

# Cours 6

## Domaines d'application de la logique

Logique – Licence Informatique



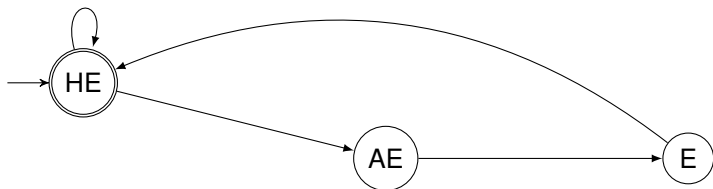
# Vérification de systèmes (model-checking)

- Problème : vérifier une propriété sur un système
- Méthode :
  - ▶ avoir un modèle du système (ensemble des exécutions possibles)
  - ▶ avoir une représentation formelle de la propriété (1 formule de logique)
  - ▶ vérifier si le modèle satisfait ou non la propriété (le modèle est-il une structure dans laquelle la propriété est satisfaite ?)
  - ▶ Réponses possibles :
    - ★ le système satisfait la propriété
    - ★ exhibition d'une séquence d'exécution responsable de la non-satisfaction de la propriété

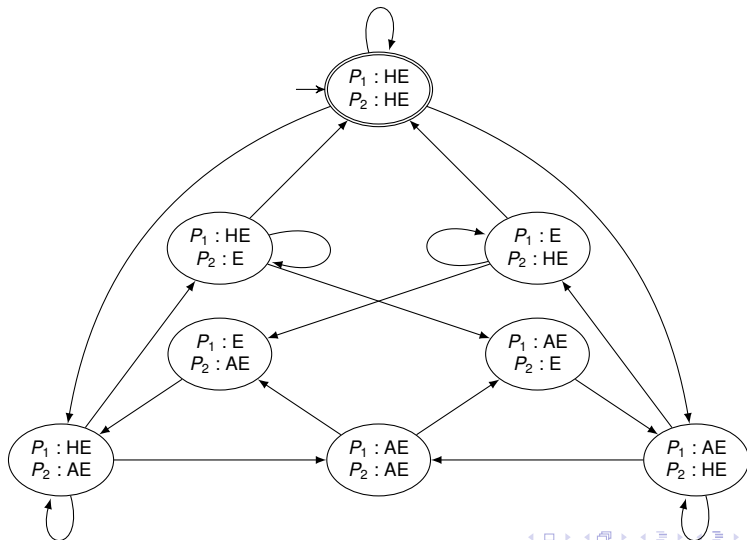
# Vérification de systèmes : modèle d'un système

- 2 processus  $P_1$  et  $P_2$  accèdent en écriture à une ressource partagée
- états possibles pour chaque processus
  - ▶ hors\_écriture ( $HE$ ) : ne cherche pas à accéder à la ressource
  - ▶ attente\_écriture ( $AE$ ) : demande à accéder à la ressource
  - ▶ écriture ( $E$ ) : accède à la ressource

Modèle d'un des deux processus  $P_1$  ou  $P_2$



# Modèle du système complet



# Structure associée au modèle

- $|\mathbf{M}| = \{\text{ensemble des états possibles d'un des composants du système}\}$
- $\mathcal{F}_0 = |\mathbf{M}|$
- prédicat *accessible*,  $accessible^{\mathbf{M}} = \{\text{ensemble des états accessibles du système}\}$  (prédicat  $n_{aire}$ ,  $n$  = nombre de composants du système)
- prédicat *eq* qui permet de faire un test d'égalité entre une variable et un élément de  $|\mathbf{M}|$

Sur notre exemple

- $|\mathbf{M}| = \{\text{HE, AE, E}\},$
- $accessible^{\mathbf{M}} = \{(\text{HE, HE}), (\text{HE, AE}), (\text{AE, HE}), (\text{AE, AE}), (\text{E, HE}), (\text{HE, E}), (\text{E, AE}), (\text{AE, E})\}$

# Exemples de propriétés

- l'accès en écriture à la ressource partagée se fait en exclusion mutuelle

$$F1 = \forall x \forall y (accessible(x, y) \Rightarrow \neg(eq(x, E) \wedge eq(y, E)))$$

- chaque processus a la possibilité d'écrire

$$F2 = (\exists x \exists y accessible(x, y) \wedge eq(x, E)) \wedge (\exists x \exists y accessible(x, y) \wedge eq(y, E))$$

Le système satisfait les propriétés si et seulement si

- $[F1]^M=1$  (satisfaite,  $(E, E) \notin accessible^M$ )
- $[F2]^M=1$  (satisfaite,
  - ▶  $(E, HE)$  (ou  $(E, AE)$ )  $\in accessible^M$  et
  - ▶  $(HE, E)$  (ou  $(AE, E)$ )  $\in accessible^M$ )

# Propriétés sur les séquences

- Propriétés qui expriment des conditions sur les séquences (enchaînement des états accessibles)
- Exemple : tout processus qui est en attente d'écriture finira par écrire
- Ensemble des états accessibles insuffisant
- Solutions
  - ▶ Ajouter des prédicats pour représenter le graphe des états accessibles
  - ▶ Logiques dédiées : logiques temporelles

# Logiques temporelles - connecteurs

- connecteurs de la logique des prédicats du premier ordre sans variables
- nouveaux connecteurs
  - ▶  $X$  (pour successeur)
  - ▶  $U$  (pour jusqu'à)
  - ▶  $G$  (pour toujours)
  - ▶  $F$  (pour finalement)



# Logiques temporelles - sémantique

## Propriétés vérifiées

- depuis un état  $s_0$  (état initial habituellement)
- sur une séquence d'exécution infinie  $seq = s_0, s_1, \dots, s_n, \dots$  (pour tout  $i > 0$ ,  $s_i$  état successeur de  $s_{i-1}$ )
- $[f]_{s_0, seq}^{\mathbf{M}} = 1$  signifie que la formule  $f$  est satisfaite par la structure  $\mathbf{M}$  (modèle du système) depuis l'état  $s_0$  sur la séquence  $seq$
- $[Xf]_{s_0, seq}^{\mathbf{M}} = 1$  si et seulement si  $[f]_{s_1, seq}^{\mathbf{M}} = 1$
- $[f_1 U f_2]_{s_0, seq}^{\mathbf{M}} = 1$  si et seulement si il existe  $i$  tel que  $[f_2]_{s_i, seq}^{\mathbf{M}} = 1$  et pour tout  $j, 0 \leq j < i, [f_1]_{s_j, seq}^{\mathbf{M}} = 1$
- $[Gf]_{s_0, seq}^{\mathbf{M}} = 1$  si et seulement si pour tout  $i \geq 0, [f]_{s_i, seq}^{\mathbf{M}} = 1$
- $[Ff]_{s_0, seq}^{\mathbf{M}} = 1$  si et seulement si il existe  $i \geq 0, [f]_{s_i, seq}^{\mathbf{M}} = 1$

# Logique temporelle linéaire (LTL)

- $|M|$  : ensemble des exécutions linéaires possibles (on perd la structure de graphe)
- formule  $f$  satisfaite par le système si et seulement si pour toute séquence  $seq$ ,  $[f]_{s_0, seq}^M = 1$
- $[eq(P_1, x)]_{s_i, seq}^M = 1$  si et seulement si, dans l'état  $s_i$  du système, le processus  $P_1$  est dans l'état  $x$  (idem pour  $P_2$ )
- $F1 = G(\neg(eq(P_1, E) \wedge eq(P_2, E)))$
- $F2$  ne s'exprime pas en LTL
- tout processus qui est en attente d'écriture **finira** par écrire  
 $G((eq(P_1, AE) \Rightarrow (F(eq(P_1, E)))) \wedge (eq(P_2, AE) \Rightarrow (F(eq(P_2, E)))))$
- propriété non satisfaite par notre système  
 $(HE, HE) \rightarrow (HE, AE) \rightarrow (AE, AE) \rightarrow (E, AE) \rightarrow (HE, AE)$  (on boucle)

# Logique temporelle arborescente (CTL)

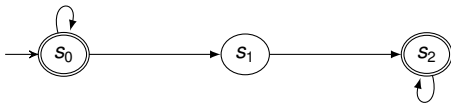
- **|M|** : ensemble des exécutions possibles (exécutions non pré-calculées, toujours la possibilité de choisir entre plusieurs successeurs, on garde la structure du graphe)
- quantification des connecteurs temporels
  - ▶  $\forall$ , formule doit être satisfaite par toutes les séquences possibles
  - ▶  $\exists$ , formule doit être satisfaite par au moins une séquence possible
- $F1 = \forall G(\neg(eq(P_1, E) \wedge eq(P_2, E)))$
- $F2 = \exists F(eq(P_1, E)) \wedge \exists F(eq(P_2, E))$
- tout processus qui est en attente d'écriture **finira** par écrire (toujours non satisfaite)  
 $\forall G((eq(P_1, AE) \Rightarrow (\forall F(eq(P_1, E)))) \wedge (eq(P_2, AE) \Rightarrow (\forall F(eq(P_2, E)))))$

# CTL $\not\subseteq$ LTL

- tout processus qui est en attente d'écriture **aura la possibilité** d'écrire  
 $\forall G((eq(P_1, AE) \Rightarrow (\exists F(eq(P_1, E)))) \wedge (eq(P_2, AE) \Rightarrow (\exists F(eq(P_2, E)))))$
- formule satisfaite  
pour  $P_1$  : (AE,E)->(AE,HE)->(AE,AE)->(E,AE)  
sous-séquence qui contient tous les états pour lesquels  $eq(P_1, AE)$  qui mènent tous à un état pour lequel  $eq(P_1, E)$   
pour  $P_2$  : (E,AE)->(HE,AE)->(AE,AE)->(AE,E)  
sous-séquence qui contient tous les états pour lesquels  $eq(P_2, AE)$  qui mènent tous à un état pour lequel  $eq(P_2, E)$
- ne s'exprime pas en logique temporelle linéaire (ne permet pas de prendre en compte une seule exécution)

# LTL $\not\subseteq$ CTL

Considérons le graphe suivant dans lequel la propriété  $p$  est satisfaite par les états  $s_0$  et  $s_2$



- toute séquence finira par toujours satisfaire  $p$  :  $FGp$
- propriété satisfaite car on ne considère que les séquences infinies : soit on boucle infiniment sur  $s_0$  soit on finit par boucler infiniment sur  $s_2$
- ne s'exprime pas en logique temporelle arborescente :
  - ▶  $\forall F \exists Gp$  (trop faible) : satisfaite mais ne prend pas en compte toutes les séquences
  - ▶  $\forall F \forall Gp$  (trop fort) : non satisfaite car depuis toute séquence qui boucle sur  $s_0$  on peut toujours aller en  $s_1$  qui ne satisfait pas  $p$