Sixième partie

CTL – logique temporelle arborescente



Systèmes de transitions 1 / 31

Plan

- 1 CTL
 - Syntaxe
 - Sémantique
- 2 Expressivité
 - Exemples
 - Propriétés classiques

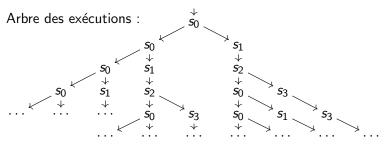


Ensemble des exécutions vs arbre des exécutions

Soit le système de transitions : $\longrightarrow s_0 \xrightarrow{s_0} s_1$

Ensemble des exécutions :

$$\langle (s_0^+ \to s_1 \to s_2)^* \to s_0^\omega \rangle, \langle (s_0^+ \to s_1 \to s_2)^\omega \rangle, \langle (s_0^+ \to s_1 \to s_2)^+ \to s_3^\omega \rangle$$
ou
$$\begin{cases} s_0 \to s_0 \to \cdots, s_0 \to s_1 \to s_2 \to s_0 \to \cdots, \\ s_0 \to s_1 \to s_2 \to s_0 \to \cdots, s_0 \to s_1 \to s_2 \to s_0 \to \cdots, \\ s_0 \to s_1 \to s_2 \to s_3 \to s_3 \to \cdots, \dots \end{cases}$$



Modèles

Une formule CTL se rapporte toujours à un état donné s d'un système, duquel partent des traces Traces(s). Les états de S constituent les modèles de cette logique.

La différence (syntaxiquement parlant) avec LTL réside dans l'apparition dans les opérateurs temporels de quantificateurs de traces.



Quantification universelle

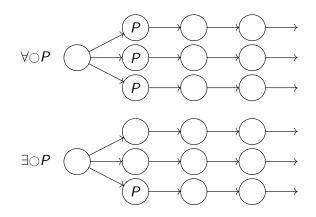
•			
formule	e interprétation (pour s un état)		
	pour toute trace partant de s		
$\forall \bigcirc P$	P est vrai à l'instant suivant		
$\forall \Box P$	P est toujours vrai à chaque état		
$\forall \Diamond P$	$\forall \Diamond P$ P finit par être vrai (dans le futur)		
$P \forall \mathcal{U} Q \mid Q$ finit par être vrai, et en attendant P res			

Quantification existentielle

U	Qualitation of the control of the co			
Ī	formule	interprétation (pour s un état)		
		pour au moins une trace partant de s		
	$\exists \bigcirc P$	P est vrai à l'instant suivant		
	$\exists \Box P$	P est toujours vrai à chaque état		
	∃◇P	P finit par être vrai (dans le futur)		
	$P \exists \mathcal{U} Q \mid Q$ finit par être vrai, et en attendant P reste vrai			

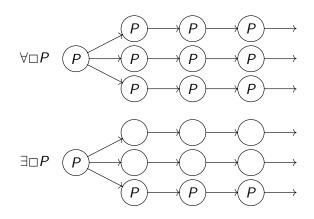


Intuition sémantique $\forall \bigcirc$, $\exists \bigcirc$



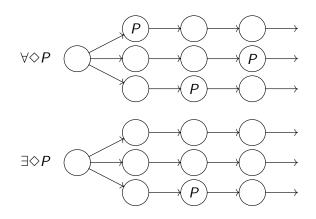


Intuition sémantique ∀□, ∃□



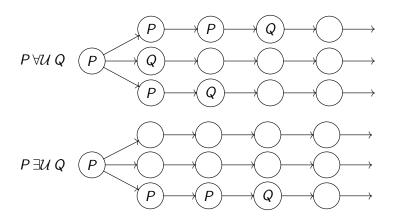


Intuition sémantique ∀♦, ∃♦





Intuition sémantique $\forall \mathcal{U}, \; \exists \mathcal{U}$





Opérateurs minimaux

Un ensemble d'opérateurs minimaux est $\forall \bigcirc$, $\forall \mathcal{U}$ et $\exists \mathcal{U}$:

- $\exists \cap P \triangleq \neg \forall \cap \neg P$
- $\forall \Diamond P \triangleq True \forall \mathcal{U} P$
- $\exists \Diamond P \triangleq True \exists \mathcal{U} P$
- $\forall \Box P \triangleq \neg \exists \Diamond \neg P$
- $\bullet \ \exists \Box P \triangleq \neg \forall \Diamond \neg P$

(autres ensembles minimaux : $\{\exists\bigcirc,\exists\Box,\exists\mathcal{U}\}\$ ou $\{\forall\diamondsuit,\exists\mathcal{U},\exists\bigcirc\}\$)



Syntaxe alternative

Syntaxe alternative

On trouve très fréquemment une autre syntaxe :

$$\forall \longleftrightarrow A (all)$$

$$\exists \leftrightarrow \mathsf{E} \ (\mathsf{exists})$$

$$\Box \qquad \leftrightarrow \mathsf{G} \; \mathsf{(globally)}$$

$$\Diamond \longleftrightarrow \mathsf{F} \text{ (finally)}$$

$$\bigcirc$$
 \leftrightarrow X (next)

Par exemple:

$$\forall \Box \exists \Diamond P \leftrightarrow \mathsf{AG} \; \mathsf{EF} \; \mathsf{P}$$

$$f \forall \mathcal{U} g \leftrightarrow \mathsf{A}(\mathsf{f} \mathsf{U} \mathsf{g})$$

Opérateur complémentaire waiting-for

$$P \ni W Q \triangleq \exists \Box P \lor P \ni U Q$$

$$P \forall W Q \not\triangleq \forall \Box P \lor P \forall U Q - trop fort$$
$$\triangleq \neg (\neg Q \exists U (\neg P \land \neg Q))$$

$$\stackrel{\triangle}{=} \neg (\neg Q \exists \mathcal{U} (\neg P \land \neg Q))$$

La relation de validation sémantique ne fait intervenir que l'état courant.

Vérification par un système

Un système $S = \langle S, \{s_0\}, R \rangle$ vérifie (valide) la formule F ssi l'état initial de S la valide :

$$\frac{s_0 \models F}{S \models F}$$

(la sémantique est moins claire s'il y a plusieurs états initiaux, du fait de l'opérateur \exists : pour tous les états initiaux, ou pour au moins un? En pratique, on peut toujours se ramener à un seul état initial, donc on évite la difficulté)



$$\overline{s \models s}$$

$$\frac{s \models P \quad s \models Q}{s \models P \land Q}$$

$$\frac{s \models P}{s \models P \lor Q} \quad \frac{s \models Q}{s \models P \lor Q}$$

$$\frac{s \models P}{s \not\models \neg P}$$



Sémantique (opérateurs temporels)

(rappel : pour une trace σ , σ_i est le i-ième élément de σ en commençant à 0, et pour un état s, Traces(s) est l'ensemble des traces issues de s)

$$\frac{\forall \sigma \in \mathit{Traces}(s) : \sigma_1 \models P}{s \models \forall \bigcirc P}$$

$$\frac{\forall \sigma \in \mathit{Traces}(s) : \exists j \geq 0 : \sigma_j \models Q \land \forall i < j : \sigma_i \models P}{s \models P \forall \mathcal{U} Q}$$

$$\frac{\exists \sigma \in \mathit{Traces}(s) : \exists j \geq 0 : \sigma_j \models Q \land \forall i < j : \sigma_i \models P}{s \models P \exists \mathcal{U} Q}$$



Sémantique (opérateurs temporels dérivés)

$$\frac{\exists \sigma \in \mathit{Traces}(s) : \sigma_1 \models P}{s \models \exists \bigcirc P}$$

$$\frac{\forall \sigma \in \mathit{Traces}(s) : \forall i \geq 0 : \sigma_i \models P}{s \models \forall \Box P}$$

$$\frac{\exists \sigma \in \mathit{Traces}(s) : \forall i \geq 0 : \sigma_i \models P}{s \models \exists \Box P}$$

$$\frac{\forall \sigma \in \mathit{Traces}(s) : \exists i \geq 0 : \sigma_i \models P}{s \models \forall \Diamond P}$$
$$\frac{\exists \sigma \in \mathit{Traces}(s) : \exists i \geq 0 : \sigma_i \models P}{s \models \exists \Diamond P}$$



Négation

Contrairement à LTL, pour toute propriété CTL, on a : soit $\mathcal{S} \models F$, soit $\mathcal{S} \models \neg F$, et $\mathcal{S} \not\models F \equiv \mathcal{S} \models \neg F$.

Négation des formules \forall , \exists , \Box , \diamondsuit

La négation d'une formule à base de \forall , \exists , \Box , \diamondsuit se fait simplement en inversant chaque opérateur pour son dual.

exemples:

$$\neg(\forall \Diamond \exists \Box p) = \exists \Box \forall \Diamond \neg p$$
$$(\forall \Diamond \neg s_0 \Rightarrow \forall \Diamond s_3) = (\exists \Box s_0 \lor \forall \Diamond s_3) \text{ car } (p \Rightarrow q) = (\neg p \lor q)$$



Définition par point-fixe

Une fois définis $\exists \bigcirc$ et $\forall \bigcirc$, chaque opérateur est le plus petit point fixe de sa définition inductive :

$$\forall \Box f = f \land \forall \Box \forall \Box f
\exists \Box f = f \land \exists \Box \exists \Box f
\forall \Diamond f = f \lor \forall \Box \forall \Diamond \Box f
\exists \Diamond f = f \lor \exists \Box \exists \Diamond \Box f
f \forall \mathcal{U} g = g \lor (f \land \forall \Box (f \forall \mathcal{U} g))
f \exists \mathcal{U} g = g \lor (f \land \exists \Box (f \exists \mathcal{U} g))$$

(surtout utile pour l'implantation d'un vérificateur de modèles)



Plan

- 1 CTL
 - Syntaxe
 - Sémantique
- 2 Expressivité
 - Exemples
 - Propriétés classiques



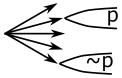
Exemples amusants

• $\exists \Box \ \forall \bigcirc \ p$: une exécution avec une "enveloppe" qui vérifie p



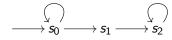
• $\exists \bigcirc \forall \Box p \land \exists \bigcirc \forall \Box \neg p$ un état successeur à partir duquel p est toujours et partout vrai,

et un état successeur à partir duquel $\neg p$ est toujours et partout vrai





Exemple



	pas d'équité	équité faible (s_0, s_1)
$s_0 \land \forall \bigcirc s_0$		
$s_0 \land \exists \bigcirc s_0$		
$\forall \Box (s_0 \Rightarrow \exists \bigcirc s_0)$		
$\forall \Box (s_0 \Rightarrow \exists \Diamond s_2)$		
$\forall \Box (s_0 \Rightarrow \forall \Diamond s_2)$		
$\exists \Diamond \neg s_0$		
$\forall \Diamond \neg s_0$		
$\forall \Box \exists \Diamond s_2$		
$\forall \Box \forall \Diamond s_2$		
$\forall \Diamond \exists \Diamond s_1$		
$\forall \Box \exists \Diamond s_1$		

77

Exemple

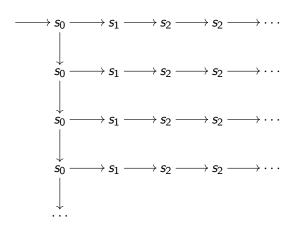
$$\longrightarrow \stackrel{\nwarrow}{s_0} \longrightarrow s_1 \longrightarrow \stackrel{\nwarrow}{s_2}$$

	pas d'équité	équité faible (s_0, s_1)	
$s_0 \land \forall \bigcirc s_0$	n	n	
$s_0 \land \exists \bigcirc s_0$	0	0	
$\forall \Box (s_0 \Rightarrow \exists \bigcirc s_0)$	0	0	ותת
$\forall \Box (s_0 \Rightarrow \exists \Diamond s_2)$	0	0	ווו
$\forall \Box (s_0 \Rightarrow \forall \Diamond s_2)$	n	0	ותת
$\exists \Diamond \neg s_0$	0	0	
$\forall \Diamond \neg s_0$	n	0	
$\forall \Box \exists \Diamond s_2$	0	0	נעע
$\forall \Box \forall \Diamond s_2$	n	0	נעע
$\forall \Diamond \exists \Diamond s_1$	0	0	נעע
$\forall \Box \exists \Diamond s_1$	n	n	111



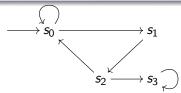
Exemple - Arbre des exécutions

Arbre des exécutions du système de transition précédent





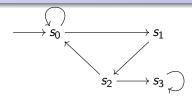
Exemple 2



	pas d'équité	faible (s_0, s_1)	forte (s_2, s_3)	forte (s_2, s_3) faible (s_0, s_1)
$\exists \Box s_0$				
$\forall \Box \exists \Diamond s_3$				
$\forall \Box \forall \Diamond s_3$				
$\forall \Diamond \forall \Box s_3$				
$\exists \Box s_0 \lor \forall \Diamond s_3$				
$\forall \Diamond \neg s_0 \Rightarrow \forall \Diamond s_3$				



Exemple 2



	pas d'équité	faible (s_0, s_1)	forte (s_2, s_3)	forte (s_2, s_3)
				faible (s_0, s_1)
$\exists \Box s_0$	0	n	0	n
$\forall \Box \exists \Diamond s_3$	0	0	0	0
$\forall \Box \forall \Diamond s_3$	n	n	n	0
$\forall \Diamond \forall \Box s_3$	n	n	n	0
$\exists \Box s_0 \lor \forall \Diamond s_3$	0	n	0	0
$\forall \Diamond \neg s_0 \Rightarrow \forall \Diamond s_3$	0	n	0	0

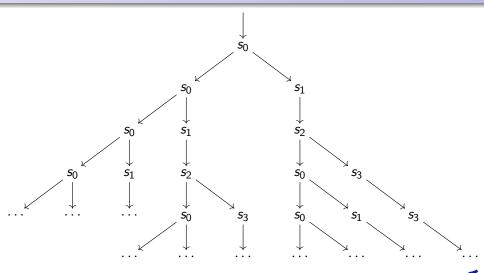
111 111

111

\ \ \ \

Contrairement à LTL, avec l'équité, on peut à la fois rendre vraie une propriété (p.e. contenant \diamondsuit ou \mathcal{U}) et rendre faux une propriété (p.e. contenant $\exists \cdot$).

Exemple 2 - Arbre des exécutions





Invariance, Possibilité

Invariance

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

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

où P est un prédicat d'état.

Stabilité

Spécifier la stabilité d'une situation si elle survient :

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

où P est un prédicat d'état.

Possibilité

Spécifier qu'il est possible d'atteindre un état vérifiant P:

$$\mathcal{S} \models \exists \Diamond P$$



Possibilité complexe

Séquence

Spécifier qu'un scénario d'exécution $\langle s_1 \to s_2 \to \ldots \to s_n \rangle$ est possible.

$$\mathcal{S} \models s_1 \land \exists \bigcirc (s_2 \land \ldots \land \exists \bigcirc (s_{n-1} \land \exists \bigcirc s_n) \ldots)$$

Réinitialisabilité

Spécifier que quelque soit l'état courant, il est possible de revenir dans un des états initiaux (définis par le prédicat *I*).

$$\mathcal{S} \models \forall \Box \exists \Diamond I$$

Possibilité arbitraire

Spécifier que si P devient vrai, il est toujours possible (mais pas nécessaire) que Q le devienne après.

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



Client/serveur

Réponse

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

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

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:

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



Infiniment souvent

Spécifier que P est infiniment souvent vrai dans toute exécution : $\mathcal{S}\models\forall\Box\forall\Diamond P$

Finalement toujours

Spécifier que P finit par rester définitivement vrai : impossible! $S \models \forall \Diamond \forall \Box P$ ne convient pas (trop fort)

Soit
$$S = \bigcirc$$

$$\longrightarrow s_0 \longrightarrow s_1 \longrightarrow s_2$$
en LTL : $S \models \Diamond \Box (s_0 \lor s_2)$
mais CTL : $S \not\models \forall \Diamond \forall \Box (s_0 \lor s_2)$
(tant qu'on est en s_0 , on $peut$ passer en $s_1 : S \models \forall \Diamond \exists \Diamond s_1$)

Note : $\mathcal{XXP} = \mathcal{XP}$ pour $\mathcal{X} \in \{ \forall \Box, \exists \Box, \forall \diamondsuit, \exists \diamondsuit \}$

Spécification d'un ST

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

$$\mathcal{S} \models I \land \forall \Box R$$

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

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

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

qui nécessite une quantification sur une variable.



Comparaison CTL vs. LTL

Contrairement à CTL, les opérateurs temporels LTL parlent tous de la même trace. Les combinaisons de connecteurs temporels ont parfois des sens (subtilement) différents.

	CTL	LTL
$\forall P$, nécessairement P ou $\neg P$	$\mathcal{S} \models P \lor \mathcal{S} \models \neg P$	$S \models P \lor S \models \neg P$
négation	$S \models \neg P \equiv S \not\models P$	$S = \neg P = S \not\equiv P$
l'un de <i>P</i> ou <i>Q</i> inévitable	$S \models \forall \Diamond P \forall \forall \Diamond Q$	$\mathcal{S} \models \Diamond P \lor \Diamond Q$
	$\mathcal{S} \models \forall \Diamond (P \lor Q)$	
l'un de P ou Q continu	$S \models \forall \Box (P \lor Q)$	$\mathcal{S} \models \Box P \lor \Box Q$
	S = YOP Y YOQ	
$\neg P$ transitoire	$S \models \forall \Diamond \forall \Box P$	$\mathcal{S}\models\Diamond\Box P$
possibilité	$S \models \exists \Diamond P$	S → P

Conséquence : l'équité n'est pas exprimable en CTL.

Néanmoins, on peut vérifier des propriétés CTL sur un ST comportant des contraintes d'équité.



777

Comparaison LTL vs. CTL

Linear Time Logic

- + Intuitive
- ...sauf la négation
- + Suffisante pour décrire un système de transition
- + y compris l'équité
- Vérification exponentielle en le nombre d'opérateurs temporels

Computational Tree Logic

- Expressivité parfois déroutante
- + Propriétés de possibilité (p.e. réinitialisabilité)
- + Suffisante pour décrire un système de transition
- ...sauf l'équité non exprimable (mais utilisable)
- + Vérification linéaire en le nombre d'opérateurs temporels



CTL* autorise tout mélange des quantificateurs de traces \forall , \exists et d'états \Box , \diamondsuit , \bigcirc , \mathcal{U} .

Exemple : $\exists ((\Box \Diamond P) \land (\Diamond Q)) = \text{il existe une exécution où } P \text{ est infiniment souvent vrai, et où } Q \text{ sera vrai.}$

CTL* est strictement plus expressif que CTL et LTL. L'usage pratique est rare (hors les fragments correspondant à CTL et LTL).

