$\frac{\vdash A \rightarrow B, \vdash A}{\vdash B}$$	$$\frac{\vdash A}{\vdash \Box A} \qquad \frac{\vdash A \rightarrow B, \vdash A}{\vdash B}$$

➢ Qu'est-ce qu'une logique modale temporelle ?

♦ **Mondes possibles** ⇒ **Moments possibles**

♦ **Introduction d'une relation de précédence R(t,t'), t précède t'**

(Anti-symétrie : $R(A,B) \rightarrow \neg R(B,A)$)

nouveaux opérateurs :

♦ **Nécessité : □** ⇒ **toujours**

♦ **Possibilité : ◊** ⇒ **parfois**

	passé	*futur*
toujours	G	H
parfois	F	P

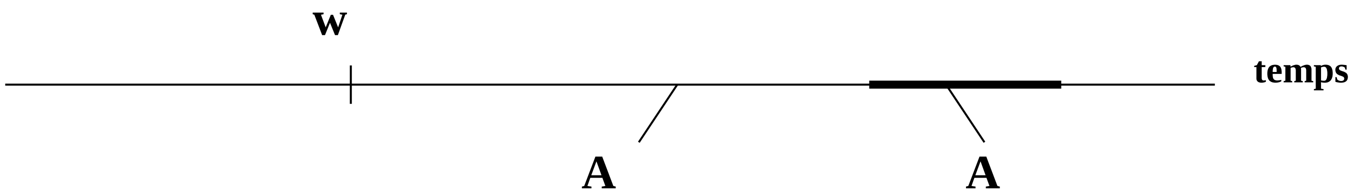
♦ **Instant suivant : ❍, Jusqu'à ce que : U, Précède : P**

Sémantique des logiques temporelles modales

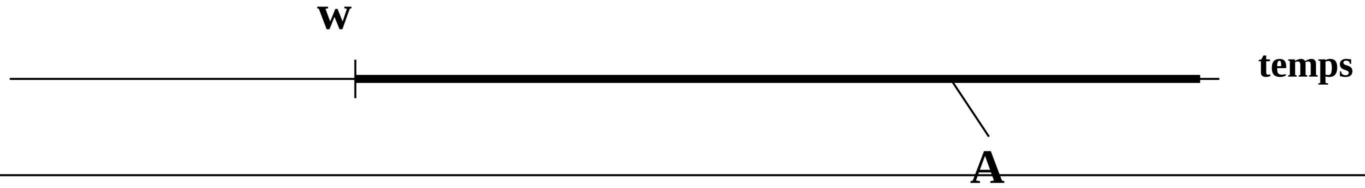
$M,w \models A$: M satisfait A à la date w

$M,w \models p$	ssi p est présent en w
$M,w \models \neg A$	ssi non $M,w \models A$
$M,w \models A \wedge B$	ssi $M,w \models A$ et $M,w \models B$
$M,w \models PA$	ssi il existe w' tel que $R(w,w')$ et $M,w' \models A$
$M,w \models FA$	ssi il existe w' tel que $R(w',w)$ et $M,w' \models A$
$M,w \models HA$	ssi quelque soit w' tel que $R(w,w')$ et $M,w' \models A$
$M,w \models FA$	ssi quelque soit w' tel que $R(w',w)$ et $M,w' \models A$

$M,w \models PA$:

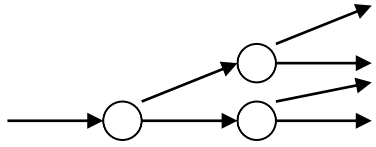


$M,w \models HA$:

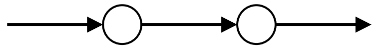


➢ Principaux éléments caractérisant les logiques temporelles modales

♦ Structure du temps :



- temps arborescent :
  - $\forall t \forall s \forall r \quad ((R(t,s) \wedge R(s,r)) \rightarrow R(t,r))$
  - $\forall t \forall s \forall r \quad ((R(t,r) \wedge R(s,r)) \rightarrow R(t,s) \vee t=s \vee R(s,t))$



- temps linéaire :
  - $\forall t \forall s \quad ((R(s,t) \vee s=t \vee R(t,s))$
  - $\forall t \forall s \forall r \quad ((R(r,t) \wedge R(s,r)) \rightarrow R(s,t) \vee t=s \vee R(t,s))$

♦ Nature du temps :

- discret
- continu
- fini ou infini
- métré, symbolique …

➢ Utilisation de la logique modale temporelle :

- Spécification

  - La routine “ Mot de Passe ” est toujours lancée avant la routine “ modif ” : $\square$((Mot de passe) P(modif))

- Raisonnement : Logique temporelle exécutable
  - MOLOG, TEMPLOG

- Démonstration Automatique des propriétés d’un programme
  - KRONOS, STEP

- Difficultés pratiques et théoriques : complexité, incomplétude et indécidabilité

➢ Système de McDermott

- Pas d'intervalles mais des états (S0, S1,…..) et des instants.

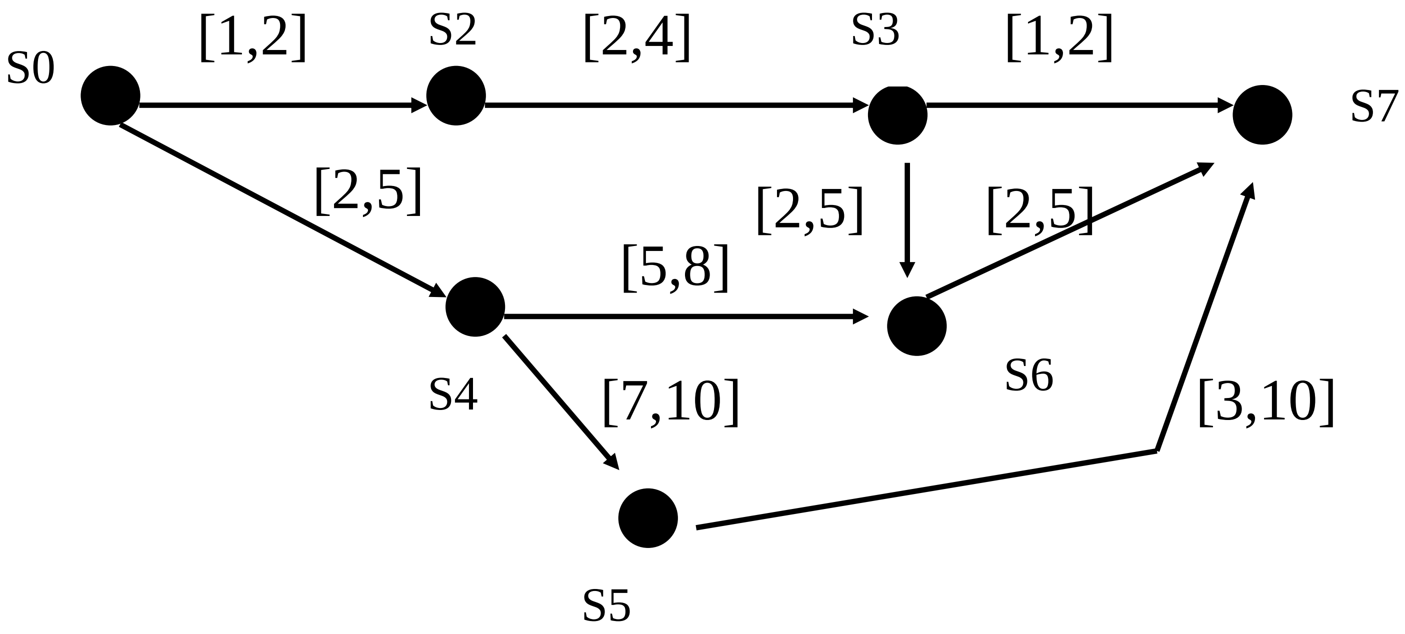
- Pas 13 relations mais 1 : avant ou même moment (≤)

- Un passé, plusieurs futurs possibles (temps arborescent à droite)
  - $(S0 \leq S1) \wedge (S0 \leq S2) \Rightarrow (S1 \leq S2) \vee (S2 \leq S1)$
  - → *Arbre composé de chroniques*

- fonction de datation : d(Si)

- Système symbolique ou numérique (combien de temps)

➢ Graphe temporel d'un système numérique



➢ Cohérence des chemins si Max(∑min) < min(∑Max)

- Cohérence des chemins S0 → S6 car 7 (s0s4s6) < 11 (s0s2s3s6)
- Incohérence des chemins S0 → S7 car (12 (s0s4s7) > 8 (s0s2s3s7))

➢ Logique temporelle réifiée de Allen :

Objectif :

- Décrire comment les choses se passent (causalité temporelle), plutôt que les moments où elles se passent.
- Introduction de termes (IA)
- Liaison avec le système temporel de Allen

Moyen :

- Association Action – Temps
- Introduction d’Axiomes spécifiques

Résultats critiqués.

- ***Exemple : A a lieu durant l'intervalle temporel I :***

HOLDS(A,I) *est une formule de la logique de Allen*

*Composant a-temporel : **A** (propriétés, événements, processus)*
*Composant temporel **I**, (intervalle ou instant)*

- **Axiomes de base :**

- **Sur les Intervalles :** *STARTS(t1,t2), FINISHES(t1,t2), BEFORE(t1,t2), OVERLAP(t1,t2), MEETS(t1,t2), EQUAL(t1,t2).*

  Reprennent les règles de compositions (ex : BEFORE(t1,t2)$\wedge$ BEFORE(t2,t3)$\Rightarrow$BEFORE(t1,t3)

- **Sur les Actions :** *HOLDS(A,I), OCCUR(E,I), OCCURING(E,I)*

HOLDS(A,I)

HOLDS(p,T) $\Leftrightarrow$ ($\forall$t, IN(t,T)$\Rightarrow$HOLDS(p,t)

HOLDS(and(p,q),t) $\Leftrightarrow$ HOLDS(p,t)&HOLDS(q,t)

HOLDS(not(p),T) $\Leftrightarrow$ ($\forall$t, IN(t,T) $\Rightarrow$ $\neg$(HOLDS(p,t)) (négation forte)

(où IN(t1,t2) $\Leftrightarrow$ (DURING(t1,t2)$\vee$STARTS(t1,t2) $\vee$ FINISHES(t1,t2) et $\forall$i $\forall$j IN(j,i).

OCCUR(E,I)

OCCUR(E,t)$\wedge$ IN(t,t') $\Rightarrow$ $\neg$OCCUR(E,t') (non décomposition des événements)

OCCURRING(E,I)

OCCURING(p,t) $\Rightarrow$ $\exists$t', IN(t',t) $\wedge$OCCURRING(p,t')

OCCUR(p,t) $\Rightarrow$ OCCURING(p,t)

➢ Description d'un événement :

1) Définir un prédicat **CHANGE-POS(Ball, x, y)**

2) Déclarer un événement particulier :

OCCUR(CHANGE-POS(Ball1, pos1, pos2), T1)

3) Préciser les conditions d'occurrence de ce type d'événement :

OCCUR(CHANGE-POS(objet, source, goal),t)$\Rightarrow$

$\exists$t1, t2.

MEETS(t1,t)&MEETS(t,t2)&

HOLDS(at(object,source),t1)&HOLDS(at(object,goal),t2).

➢ **Description d'une causalité d'événements :**

ECAUSE(e,t,e',t') $\Leftrightarrow$ OCCUR(e,t) $\Rightarrow$ OCCUR(e',t') $\wedge$IN(t',t)$\vee$BEFORE(t,t') $\vee$MEETS(t,t') $\vee$OVERLAPS(t,t') $\vee$ EQUALS(t,t')

Exemple :

ECAUSE(court-circuit(alimentation),i1 ,explosion(machine),i2)

- **Notion d'ACTIONS $\Rightarrow$ Notion d'AGENTS**

ACAUSE(i,a,t) $\Rightarrow$OCCUR(i,t)$\vee$OCCURING(i,t)

Si a est vrai, i aura lieu durant t

Exemple :

ACAUSE(i1, processeur1, déplacer(fichier1,disque1,disque2))

- **Une action peut être générée par une autre :**

GENERATE(a1,a2,t)

➢ PLANIFICATION

Séquence d'actions obligatoires :

TO-DO(action, t, plan)

Plus :

Notion de croyance, d'intentionnalité, (monde attendu, souhaité, planifié), de dissimulation.

COMMITED(agent,plan,t)
INTEND(agent, ACAUSE(agent, occurrence), t1, t2)
HIDE(agent, observer, object)

PLANIFICATION ⇒ faire coïncider le monde «souhaité» avec le monde planifié.

BILAN

- **Le système de ALLEN est le plus utilisé en IA**
- **La logique de ALLEN est critiquée par les puristes (ambiguités, bidouilles, pas de formalisation complète …44 axiomes)**
- **Il existe un Outil de planification : NOAH**
    - Spécification des contraintes temporelles pour appliquer une action.
    - Spécification des contraintes temporelles impliquées par les actions.
- **Mc Dermott présente une logique similaire basée sur son système**

➢ Bibliographie :

Ouvrages :

J.P. HATON, N. BOUZID et al, « Le raisonnement en intelligence artificielle », Interedition, 1991.

JM Allio et T.SCHIEX, « Intelligence artificielle et informatique théorique », Cepadues edition,1994.

A.THAYSE et al, « Approche Logique de l'intelligence artificielle, n°2 : de la logique modale à la logique des bases de données », Dunod Informatique, 1988.

E.AUDUREAU, P.ENJALBERT, L.FARINAS del CERRO, « Logique temporelle, sémantique et validation de programmes parallèles », Masson, collection étude et recherche en informatique, 1990.

H. BESTOUGEFF et G.LIGOZAT, « Outils Logiques pour le traitement du temps, de la linguistique à l'intelligence artificielle », Masson, collection étude et recherche en informatique, 1989.

R.TURNER, MASSON, « Logiques pour l'Intelligence Artificielle », 1986.

Articles :

J.F. ALLEN, « Maintaining Knowledge about Temporal Intervals », Communication of the ACM, , Vol 26, n°11, pp. 832-843, November 1983.

J.F. ALLEN, « Toward a General Theory of Action and Time », Artificial Intelligence 23, pp 123-154, 1984.

J.F. ALLEN, « An Interval-Based Representation of Temporal Knowledge », Proc 7Th IJCAI, Vancouver, B.C., August 1981 .

J.F. ALLEN, « Planning Using a Temporal World Model », Proc. 8th IJCAI, Karlsruhe, pp.741-747, 1983.

J. F ; ALLEN, « A Common Sense Theory of Time », Proc 9th IJCAI, Los Angeles, pp.528-831, 1985.

Mc DERMOTT, « A Temporal Logic for Reasonning About Processes and Plans », Cognitive Science 6, pp.101-155, (1982)

Mc DERMOT, « Planning : What it is, What it could be, An introduction to the Special Issue on Planning and Scheduling », Artificial Intelligence, 76, pp 1-16, (1995).

M.FISHER, R. OWENS, « An introduction to Executable Modal and Temporal Logics », Artificial Intelligence, p.1-20.