

Quatrième partie

LTL – logique temporelle linéaire



Plan

- 1 Logiques temporelles
- 2 Logique temporelle linéaire – LTL
 - Syntaxe
 - Sémantique
 - Réduction
- 3 Expressivité
 - Exemples
 - Propriétés classiques



Logiques temporelles



Objectif

Exprimer des **propriétés** portant sur les **exécutions** des systèmes.

Spécification non opérationnelle : pas de relation de transition explicite, pas de notion d'états initiaux.

Une logique est définie par :

- une syntaxe : opérateurs de logique classique plus des opérateurs temporels pour parler du futur et du passé.
- une sémantique : domaine des objets (appelés modèles) sur lesquels on va tester la validité des formules, plus l'interprétation des opérateurs.



Plan

- 1 Logiques temporelles
- 2 **Logique temporelle linéaire – LTL**
 - Syntaxe
 - Sémantique
 - Réduction
- 3 Expressivité
 - Exemples
 - Propriétés classiques



Linear Temporal Logic



Modèles

Une formule LTL se rapporte toujours à une **trace** donnée σ d'un système : les traces constituent les modèles de cette logique.

Note : plutôt que d'*état*, on parle souvent d'*instant* pour désigner les éléments d'une trace.

Rappel : pour un ST $\langle S, I, R \rangle$, une trace est une séquence $\sigma \in S^* \cup S^\omega$, tel que pour tout s_i, s_{i+1} consécutifs, $(s_i, s_{i+1}) \in R$.



Syntaxe de la LTL

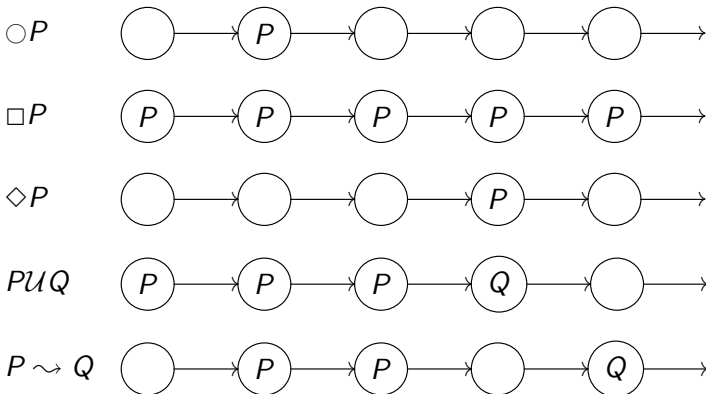


formule	nom	interprétation
s		le premier état de la trace est s
$\neg P$		
$P \vee Q$		
$P \wedge Q$		
$\bigcirc P$	<i>next</i>	P est vrai à l'instant suivant
$\Box P$	<i>always</i>	P est toujours vrai i.e. à tout instant à partir de l'instant courant
$\Diamond P$	<i>eventually</i>	P sera vrai (dans le futur)
$P \mathcal{U} Q$	<i>until</i>	Q sera vrai, et en attendant P reste vrai
$P \leadsto Q$	<i>leadsto</i>	quand P est vrai, alors Q est vrai plus tard

Dans les approches symboliques, l'opérateur \bigcirc représentant l'instant suivant peut être remplacé par des variables primées qui représentent la valeur des variables du système dans l'état suivant.



Intuition sémantique



Opérateurs minimaux



Les opérateurs minimaux sont $\bigcirc P$ et $P \mathcal{U} Q$:

- $\diamond P \stackrel{\Delta}{=} \text{True } \mathcal{U} P$
- $\square P \stackrel{\Delta}{=} \neg \diamond \neg P$
- $P \leadsto Q \stackrel{\Delta}{=} \square(P \Rightarrow \diamond Q)$



Syntaxe alternative



Syntaxe alternative

On trouve fréquemment une autre syntaxe :

- $\Box \leftrightarrow G$ (*globally*)
- $\Diamond \leftrightarrow F$ (*finally*)
- $\bigcirc \leftrightarrow X$ (*next*)

Opérateurs complémentaires

- Opérateur *waiting-for* (ou *unless* ou *weak-until*)
 $PWQ \triangleq \Box P \vee PUQ$
 Q finit peut-être par être vrai et en attendant P reste vrai
- Opérateur *release*
 $PRQ \triangleq QU(P \wedge Q)$
 Q reste vrai jusqu'à ce que P le devienne.



Opérateurs du passé

formule	nom	interprétation
$\ominus P$	<i>previously</i>	P est vrai dans l'instant précédent
$\boxminus P$	<i>has-always-been</i>	P a toujours été vrai jusqu'à l'instant courant
$\Diamond P$	<i>once</i>	P a été vrai dans le passé
PSQ	<i>since</i>	Q a été vrai dans le passé et P est resté vrai depuis la dernière occurrence de Q
PBQ	<i>back-to</i>	P est vrai depuis la dernière occurrence de Q , ou depuis l'instant initial si Q n'a jamais été vrai

Peu utilisés en pratique.



Sémantique (système)



On note (σ, i) pour le suffixe $\langle s_i \rightarrow s_{i+1} \rightarrow \dots \rangle$ d'une trace $\sigma = \langle s_0 \rightarrow s_1 \rightarrow \dots \rangle$.

Vérification par un système

Un système \mathcal{S} vérifie (valide) la formule F ssi toutes les exécutions de \mathcal{S} la valident à partir de l'instant initial :

$$\frac{\forall \sigma \in Exec(\mathcal{S}) : (\sigma, 0) \models F}{\mathcal{S} \models F}$$

Rappel : les exécutions d'un système sont ses traces finies maximales et infinies, et qui débutent par un état initial.



Sémantique (opérateurs logiques)

Sémantique standard des opérateurs logiques

$$\frac{(\sigma, i) \models P \quad (\sigma, i) \models Q}{(\sigma, i) \models P \wedge Q}$$

$$\frac{(\sigma, i) \models P}{(\sigma, i) \models P \vee Q} \quad \frac{(\sigma, i) \models Q}{(\sigma, i) \models P \vee Q}$$

$$\frac{\neg (\sigma, i) \models P}{(\sigma, i) \models \neg P}$$



Sémantique (opérateurs temporels)



$$\frac{\sigma_i = s}{(\sigma, i) \models s}$$

$$\frac{(\sigma, i + 1) \models P}{(\sigma, i) \models \bigcirc P}$$

$$\frac{\exists k \geq 0 : (\sigma, i + k) \models Q \wedge \forall k', 0 \leq k' < k : (\sigma, i + k') \models P}{(\sigma, i) \models P \mathcal{U} Q}$$



Sémantique (opérateurs temporels dérivés)



$$\frac{\exists k \geq 0 : (\sigma, i + k) \models P}{(\sigma, i) \models \Diamond P}$$

$$\frac{\forall k \geq 0 : (\sigma, i + k) \models P}{(\sigma, i) \models \Box P}$$

$$\frac{\forall k \geq 0 : ((\sigma, i + k) \models P \Rightarrow \exists k' \geq k : (\sigma, i + k') \models Q)}{(\sigma, i) \models P \leadsto Q}$$



Réduction à la logique pure



- La logique temporelle linéaire possède une expressivité telle qu'elle peut représenter exactement n'importe quelle spécification opérationnelle décrite en termes de système de transitions, d'où :
- vérifier qu'un système de transitions \mathcal{M} possède la propriété temporelle F_{Spec} :

$$\mathcal{M} \models F_{Spec}$$

- revient à déterminer la validité de :

$$F_{\mathcal{M}} \Rightarrow F_{Spec}$$

où $F_{\mathcal{M}}$ est une formule représentant exactement les exécutions du modèle \mathcal{M} (i.e. ses états initiaux, ses transitions, ses contraintes d'équité).

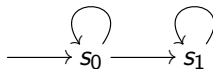


Plan

- 1 Logiques temporelles
- 2 Logique temporelle linéaire – LTL
 - Syntaxe
 - Sémantique
 - Réduction
- 3 **Expressivité**
 - Exemples
 - Propriétés classiques

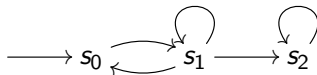


Exemple 1



	pas d'équité	équité faible (s_0, s_1)
$s_0 \wedge \bigcirc s_0$		
$s_0 \wedge \bigcirc (s_0 \vee s_1)$		
$\Box (s_0 \Rightarrow \bigcirc s_0)$		
$\Box (s_0 \Rightarrow \bigcirc (s_0 \vee s_1))$		
$\Box (s_1 \Rightarrow \bigcirc s_1)$		
$\Diamond (s_0 \wedge \bigcirc s_1)$		
$\Box s_0$		
$\Diamond \neg s_0$		
$\Diamond \Box s_1$		
$s_0 \mathcal{W} s_1$		
$s_0 \mathcal{U} s_1$		

Exemple 2



	pas d'équité	faible (s_1, s_2)	forte (s_1, s_2)
$\Box \Diamond \neg s_1$			
$\Box (s_1 \Rightarrow \Diamond s_2)$			
$\Diamond \Box (s_1 \vee s_2)$			
$\Box (s_1 \mathcal{U} s_2)$			
$\Box (s_0 \Rightarrow s_0 \mathcal{U} s_1)$			
$\Box (s_0 \mathcal{U} (s_1 \vee s_2))$			
$\Box (s_1 \Rightarrow s_1 \mathcal{U} s_2)$			
$\Diamond (s_1 \mathcal{U} s_2)$			
$\Diamond (s_1 \mathcal{W} s_2)$			
$\Box \Diamond (s_1 \mathcal{U} (s_0 \vee s_2))$			

Sûreté/vivacité – *Safety/Liveness*

On qualifie de

- **Sûreté** : rien de mauvais ne se produit
= propriété qui s'invalidé sur un préfixe fini d'une exécution :
 $\Box P, \Box(P \Rightarrow \Box P), P\mathcal{W}Q \dots$
- **Vivacité** : quelque chose de bon finit par se produire
= propriété qui peut toujours être validée en étendant le préfixe d'une exécution :
 $\Diamond P, P \leadsto Q \dots$
- Certaines propriétés combinent vivacité et sûreté :
 $PUQ, \Box P \wedge \Diamond Q \dots$
 - Réponse : $\Box \Diamond P$
 - Persistance : $\Diamond \Box P$



Invariance, stabilité

Invariance

Spécifier un sur-ensemble des états accessibles d'un système :

$$\mathcal{S} \models \Box P$$

où P est un prédicat d'état.

Stabilité

Spécifier la stabilité d'une situation si elle survient :

$$\mathcal{S} \models \Box(P \Rightarrow \Box P)$$

où P est un prédicat d'état.



Possibilité



Possibilité

Spécifier qu'il est possible d'atteindre un certain état vérifiant P dans une certaine exécution :

Impossible pour P arbitraire, mais pour P un prédicat d'état :

$$\mathcal{S} \not\models \Box \neg P$$

Attention à la négation : $\neg \Box P = \Diamond \neg P$ mais $\mathcal{S} \not\models \Box P \not\Rightarrow \mathcal{S} \models \Diamond \neg P$



Négation



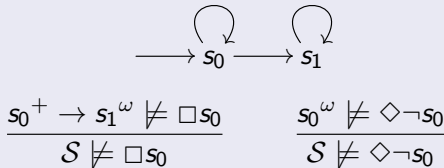
Négation : danger !

Pour σ exécution : $\sigma \models \neg P \equiv \sigma \not\models P$

Pour \mathcal{S} système : $\mathcal{S} \models \neg P \Rightarrow \mathcal{S} \not\models P$ mais pas l'inverse !

$\mathcal{S} \not\models Q$ signifie qu'il existe **au moins une** exécution qui invalide Q (= qui valide $\neg Q$), mais pas que toutes les exécutions le font.

En LTL, on peut avoir $\mathcal{S} \not\models Q \wedge \mathcal{S} \not\models \neg Q$:



Combinaisons

Infiniment souvent – Réponse

Spécifier que P est infiniment souvent vrai dans toute exécution :

$$S \models \Box \Diamond P$$

Finalement toujours – Persistance

Spécifier que P finit par rester définitivement vrai :

$$S \models \Diamond \Box P$$

Note : $\Box \Box P = \Box P$ et $\Diamond \Diamond P = \Diamond P$



Client/serveur

Réponse

Spécifier qu'un système (jouant le rôle d'un serveur) répond toujours (Q) à une requête donnée (P) :

$$\mathcal{S} \models \Box(P \Rightarrow \Diamond Q)$$

Souvent nommé leads-to :

$$\mathcal{S} \models P \leadsto Q$$

Stabilité d'une requête

Spécifier que la requête P d'un système (jouant le rôle d'un client) est stable tant qu'il n'y a pas de réponse favorable Q :

$$\mathcal{S} \models \Box(P \Rightarrow P\mathcal{W}Q)$$



Équité des transitions – *Fairness*



Rappel informel :

faible : continûment faisable \rightarrow infiniment souvent fait

forte : infiniment souvent faisable \rightarrow infiniment souvent fait

Équité faible des transitions

Soit $r \subseteq R$. Les transitions r sont en équité faible dans \mathcal{S} :

$$\mathcal{S} \models \Diamond \Box \text{dom}(r) \Rightarrow \Box \Diamond r$$

$$\mathcal{S} \models \Box \Diamond \neg \text{dom}(r) \vee \Box \Diamond r$$

Équité forte des transitions

Soit $r \subseteq R$. Les transitions r sont en équité forte dans \mathcal{S} :

$$\mathcal{S} \models \Box \Diamond \text{dom}(r) \Rightarrow \Box \Diamond r$$

$$\mathcal{S} \models \Diamond \Box \neg \text{dom}(r) \vee \Box \Diamond r$$

(une transition $s \rightarrow s'$ est équivalente à $s \wedge \bigcirc s'$, et un ensemble de transition $\{t_1, t_2, \dots\}$ est équivalent à $t_1 \vee t_2 \vee \dots$)



Spécification d'un système de transitions

Si on utilise une description en intention, et si l'on remplace l'utilisation de l'opérateur \bigcirc par les variables primées, alors on peut spécifier toutes les exécutions permises par un système $\langle S, I, R \rangle$:

$$S \models I \wedge \Box R$$

L'utilisation de variables primées n'est pas nécessaire mais simplifie les formules.

Par exemple $P(x, x')$ est équivalent à la formule :

$$\forall v : x = v \Rightarrow \bigcirc P(v, x)$$

qui nécessite une quantification sur une variable.



Limites de l'expressivité

Tout n'est pas exprimable en LTL :

- Possibilité arbitraire : si P devient vrai, il est toujours possible (mais pas nécessaire) que Q le devienne après.
- Accessibilité d'un état : depuis l'état initial, il est possible d'atteindre cet état.
- Réinitialisabilité : quelque soit l'état, il est possible de revenir dans un des états initiaux.

(ces propriétés sont exprimables en Computational Tree Logic (CTL), à venir)



Conclusion



La logique temporelle linéaire (LTL) permet d'exprimer, abstraitement, des propriétés sur les exécutions d'un système

Logiques modales

La LTL est un cas particulier de logique modale.

Autres interprétations :

- \Box = nécessité, \Diamond = possibilité
- logique de la croyance : « je crois que P est vrai »
- logique épistémique : « X sait que P »
- logique déontique : « P est obligatoire/interdit/permis »
- ...

