Université de Sherbrooke
Département d'informatique

IGL501: Méthodes formelles en génie logiciel

1 Logique temporelle linéaire (LTL)

1.1 Syntaxe

Les éléments suivants sont des formules *atomiques* de la LTL :

- **true** et **false**;
- une variable propositionnelle;
- une formule atomique de la logique du premier ordre.

Une variable propositionnelle dénote soit un événement, soit une variable booléenne d'un programme ou d'une spécification. Une formule atomique de la logique du premier ordre porte typiquement sur les variables d'un programme ou d'une spécification. Ces variables déterminent alors l'état du système. Un événement avec des paramètres (par exemple, $e(x_1, \ldots, x_n)$) constitue aussi une formule atomique de la logique du premier ordre.

Des formules complexes sont construites avec les connecteurs de la logique propositionnelle et des connecteurs temporels : W (*until* faible), U (*until* fort), G (*globally*, aussi appelé *always* et noté $\Box$), F (*finally*, aussi appelé *eventually* et noté $\Diamond$), X (*next*, aussi noté $\bigcirc$). Soit p et q des formules LTL. Alors les expressions suivantes sont sont des formules LTL:

- *connecteurs propositionnels* : $\neg p$, $p \wedge q$, $p \vee q$, $p \Rightarrow q$, $p \Leftrightarrow q$,
- *connecteurs temporels* : $p \mathsf{W} q$, $p \mathsf{U} q$, $\mathsf{G}p$, $\mathsf{F}p$, $\mathsf{X}\ p$,
- *quantificateurs de la logique du premier ordre* : $\forall x \cdot p$, $\exists x \cdot p$.

1.2 Sémantique

Soit $A = (Q, L, T, q_0)$ un *automate* (aussi appelé *système de transition*) où

- Q est un ensemble d'états,
- L est un ensemble d'étiquettes (alphabet) qui dénote les événements observables du système,
- $T \subseteq Q \times L \times Q$ est un ensemble de transitions
- $q_0 \in Q$ est l'état initial.

Une exécution dans A est une séquence de transitions, possiblement infinie, de la forme

$$q_0 \xrightarrow{l_0} q_1 \xrightarrow{l_1} q_2 \xrightarrow{l_2} \ldots$$

où

$$\forall i \cdot i \geq 0 \Rightarrow q_i \xrightarrow{l_i} q_{i+1} \in T$$

De plus, si l'exécution est finie et si q_n dénote le dernier état atteint, alors

$$\{q_n\} \lhd T = \emptyset,$$

c'est-à-dire qu'une exécution finie termine par un état où plus aucune transition n'est possible. La sémantique de la LTL est définie sur les exécutions. On s'intéresse soit à la séquence d'états (i.e., $[q_0, q_1, q_2, \ldots]$) ou soit à la séquence d'événements (i.e., $[l_0, l_1, l_2, \ldots]$) d'une exécution, selon l'aspect auquel le spécifieur s'intéresse. Soit σ une séquence. L'expression $\mathsf{head}(\sigma)$ dénote le premier élément de σ; l'expression $\mathsf{tail}(\sigma)$ dénote σ amputée de son premier élément; l'expression $\sigma_1 \frown \sigma_2$ dénote la concaténation de la séquence finie σ_1 avec la séquence σ_2. On dénote par $\sigma \models p$ la satisfaction par σ de la formule LTL p. On définit $\sigma \models p$ comme suit.

- Si p est une formule atomique de la LTL, alors $\sigma \models p \;\triangleq\; \mathsf{head}(\sigma)$ satisfait p. La satisfaction de p par $\mathsf{head}(\sigma)$ dépend de la nature de p et de σ, car on s'intéresse soit aux étiquettes ou soit aux états.
- $\sigma \models p \,\mathsf{W}\, q \;\triangleq\; \sigma \models q \lor (\sigma \models p \land \mathsf{tail}(\sigma) \models p \,\mathsf{W}\, q)$
- $\sigma \models p \,\mathsf{U}\, q \;\triangleq\; \sigma \models (p \,\mathsf{W}\, q) \land (\exists\, \sigma_1, \sigma_2 \cdot \sigma = \sigma_1 \frown \sigma_2 \land \sigma_2 \models q)$
- $\sigma \models \mathsf{G}p \;\triangleq\; \sigma \models p \,\mathsf{W}\, \mathbf{false}$
- $\sigma \models \mathsf{F}p \;\triangleq\; \sigma \models \mathbf{true} \,\mathsf{U}\, p$
- $\sigma \models \mathsf{X}p \;\triangleq\; \mathsf{tail}(\sigma) \models p$
- $\sigma \models \neg p \;\triangleq\; \neg(\sigma \models p)$
- si Ξ est un connecteur binaire de la logique propositionnelle (i.e., $\lor$, $\land$, $\Rightarrow$ ou $\Leftrightarrow$), alors

  $\sigma \models p \;\Xi\; q \;\triangleq\; (\sigma \models p) \;\Xi\; (\sigma \models p)$
- $\sigma \models \forall\, x \cdot p \;\triangleq\; \forall\, x \cdot (\sigma \models p)$
- $\sigma \models \exists\, x \cdot p \;\triangleq\; \exists\, x \cdot (\sigma \models p)$

Soit $\mathcal{T}(A)$ l'ensemble des séquences d'états (ou d'événements) extraites des exécutions de A. On dénote par $A \models p$ la satisfaction par l'automate A de la formule p.

$$A \models p \;\triangleq\; \forall\, \sigma \cdot \sigma \in \mathcal{T}(A) \Rightarrow \sigma \models p$$

On note que

$$\neg(A \models p) \quad\Leftrightarrow\quad \exists\, \sigma \cdot \sigma \in \mathcal{T}(A) \land \sigma \models \neg p \quad .$$

1.3 Quelques lois de la LTL

(1)	$\neg\mathsf{G}p$	$\Leftrightarrow$	$\mathsf{F}\neg p$	(2)	$\neg\mathsf{F}p$	$\Leftrightarrow$	$\mathsf{G}\neg p$
(3)	$\mathsf{G}p$	$\Leftrightarrow$	$\neg\mathsf{F}\neg p$	(4)	$\mathsf{F}p$	$\Leftrightarrow$	$\neg\mathsf{G}\neg p$
(5)	$\neg(p \,\mathsf{U}\, q)$	$\Leftrightarrow$	$\neg q \,\mathsf{W}\, (\neg p \land \neg q)$	(6)	$\neg(p \,\mathsf{W}\, q)$	$\Leftrightarrow$	$\neg q \,\mathsf{U}\, (\neg p \land \neg q)$
(7)	$\mathsf{G}(p \land q)$	$\Leftrightarrow$	$(\mathsf{G}p) \land (\mathsf{G}q)$	(8)	$\mathsf{F}(p \lor q)$	$\Leftrightarrow$	$(\mathsf{F}p) \lor (\mathsf{F}q)$
(9)	$p \,\mathsf{U}\, (q \lor r)$	$\Leftrightarrow$	$(p \,\mathsf{U}\, q) \lor (p \,\mathsf{U}\, r)$	(10)	$p \,\mathsf{W}\, (q \lor r)$	$\Leftrightarrow$	$(p \,\mathsf{W}\, q) \lor (p \,\mathsf{W}\, r)$
(11)	$(p \land q) \,\mathsf{U}\, r$	$\Leftrightarrow$	$(p \,\mathsf{U}\, r) \lor (q \,\mathsf{U}\, r)$	(12)	$(p \land q) \,\mathsf{W}\, r$	$\Leftrightarrow$	$(p \,\mathsf{W}\, r) \lor (q \,\mathsf{W}\, r)$
(13)	$\mathsf{GF}(p \lor q)$	$\Leftrightarrow$	$(\mathsf{GF}p) \lor (\mathsf{GF}q)$	(14)	$\mathsf{FG}(p \land q)$	$\Leftrightarrow$	$(\mathsf{FG}p) \land (\mathsf{FG}q)$

1.4 Exemples

1. Soit les événements suivants d'un système de gestion de bibliothèque.

   (a) $C(bId)$: créer le livre bId

   (b) $S(bId)$: supprimer le livre bId

   (c) $P(bId, mId)$: prêter le livre bId au membre mId

   (d) $R(bId)$: retourner le livre bId

   (e) $V(bId, mId)$: réserver le livre bId pour le membre mId

   Traduisez chaque énoncé suivant en formule de logique temporelle linéaire. utilisez les quantificateurs $\forall$ et $\exists$ afin que vos formules soient fermées (c'est-à-dire qu'il n'y ait pas de variable libre). Prière de bien indenter vos formules afin de les rendre lisibles. Voici un exemple (bidon) où l'indentation fait office de parenthèse:

$$
\begin{array}{l}
\forall\, bId \cdot \\
\quad \mathsf{F} \\
\qquad\qquad C(bId) \\
\qquad \wedge \\
\qquad\qquad \neg \forall\, mId \cdot (P(bId, mId) \wedge R(bId) \wedge V(bId, mId)) \\
\qquad\quad \mathsf{U} \\
\qquad\qquad C(bId)
\end{array}
$$

   (a) Entre deux événements $S(bId)$ et $C(bId)$, il ne peut survenir d'événement $P(bId, mId)$, $R(bId)$, $V(bId, mId)$.
   Solution:

$$
\begin{array}{l}
\forall\, bId \cdot \\
\quad \mathsf{G} \\
\qquad\qquad S(bId) \\
\qquad \Rightarrow \\
\qquad\qquad\quad \neg \exists\, mId \cdot (P(bId, mId) \vee R(bId) \vee V(bId, mId)) \\
\qquad\qquad \mathsf{W} \\
\qquad\qquad\quad C(bId)
\end{array}
$$

   (b) Un livre disponible doit toujours pouvoir être emprunté. On sait qu'un livre est disponible lorsque l'événement $R(bId)$ survient.
   Solution:

$$
\begin{array}{l}
\forall\, bId \cdot \\
\quad \mathsf{G} \\
\qquad\qquad R(bId) \\
\qquad \Rightarrow \\
\qquad\qquad \mathsf{F}\, \exists\, mId \cdot P(bId, mId)
\end{array}
$$

   (c) Un membre ne peut emprunter deux livres en même temps.

Solution:

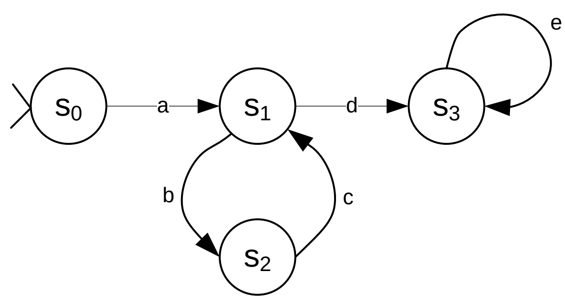
$$\begin{array}{l} \forall\, mId, bId_1 \cdot \\ \quad \mathsf{G} \\ \qquad\quad P(bId_1, mId) \\ \qquad \Rightarrow \\ \qquad\quad \mathsf{X} \\ \qquad\qquad\quad \neg \exists\, bId_2 \cdot P(bId_2, mId) \\ \qquad\qquad \mathsf{W} \\ \qquad\qquad\quad R(bId_1) \end{array}$$

(d) La bibliothèque est équitable, c'est-à-dire que chaque membre peut emprunter chaque livre autant de fois qu'il le veut.

Solution:

$$\forall\, mId, bId \cdot \mathsf{GF} P(bId, mId)$$

2. Soit l'automate A suivant.



(a) Est-ce que $A \models \mathsf{FG} e$?

Solution: Non, car la séquence suivante n'atteint jamais e. Elle permet de boucler infiniment sur b et c sans jamais atteindre e.

$$s_0 \xrightarrow{a} s_1 \xrightarrow{b} s_2 \xrightarrow{c} s_1 \xrightarrow{b} s_2 \xrightarrow{c} s_1 \ldots$$

(b) Est-ce que $A \models a \wedge \mathsf{X}(b \wedge \mathsf{XF} e)$?

Solution: Non, car la séquence suivante ne la satisfait pas

$$s_0 \xrightarrow{a} s_1 \xrightarrow{d} s_3 \xrightarrow{e} s_3 \xrightarrow{e} s_3 \ldots$$

(c) Comment pouvez-vous exprimer, à l'aide de la relation $A \models$ et sans utiliser de quantification explicite sur les traces, que l'automate A peut faire la séquence suivante:

$$s_0 \xrightarrow{a} s_1 \xrightarrow{b} s_2 \xrightarrow{c} s_1 \xrightarrow{d} s_3 \xrightarrow{e} s_3 \xrightarrow{e} s_3 \ldots$$

Solution: $\neg(A \models \neg(a \wedge \mathsf{X}(b \wedge \mathsf{X}(c \wedge \mathsf{X}(d \wedge \mathsf{X}(\mathsf{G} e))))))$

(d) Comment peut-on vérifier que l'événement e est accessible pour toujours dans au moins une séquence de transitions de A?

Solution: En vérifiant $\neg(A \models \neg \mathsf{FG} e)$.

3. Considérez la machine Event B suivante. On suppose que si la précondition d'une opération n'est pas satisfaite, alors l'opération ne peut s'exécuter. L'expression $x :\in S$ est une abbréviation en B pour la substitution ANY z WHERE $z \in S$ THEN $x := z$ END.

   MACHINE M
   VARIABLES s, x
   INVARIANT
   $s \in 0..1$
   $x \in \mathbb{N}$

   INITIALIZATION
   $s := 0$
   $x :\in \mathbb{N}$

   EVENTS
   $\mathsf{a} \triangleq$
   WHEN $s = 0$
   THEN $s, x :|$
   $(x > 0 \Rightarrow x' = x - 1 \wedge s' = s)$
   $\wedge$
   $(x = 0 \Rightarrow s' = 1 \wedge x' = x)$
   END

   $\mathsf{b} \triangleq$
   WHEN $s = 1$
   THEN
   $s := 0$
   $x :\in \mathbb{N}$
   END
   END.

   (a) Est-ce que la formule GFb est satisfaite par la machine M?
   Solution: Oui, car l'opération a décrémente x jusqu'à ce sa valeur tombe à zéro. Ensuite, elle passe à l'état $s = 1 \wedge x = 0$, d'où l'opération b peut s'exécuter pour ramener la machine à son état initial et recommencer infiniment ce comportement.

   (b) Si on ajoute l'opération c suivante, est-ce que la formule GFb est satisfaite?

   $$\mathsf{c} \triangleq \text{WHEN } s = 0 \text{ THEN } x :\in \mathbb{N} \text{ END}$$

   Solution: Non, car l'opération c permet de boucler infiniment sur a et c sans jamais atteindre b.

4. Est-ce que les formules suivantes sont équivalentes. Justifiez votre réponse

   (a) $p \Rightarrow \mathsf{XF}q$
   et
   $p \wedge \mathsf{XF}q$
   Solution: Non. La première n'exige pas que la séquence débute par p, alors que la deuxième oui.

(b) $(\mathsf{G}p)\ \mathsf{W}\ q$
et
$\mathsf{G}p$

Solution: Non. Dans la première, p peut être faux pour le premier élément de la séquence, alors qu'il doit être vrai dans la deuxième.

(c) $(\mathsf{G}p)\ \mathsf{U}\ (\mathsf{F}p)$
et
$\mathsf{G}p$

Solution: Non. Dans la première, lorsque p devient vrai, il peut devenir faux ensuite, alors que la deuxième exige que p soit toujours vrai.

(d) $(\mathsf{F}p)\ \mathsf{W}\ (\mathsf{G}p)$
et
$\mathsf{F}p$

Solution: Non. La séquence $p \neg p \neg p \ldots$ satisfait la deuxième mais pas la première.

5. Pour chaque formule ci-dessous, donnez une sequence qui la satisfait. Utilisez "…" et des commentaires pour rendre votre séquence précise, générale et bien illustrative.

(a) $\mathsf{F}a$

Solution: $\ldots a \ldots$

(b) $\mathsf{G}a$

Solution: $aaaa\ldots$; que des a

(c) $\mathsf{GF}a$

Solution: $\ldots a \ldots a \ldots$; une suite infinie de a avec potentiellement autre chose entre les a

(d) $\mathsf{FG}a$

Solution: $\ldots aaaaa \ldots$; n'importe quoi suivi d'une suite infinie de a

(e) $(a \vee b)\ \mathsf{W}\ c$

Solution: $aaabbbabbabbbaa \ldots c \ldots$; que des a ou b ensuite un c optionnel

(f) $(a \vee b)\ \mathsf{W}\ (c \wedge \mathsf{X}d)$

Solution: $aaabbbabbabbbaa \ldots cd \ldots$; que des a ou b ensuite un cd optionnel

(g) $(a \vee b)\ \mathsf{U}\ c$

Solution: $aaabbbabbabbbaa \ldots c \ldots$; que des a ou b ensuite un c obligatoire

(h) $(a \vee b)\ \mathsf{U}\ \mathsf{G}c$

Solution: $aaabbbabbabbbaa \ldots ccccc \ldots$; que des a ou b ensuite que des c obligatoirement

(i) $(\mathsf{F}a)\ \mathsf{U}\ \mathsf{G}a$

Solution: $\ldots a \ldots aaaaa \ldots$; un a survient et ensuite peut-être autre chose et ensuite une suite infinie de a

(j) $a \wedge \mathsf{X}(b \wedge \mathsf{X}c)$

Solution: $abc\ldots$; après c, n'importe quoi

(k) $\mathsf{GX}a$

Solution: $yaaaa\ldots$; après y (qui peut être n'importe quoi), que des a

(l) $\mathsf{FX}a$

Solution: $y \ldots a \ldots$; au moins un a après y (qui peut être n'importe quoi)

(m) $\mathsf{G}(a \,\mathsf{U}\, b)$
Solution: $aaaaaa \ldots baaaaaa \ldots baaaaaa \ldots b$; une suite de a suivie d'un b, répéter ce pattern à l'infini

6. Traduisez chacune des phrases suivantes en une formule de logique temporelle linéaire.

(a) Le système accepte a jusqu'à ce que b arrive; b doit obligatoirement arriver.
Solution: $a \,\mathsf{U}\, b$

(b) Le système accepte a jusqu'à ce que b arrive; b doit obligatoirement arriver. Ensuite, après b, le système alterne entre c et d.
Solution: $a \,\mathsf{U}\, (b \wedge \mathsf{XG}((c \vee d) \wedge (c \Rightarrow \mathsf{X}d) \wedge (d \Rightarrow \mathsf{X}c))$

(c) Le système doit servir de manière équitable les a et les b, c'est-à-dire que le système ne doit pas boucler infiniment sur l'un de sorte que l'autre n'est jamais servi.
Solution: $\mathsf{G}((\mathsf{F}a) \wedge (\mathsf{F}b))$

(d) Un livre réservé doit être emprunté ou bien sa réservation doit être annulée.
Solution: $\mathsf{G}(\mathsf{reserver}(b) \Rightarrow \mathsf{X}(\mathsf{F}(\mathsf{preter}(b) \vee \mathsf{annuler}(b))))$

(e) Un livre peut être réservé ou emprunté.
Solution: $\mathsf{F}(\mathsf{reserver}(b) \vee \mathsf{preter}(b))$

(f) Un membre qui s'inscrit peut toujours se désinscrire.
Solution: $\mathsf{G}(\mathsf{inscrire}(m) \Rightarrow \mathsf{Fdésinscrire}(m))$

(g) Un livre emprunté ne peut être supprimé.
Solution: $\mathsf{G}(\mathsf{preter}(b) \Rightarrow (\neg\mathsf{supprimer}(b) \,\mathsf{U}\, \mathsf{retourner}(b)))$

2 Logique temporelle arborescente

La logique temporelle arborescente (*Computational Tree Logic* - CTL) considère le système de transition plutôt que les traces du système. Elle permet d'exprimer des propriétés que l'on ne peut exprimer en LTL, et vice-versa. Certaines propriétés peuvent être exprimées dans les deux logiques; elles ont donc une intersection non-vide.

2.1 Syntaxe

Les éléments suivants sont des formules *atomiques* de la CTL :

- **true** et **false**;
- une variable propositionnelle;
- une formule atomique de la logique du premier ordre.

Des formules complexes sont construites avec les connecteurs de la logique propositionnelle et des connecteurs temporels. Soit p et q des formules CTL. Alors les expressions suivantes sont sont des formules CTL:

- *connecteurs propositionnels* : $\neg p$, $p \wedge q$, $p \vee q$, $p \Rightarrow q$, $p \Leftrightarrow q$,
- *connecteurs temporels : tous les chemins* : $\mathsf{A}(p \,\mathsf{W}\, q)$, $\mathsf{A}(p \,\mathsf{U}\, q)$, $\mathsf{AG}p$, $\mathsf{AF}p$, $\mathsf{AX}\ p$
- *connecteurs temporels : existe un chemin* : $\mathsf{E}(p \,\mathsf{W}\, q)$, $\mathsf{E}(p \,\mathsf{U}\, q)$, $\mathsf{EG}p$, $\mathsf{EF}p$, $\mathsf{EX}\ p$
- *quantificateurs de la logique du premier ordre* : $\forall\, x \cdot p$, $\exists\, x \cdot p$.

2.2 Sémantique

La sémantique d'une formule CTL est basée sur les exécutions à partir d'un état q. Soit $\mathcal{T}(q)$ l'ensemble des séquences d'états extraites des exécutions partant de q. On dénote par $q \models p$ la satisfaction par q de la formule CTL p. On définit $q \models p$ comme suit, en ré-utilisant la sémantique de LTL pour les séquences d'états démarrant à q.

- Si p est une formule atomique de la CTL, alors $q \models p \ \triangleq\ q$ satisfait p. La satisfaction de p par q dépend de la nature de p.
- $q \models \mathsf{A}p \ \triangleq\ \forall \sigma : \mathcal{T}(q) \bullet \sigma \models p$
- $q \models \mathsf{E}p \ \triangleq\ \exists \sigma : \mathcal{T}(q) \bullet \sigma \models p$
- $\sigma \models p_1 \,\mathsf{W}\, p_2 \ \triangleq\ \mathsf{head}(\sigma) \models p_2 \vee (\mathsf{head}(\sigma) \models p_1 \wedge \mathsf{tail}(\sigma) \models p_1 \,\mathsf{W}\, p_2)$
- $\sigma \models p \,\mathsf{U}\, q \ \triangleq\ \sigma \models (p \,\mathsf{W}\, q) \wedge (\exists \sigma_1, \sigma_2 \cdot \sigma = \sigma_1 {}^\frown \sigma_2 \wedge \mathsf{head}(\sigma_2) \models q)$
- $\sigma \models \mathsf{G}p \ \triangleq\ \sigma \models p \,\mathsf{W}\, \mathbf{false}$
- $\sigma \models \mathsf{F}p \ \triangleq\ \sigma \models \mathbf{true} \,\mathsf{U}\, p$
- $\sigma \models \mathsf{X}p \ \triangleq\ \mathsf{head}(\mathsf{tail}(\sigma)) \models p$
- $\sigma \models \neg p \ \triangleq\ \neg(\sigma \models p)$
- si Ξ est un connecteur binaire de la logique propositionnelle (i.e., $\vee$, $\wedge$, $\Rightarrow$ ou $\Leftrightarrow$), alors $\sigma \models p \;\Xi\; q \ \triangleq\ (\sigma \models p) \;\Xi\; (\sigma \models p)$
- $\sigma \models \forall x \cdot p \ \triangleq\ \forall x \cdot (\sigma \models p)$
- $\sigma \models \exists x \cdot p \ \triangleq\ \exists x \cdot (\sigma \models p)$

On dénote par $A \models p$ la satisfaction par l'automate A de la formule p.

$$A \models p \quad \Leftrightarrow \quad q_0 \models p$$

2.3 Exemples

1. Soit les événements suivants d'un système de gestion de bibliothèque.

   (a) $C(bId)$: créer le livre bId
   (b) $S(bId)$: supprimer le livre bId
   (c) $P(bId, mId)$: prêter le livre bId au membre mId
   (d) $R(bId)$: retourner le livre bId
   (e) $V(bId, mId)$: réserver le livre bId pour le membre mId

   Traduisez chaque énoncé suivant en formule de logique temporelle arborescente.

(a) Entre deux événements $S(bId)$ et $C(bId)$, il ne peut survenir d'événement $P(bId, mId)$, $R(bId)$, $V(bId, mId)$.
Solution:

$$
\begin{array}{l}
\forall\, bId \cdot \\
\quad \mathsf{AG} \\
\qquad\quad S(bId) \\
\qquad \Rightarrow \\
\qquad\quad \mathsf{A} \\
\qquad\qquad\quad \neg\, \exists\, mId \cdot (P(bId, mId) \vee R(bId) \vee V(bId, mId)) \\
\qquad\qquad \mathsf{W} \\
\qquad\qquad\quad C(bId)
\end{array}
$$

(b) Un livre disponible doit toujours pouvoir être emprunté. On sait qu'un livre est disponible lorsque l'événement $R(bId)$ survient.
Solution:

$$
\begin{array}{l}
\forall\, bId \cdot \\
\quad \mathsf{AG} \\
\qquad\quad R(bId) \\
\qquad \Rightarrow \\
\qquad\quad \mathsf{EF}\, \exists\, mId \cdot P(bId, mId)
\end{array}
$$

(c) Un membre ne peut emprunter deux livres en même temps.
Solution:

$$
\begin{array}{l}
\forall\, mId, bId_1 \cdot \\
\quad \mathsf{AG} \\
\qquad\quad P(bId_1, mId) \\
\qquad \Rightarrow \\
\qquad\quad \mathsf{AX} \\
\qquad\qquad \mathsf{A} \\
\qquad\qquad\qquad \neg\, \exists\, bId_2 \cdot P(bId_2, mId) \\
\qquad\qquad\quad \mathsf{W} \\
\qquad\qquad\qquad R(bId)
\end{array}
$$

(d) Un livre est toujours empruntable, c'est-à-dire que chaque membre peut emprunter chaque livre autant de fois qu'il le veut.
Solution:

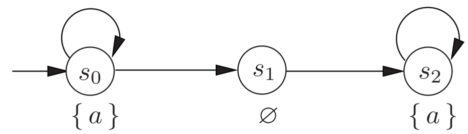
$$\forall\, mId, bId \cdot \mathsf{AGEF} P(bId, mId)$$

3 Équivalence LTL-CTL

Il semble facile de transformer une formule LTL en une formule CTL en ajoutant le quantificateur de chemin A devant chaque opérateur LTL. Cela ne donne pas toujours une formule équivalente. Par exemple, les formules suivantes ne sont pas équivalentes

$\mathsf{AFAG}\phi$ $\qquad\qquad$ $\mathsf{FG}\phi$

La formule LTL est vraie pour l'automate suivant, mais la formule CTL est fausse.



La formule LTL est vraie pour cet automate, car toute trace infinie contient toujours un suffixe ne comportant que des a. Une trace infinie boucle soit sur l'état s_0 ou sur l'état s_2. Dans les deux cas, on a une suite infinie de a dans le suffixe de la trace. La formule CTL est fausse, car une trace qui boucle sur s_0 offre toujours une branche menant à s_1 où a n'est pas satisfaite.

Il existe des formules LTL qui n'ont pas d'équivalent CTL, et vice-versa. Par exemple, les formules LTL suivantes n'ont pas d'équivalent en CTL.

$\mathsf{FG}\phi \qquad \mathsf{F}(\phi \wedge \mathsf{X}\phi)$

Les formules CTL suivantes n'ont pas d'équivalent en LTL

$\mathsf{AFAG}\phi \qquad \mathsf{AF}(\phi \wedge \mathsf{AX}\phi) \qquad \mathsf{AGEF}\phi$

4 Classes et patrons de propriétés

On distingue deux grandes classes de propriétés:

1. propriété de sûreté : *quelque chose de mauvais n'arrivera jamais.*

   Exemple: La porte ne peut jamais être ouverte en insérant une seule des deux clés dans un coffre de sécurité.

   Exemple: La distance entre deux trains est toujours supérieure à 300m (invariant).

   Les invariant du langage B sont des propriétés de sûreté.

2. propriété de vivacité : *quelque chose de bon arrivera inévitablement.*

   Exemple: une requête du client sera toujours servie.

4.1 Patrons de propriétés temporelles (Dwyer)

Absence: P est faux *(sûreté)*

Globalement	$\mathsf{G}(\neg P)$
Avant R	$\mathsf{F}R \Rightarrow (\neg P \,\mathsf{U}\, R)$
Après Q	$\mathsf{G}(Q \Rightarrow \mathsf{G}(\neg P))$
Entre Q et R	$\mathsf{G}((Q \wedge \neg R \wedge \mathsf{F}R) \Rightarrow (\neg P \,\mathsf{U}\, R))$
Après Q jusqu'à R	$\mathsf{G}(Q \wedge \neg R \Rightarrow (\neg P \,\mathsf{W}\, R))$

Existence: P devient vrai *(vivacité)*

Globalement	$\mathsf{F}(P)$
Avant R	$\neg R \,\mathsf{W}\, (P \wedge \neg R)$
Après Q	$\mathsf{G}(\neg Q) \vee \mathsf{F}(Q \wedge \mathsf{F}P))$
Entre Q et R	$\mathsf{G}(Q \wedge \neg R \Rightarrow (\neg R \,\mathsf{W}\, (P \wedge \neg R)))$
Après Q jusqu'à R	$\mathsf{G}(Q \wedge \neg R \Rightarrow (\neg R \,\mathsf{U}\, (P \wedge \neg R)))$

Existence avec un nombre d'instances fixé: P devient vrai au plus 2 fois dans les exemples ci-dessous *(vivacité)*

Globalement	$(\neg P\ \mathsf{W}\ (P\ \mathsf{W}\ (\neg P\ \mathsf{W}\ (P\ \mathsf{W}\ \mathsf{G}\neg P))))$
Avant R	$\mathsf{F}R \Rightarrow ((\neg P \wedge \neg R)\ \mathsf{U}\ (R \vee ((P \wedge \neg R)\mathsf{U}$ $(R \vee ((\neg P \wedge \neg R)\ \mathsf{U}\ (R \vee ((P \wedge \neg R)\mathsf{U}$ $(R \vee (\neg P\ \mathsf{U}\ R))))))))$
Après Q	$\mathsf{F}Q \Rightarrow (\neg Q\ \mathsf{U}\ (Q \wedge (\neg P\ \mathsf{W}\ (P\ \mathsf{W}\ (\neg P\ \mathsf{W}\ (P\ \mathsf{W}\ \mathsf{G}\neg P))))))$
Entre Q et R	$\mathsf{G}((Q \wedge \mathsf{F}R) \Rightarrow$ $((\neg P \wedge \neg R)\ \mathsf{U}\ (R \vee ((P \wedge \neg R)\mathsf{U}$ $(R \vee ((\neg P \wedge \neg R)\ \mathsf{U}\ (R \vee ((P \wedge \neg R)\mathsf{U}$ $(R \vee (\neg P\ \mathsf{U}\ R)))))))))$
Après Q jusqu'à R	$\mathsf{G}(Q \Rightarrow ((\neg P \wedge \neg R)\ \mathsf{U}\ (R \vee ((P \wedge \neg R)\mathsf{U}$ $(R \vee ((\neg P \wedge \neg R)\ \mathsf{U}\ (R \vee ((P \wedge \neg R)\mathsf{U}$ $(R \vee (\neg P\ \mathsf{W}\ R) \vee \mathsf{G}P)))))))))$

Universalité: P est vrai *(sûreté)*

Globalement	$\mathsf{G}(P)$
Avant R	$\mathsf{F}R \Rightarrow (P\ \mathsf{U}\ R)$
Après Q	$\mathsf{G}(Q \Rightarrow \mathsf{G}(P))$
Entre Q et R	$\mathsf{G}((Q \wedge \neg R \wedge \mathsf{F}R) \Rightarrow (P\ \mathsf{U}\ R))$
Après Q jusqu'à R	$\mathsf{G}(Q \wedge \neg R \Rightarrow (P\ \mathsf{W}\ R))$

Précédence: S précède P *(sûreté)*

Globalement	$\neg P\ \mathsf{W}\ S$
Avant R	$\mathsf{F}R \Rightarrow (\neg P\ \mathsf{U}\ (S \vee R))$
Après Q	$\mathsf{G}\neg Q \vee \mathsf{F}(Q \wedge (\neg P\ \mathsf{W}\ S))$
Entre Q et R	$\mathsf{G}((Q \wedge \neg R \wedge \mathsf{F}R) \Rightarrow (\neg P\ \mathsf{U}\ (S \vee R)))$
Après Q jusqu'à R	$\mathsf{G}(Q \wedge \neg R \Rightarrow (\neg P\ \mathsf{W}\ (S \vee R)))$

Réponse: S en réponse à P *(vivacité)*

Globalement	$\mathsf{G}(P \Rightarrow \mathsf{F}S)$
Avant R	$\mathsf{F}R \Rightarrow (P \Rightarrow (\neg R\ \mathsf{U}\ (S \wedge \neg R)))\ \mathsf{U}\ R$
Après Q	$\mathsf{G}(Q \Rightarrow \mathsf{G}(P \Rightarrow \mathsf{F}S))$
Entre Q et R	$\mathsf{G}((Q \wedge \neg R \wedge \mathsf{F}R) \Rightarrow (P \Rightarrow (\neg R\ \mathsf{U}\ (S \wedge \neg R)))\ \mathsf{U}\ R)$
Après Q jusqu'à R	$\mathsf{G}(Q \wedge \neg R \Rightarrow ((P \Rightarrow (\neg R\ \mathsf{U}\ (S \wedge \neg R)))\ \mathsf{W}\ R)$