INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Model-Checking

François Vernadat
INSA-DGEI

vernadat@laas.fr

Université de Toulouse

Outline

Introduction

What is model checking ? – Amir Pnueli 2000

"Model checking is the method by which a desired behavioral property of a reactive system is verified over a given system (the model) through exhaustive enumeration (explicit or implicit) of all the states reachable by the system and the behaviors that traverse through them."

Programme "Classique"

- Termine
- Retourne un résultat
- Données +/- complexes mais contrôle séquentiel ("simple")

Ex : Algo de tri, compilateur, , ...

Correction = correction partielle + terminaison

Système Réactif (> 70)

- Interactions entre Σ et son envt
- Ne retourne pas de résultat
- Terminaison en général indésirée (deadlock)
- Données +/- simples mais contrôle concurrent ("compliqué")

Ex : Protocoles, Contrôle/Commande, Σ-Embarqués, ...

Correction = sûreté, vivacité, équité, ...

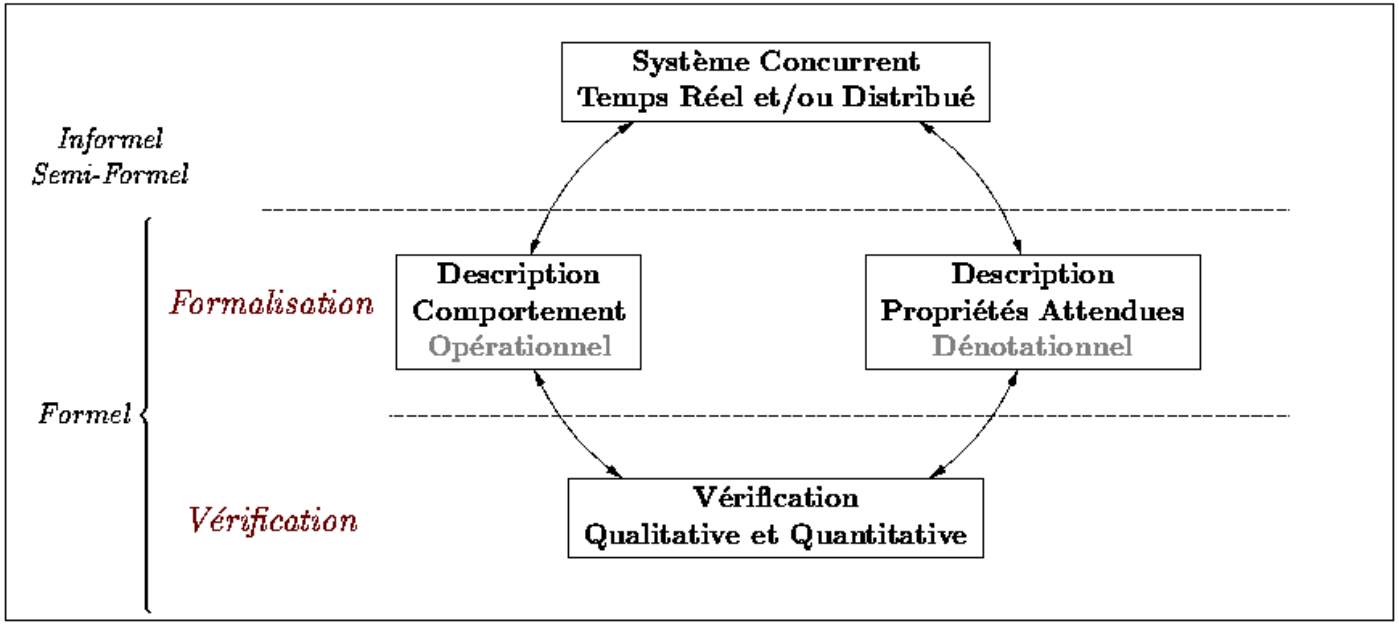
Classe de propriétés des systèmes réactifs

- **Accessibilité**
  - Une certaine situation peut être atteinte
  - Chaque action peut avoir lieu
- **Invariant**
  - Propriété vraie en chaque état
    La barrière est fermée lorsqu'un train est sur le passage à niveau
- **Sûreté**
  - Quelque chose de mauvais n'arrive jamais
    Jamais plus de 2 processus en section critique
    Jamais de débordement de buffers
- **Vivacité**
  - Quelque chose de bon finit par arriver
    Un philosophe affamé finira par manger
- **Equité(s)**
  - Si une action a lieu une infinité de fois alors une autre action aura lieu une (infinité) de fois
  - Quelque chose (de bon) se répète infiniment souvent
    S'il n'y a pas une infinité de pertes, tout message émis - perdu et retransmis - finira par arriver (Alternating Bit Protocol)

Model-Checking

Ingrédients

$\mathcal{M}$: Modèle (formel) du comportement (**issu de** Rdp, Automates, Alg.Proc,..)
ϕ : Modèle (formel) des propriétés (formalisme dédié)
MC : Algorithme de vérification permettant de décider si $\mathcal{M}$ satisfait ϕ?
nb : Formel est une condition requise pour envisager l'automaticité



Assertionnel

Propriété ϕ exprimée dans une logique ad hoc
$\mathcal{M} \models \phi$ (ce cours-ci)

Comportemental

Propriété ϕ exprimée par un "graphe" G_ϕ
$\mathcal{M} \equiv G_\phi$ (un autre cours)

Model-Checking : les + et les -

Inconvénients

- (Longtemps) "Eloigné" des pratiques usuelles industrielles
  - Disposer d'un modèle formel du comportement
  - Formalisation "complexe" des exigences/propriétés
- Passage à l'échelle des outils
  - Pb adapté au model-checking
  - Modèle fini et "petit"

Avantages

Modèle formel

- Lève des ambiguités en phases amont : Specs/Conception
- Révèle les erreurs au plus tôt
- Σ plus mûr plut tôt
- Développement plus efficace (même si les phases amont sont plus coûteuses) **Model-Checking**
- Générique, Automatique (Preuve manuelle)
- Exhaustif (Test partiel)
- Confiance (Σ Critiques)

Evolutions

Approches modèles (IDM)

Reconnaissance des méthodes formelles par les organismes de certification

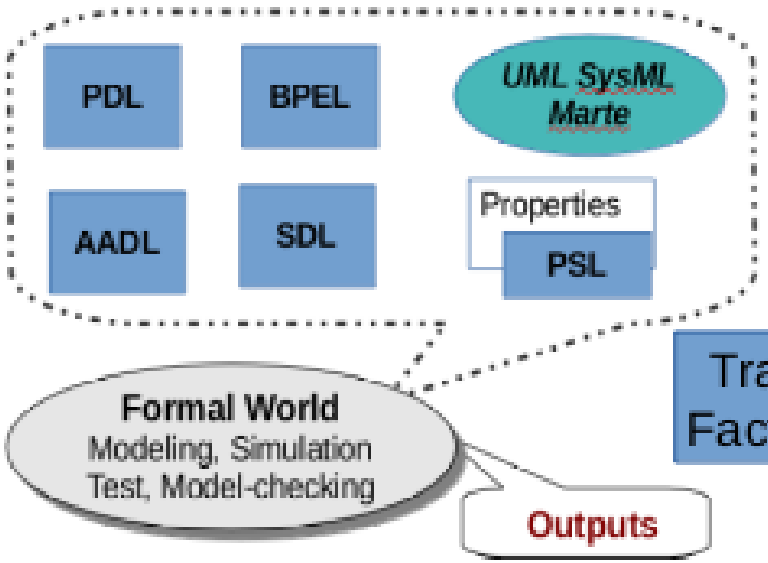
Utilisation des méthodes formelles recommandées dans le domaine ferroviaire

DO-333 supplément technique du DO-178C sur les méthodes formelles

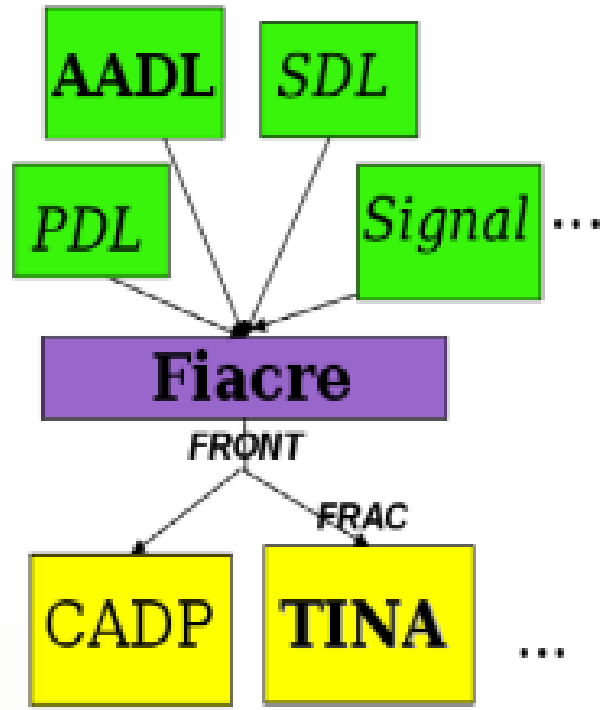
Intérêt fort des industriels : Pôle AESE, IRT Saint-Exupéry

http ://projects.laas.fr/IFSE/FMF

Model-checking et Process industriels



Translation
Factorization



FIACRE "Intermediate Format for the Embedded Distributed Component Architectures"

INRIA/VASY, IRIT, LAAS-CNRS

TOPCASED

OpenEmbeDD

QUARTET

IRT SAINT EXUPERY

Panorama du Cours & Organisation/Evaluation

Panorama

1. Introduction au model-checking et aux logiques temporelles
2. Logiques modales
3. Logiques temporelles : Linéaire & Arborescente
   Définition, Ex de Spécification, Formalisation, Algorithmique
4. Conclusion
   Compléments, Linéaire Vs Arborescent, ... Problèmes en cours
5. Références
   Outils, Bibliographie

Organisation & Evaluation

Volumes 10H CM + 2*TP

Evaluation 1H Exam - Documents autorisés

Outline

Logiques Modales

Historique

Aristote (322 A J.C) / Clarence Lewis ($\approx$ 1910) Forme Actuelle

Logiques Temporelles ($\geq$ 1980)
- Manna-Pnuelli (WI)
- Clark-Emerson (CMU)
- Sifakis (Verimag)

Logiques Epistemologiques ($\geq$ 1970)
- Hintikka (univ / Hailpern (IBM)

Logiques Déontiques, etc, etc

Awards

1996 ACM Turing : Pnuelli – Logique temporelle

2007 ACM Turing : Clarke, Emerson, & Sifakis – Model-Checking

Logique : Terminologie

Logique = Syntaxe + Axiomatique + Sémantique

- Syntaxe $\rightsquigarrow$ Définition du langage
- Axiomatique $\rightsquigarrow$ Mécanisme déductif
  (Axiomes + Règles d'Inférence, $\vdash$)
- Sémantique $\rightsquigarrow$ Validité des formules
  (Modèle, Relation de Satisfaisabilité $\models$)

Plan

- Calcul propositionnel (déjà connu)
- Logique Modale

Calcul Propositionnel (1/3)

Ingrédients

$\mathcal{P}$, Un ensemble de variables propositionnelles
$\mathcal{P} = \{p_1, p_2 \dots p_n\}$

c, Un ensemble de connecteurs
$c = \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$

$C_{Prop}(\mathcal{P})$ **Calcul propositionnel** construit sur $\mathcal{P}$

Syntaxe

- $p \in C_{Prop}(\mathcal{P}) \quad \forall p \in \mathcal{P}$
- Si $f \in C_{Prop}(\mathcal{P})$ Alors $\neg f \in C_{Prop}(\mathcal{P})$
- Si $f, g \in C_{Prop}(\mathcal{P})$ Alors $f \ \# \ g \in C_{Prop}(\mathcal{P})$
  où $\# \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$

Calcul Propositionnel (2/3)

Sémantique

1. $\mathcal{M}$, **Modèle**
   pour le calcul propositionnel, $\mathcal{M}$ est un sous-ensemble de $\mathcal{P}$
2. $\models$, **Relation de satisfaction**,
   définie par induction sur le structure des formules

   Pour $p \in \mathcal{P}, f$ et $g \in C_{Prop}(\mathcal{P})$
   - $\mathcal{M} \models p$ ssi $p \in \mathcal{M}$ $(\forall p \in \mathcal{P})$
   - $\mathcal{M} \models \neg f$ ssi Non $(\mathcal{M} \models f)$
   - $\mathcal{M} \models f \wedge g$ ssi $(\mathcal{M} \models f$ et $\mathcal{M} \models f)$

   Définition : Formule A est **valide** (notée $\models A$) ssi $\mathcal{M} \models A \quad \forall M$

Sucre

$A \vee B \equiv \neg(\neg A \wedge \neg B)$
$A \Rightarrow B \equiv \neg A \vee B$
$A \Leftrightarrow B \equiv (A \Rightarrow B) \wedge (A \Leftarrow B)$

Calcul Propositionnel (3/3)

Axiomatique $\rightsquigarrow$ **Démontration, théorèmes**

- Axiomes
  1. $A \to (B \to A)$
  2. $((A \to (B \to C)) \to ((A \to B) \to (A \to C)))$
  3. $((\neg B \to \neg A) \to ((\neg B \to A) \to B))$
- Règle d'inférence (Modus ponens) $\dfrac{\vdash A \to B, \quad \vdash A}{\vdash B}$

Liens entre Sémantique et Axiomatique

Adéquation : $\vdash A \Rightarrow \models A$ **& Complétude** $\models A \Rightarrow \vdash A$

Une formule démontrable est valide / Une formule valide est démontrable

NB : Par la suite, l'aspect axiomatique ne sera plus considéré.

Toutes les logiques considérées sont complètes et adéquates

Logiques Modales (1/)

Ingrédients : Les mêmes augmentées par deux **Modalités : $\Box, \Diamond$**

$\mathcal{P}$, Un ensemble de variables propositionnelles $\mathcal{P} = \{p_1, p_2 \dots p_n\}$

c, Un ensemble de connecteurs $c = \{\neg, \vee, \Box, \Diamond\}$

$LM_{Prop}(\mathcal{P})$ **Logique Modale** construite sur $\mathcal{P}$

Syntaxe

- $p \in LM_{Prop}(\mathcal{P}) \quad \forall p \in \mathcal{P}$
- Si $f \in LM_{Prop}(\mathcal{P})$ Alors $\neg f, \Box f, \Diamond f \in LM_{Prop}(\mathcal{P})$
- Si $f, g \in LM_{Prop}(\mathcal{P})$ Alors $f \ \wedge \ g \in LM_{Prop}(\mathcal{P})$

Sémantique

A définir : : Un modèle $\mathcal{M}$ et une relation de satisfaction $\models$
.... 1960 !

Modèle pour la logique modale

Sémantique des Mondes Possibles (S. Kripke 1960)

Structure de Kripke : $\mathcal{M} =< W, R, \nu >$ où

W est un ensemble de mondes

R est une relation entre ces mondes ($R \subset W \times W$)

ν est une valuation $\nu : W \mapsto \mathcal{P}(P)$

Exemple de Structure de Kripke

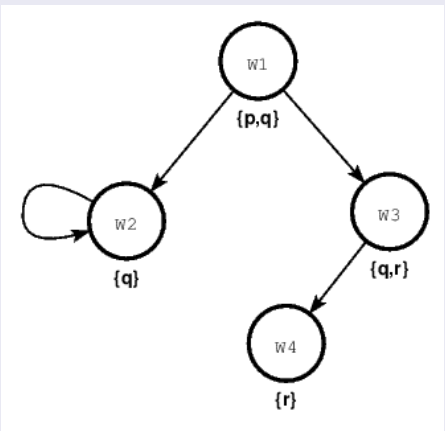
$\mathcal{M} =< W, R, \nu >$

$W = \{w_1, w_2, w_3, w_4\}$

$R = \{(w_1, w_2), (w_1, w_3), (w_2, w_2), (w_3, w_4)\}$

ν : $w_1 \mapsto \{p, q\}$ $\quad$ $w_3 \mapsto \{r, q\}$

$\quad\quad$ $w_2 \mapsto \{q\}$ $\quad$ $w_4 \mapsto \{r\}$



Notations : $w_i \ R \ w_j$ ssi $(w_i, w_j) \in R$ **&** $R(w_i) =_{Def} \{w_j \in M : w_i R w_j\}$

$\models$: relation de satisfaction

Notations

$\mathcal{M}, w \models A$ signifie que la formule A est satisfaite
dans le monde w du modèle $\mathcal{M}$.
$\mathcal{M} \models A$ signifie A est vraie dans tout monde de $\mathcal{M}$

Définition de $\models$

Pour $p \in \mathcal{P}$ & $f, g \in LM(\mathcal{P})$

$\mathcal{M}, w \models p$ ssi $p \in \nu(w)$ $(\forall p \in \mathcal{P})$

$\mathcal{M}, w \models \neg f$ ssi Non $(\mathcal{M}, w \models f)$

$\mathcal{M}, w \models f \wedge g$ ssi $\mathcal{M}, w \models f$ et $\mathcal{M}, w \models g$

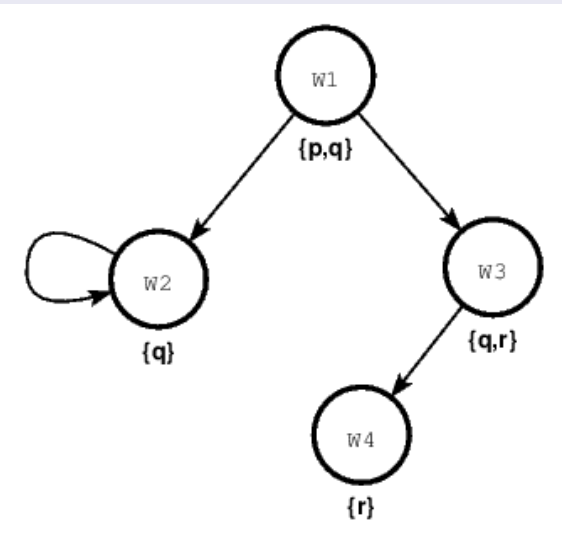
$\mathcal{M}, w \models \Box f$ ssi $\forall w' \in \mathcal{M} : w \ R \ w' \Rightarrow \mathcal{M}, w' \models f$

$\mathcal{M}, w \models \Diamond f$ ssi $\exists w' \in \mathcal{M} : w \ R \ w'$ et $\mathcal{M}, w' \models f$

Conventions : $\Box$ *Nécessaire* & $\Diamond$ *Possible*
A manier avec prudence (cf exo #2)

Sémantique Logique Modale

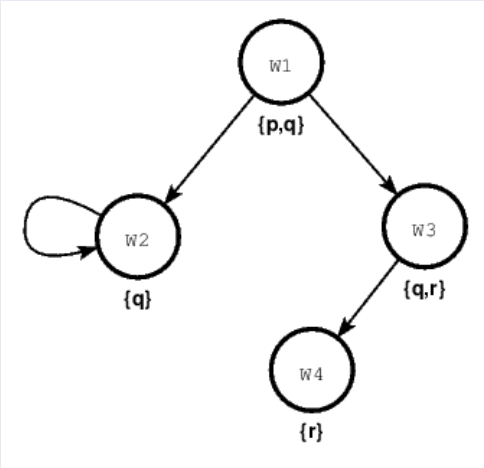
Exemple



1. $\mathcal{M}, w_1 \models \Box q$
   car $R(w_1) = \{w_2, w_3\}$,
   $q \in \nu(w_2)$ & $q \in \nu(w_3)$
   donc $w_2 \models q$ et $w_3 \models q$
2. $\mathcal{M}, w_1 \models \Diamond r$
   car w_1 R w_3 & $w_3 \models r$
3. $\mathcal{M}, w_1 \models \Diamond \neg r$
   car w_1 R w_2 & $w_2 \not\models r$
   (i.e. $r \notin \nu(w_2)$)
4. $\mathcal{M}, w_1 \models \neg \Box r$
   car $w_2 \in R(W_1)$ & $w_2 \not\models r$
5. $\mathcal{M}, w_1 \models \Diamond \Diamond r$
   car w_1 R w_3 & $w_3 \models \Diamond r$

Exercices

Le Modèle



Exo #1

	w_1	w_2	w_3	w_4
q				
$\Diamond q$				
$\Diamond \neg q$				
$\Box q$				
$\Box \neg q$				

Exo #2

1. $\mathcal{M}, w_1 \models \Diamond \Box q$
2. $\mathcal{M}, w_2 \models (\Box q) \wedge (\Diamond q) \wedge q$
3. $\mathcal{M}, w_4 \models (\Box q) \wedge (\Box \neg q) \wedge (\neg \Diamond q) \wedge (\neg \Diamond \neg q)$

Exo #3 : Montrer que pour toute SK $\mathcal{M} =< W, R, \nu >$

$\forall w \in W$ Si $R(w) = \emptyset$ alors $\quad \mathcal{M}, w \models \Box f \quad (\forall f)$

$\mathcal{M}, w \models \neg \Diamond f \quad (\forall f)$

"Instanciation" des Logiques Modales
Interprétation des modalités et de la relation $\rightarrow$

Logiques Temporelles

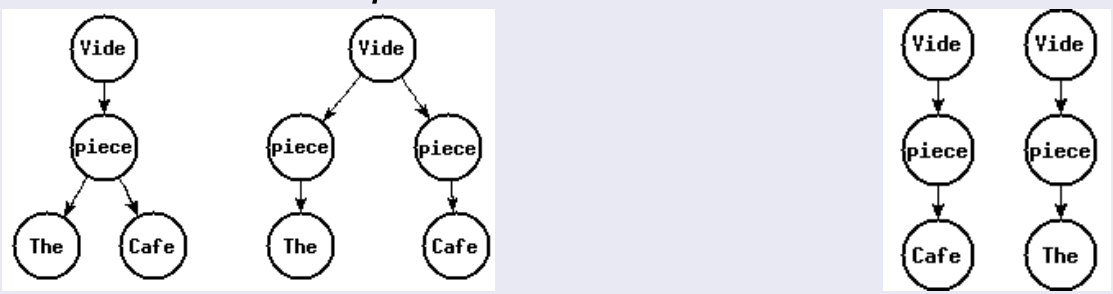
$\rightarrow$ accessibilité temporelle, $\rightarrow$ réflexive, transitive
$\Box$ "pour tous les mondes", $\Diamond$ "pour un monde futur"

Quel modèle de temps

Linéaire (un seul futur possible) : on raisonne sur des séquences/traces
Arborescent (non déterministe) : on raisonne sur des graphes

Arborescent Vs Linéaire

2 distributeurs de boissons possédant les mêmes traces
tout en offrant des comportements différents



... les deux approches - linéaire et arborescent - sont complémentaires !

"Instanciation" des Logiques Modales

Quel modèle de temps (suite)

Linéaire (un seul futur possible) : on raisonne sur des séquences/traces
Arborescent (non déterministe) : on raisonne sur des graphes

Futur / Passé (fini ou non), Discret Vs Dense, Qualitatif Vs Quantitatif
Logiques temporelles temporisées (timed temporal logic) ...

Suite du cours & TP associés

Futur, Discret, Qualitatif, Linéaire & Arborescent
Logique temporelle à temps linéaire (LTL)
Logique temporelle à temps arborescent (CTL)

Autres instances de logiques modales (non traité)

Epistémique : Savoir / Déontique : Devoir / ...

Outline

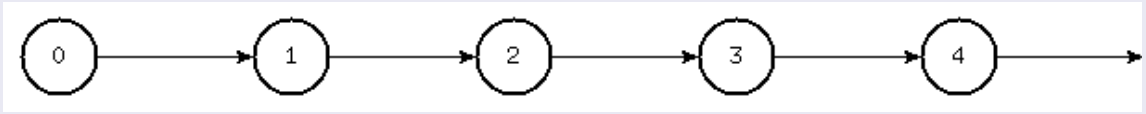
Logique Temporelle Linéaire : introduction

Structure de Kripke (rappel)

SK $= \mathcal{M} =< W, R, \nu >$ où W, l'ensemble de mondes, $R \subset W \times W$, la relation entre-eux et ν, la valuation $\nu : W \mapsto \mathcal{P}(P)$

Structure de Kripke et temps linéaire

Temps Linéaire : un seul successeur possible pour R



$\mapsto$ Simplification possible des structures de Kripke

On peut supprimer R en indexant les mondes

SK $= \mathcal{M} =< W, Index, \nu >$ avec W et ν inchangés et $Index : W \mapsto \mathbb{N}$

Par la suite et sans perte de généralité, on considère uniquement des traces/séquences/exécutions W infinies (R est totale)

Pour les éventuels états de blocage, on intorduit une boucle mais on ajoute à P une nouvelle proposition dead (traitement identique pour CTL)

Prise en compte du non-déterministe vu plus tard

Logique Temporelle Linéaire (LTL) (1/2)

Les origines

DUX : Logique temporelle linéaire à temps discret Gabbay-Pnuelli (1980)

Ingrédients

Connecteurs classiques + $\Box, \Diamond, \bigcirc, \sqcup$
$\Box, \Diamond, \bigcirc$ Connecteurs Unaires
$\sqcup$ Binaire

Syntaxe

- $p \in LTL_p(\mathcal{P}) \quad \forall p \in \mathcal{P}$
- Si $f \in LTL_p(\mathcal{P})$ Alors $\neg f, \bigcirc f, \Box f, \Diamond f \in LTL_p(\mathcal{P})$
- Si $f, g \in LTL_p(\mathcal{P})$ Alors $f \ \# \ g \in LTL_p(\mathcal{P})$ où $\# \in \{\wedge, \ \sqcup \}$

Modèle

SK $= < (w_o, w_1, \ldots w_n \ldots), \nu >$ où $(w_o, w_1, \ldots w_n \ldots)$ est une suite **infinie** de mondes
et ν une valuation (inchangé)

Logique Temporelle Linéaire (LTL) (2/2)

$\models$, relation de satisfaction

Pour $p \in \mathcal{P}$ & $f, g \in Ltl(\mathcal{P})$

$w \models p$ ssi $p \in \nu(w)$

$w \models \neg f$ ssi Non $(M, w \models f)$

$w \models f \wedge g$ ssi $(M, w \models f)$ et $(M, w \models g)$

◯ next-time operator

$w_k \models \bigcirc f$ ssi $M, w_{k+1} \models f$

f doit être vraie dans l'état suivant

◇ opérateur de Possibilité

$w_k \models \Diamond f$ ssi $\exists j \geq k : M, w_j \models f$

f doit être vraie dans le futur

□ opérateur de Nécessité

$w_k \models \Box f$ ssi $\forall j \geq k : M, w_j \models f$

f reste toujours vraie dans le futur

⊔ Until operator

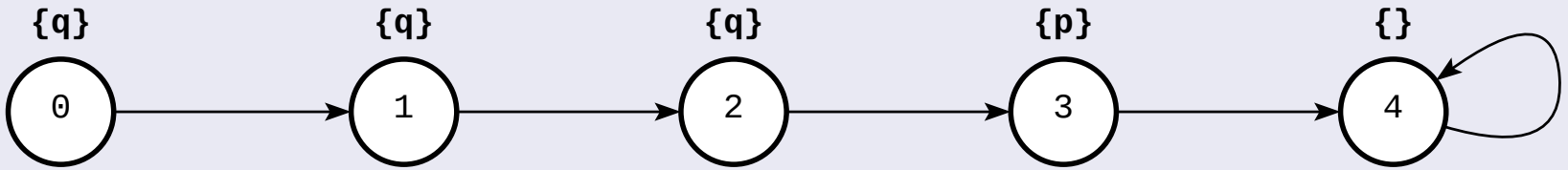
$w_k \models f \sqcup g$ ssi $\exists j \geq k : M, w_j \models g$ et $\forall i \in [k, j[\quad M, w_i \models f$

g deviendra vraie et f restera vraie jusqu'à ce que g le devienne

Futur large : le présent est compris dans le futur ($\geq$)

Exemples

Le modèle

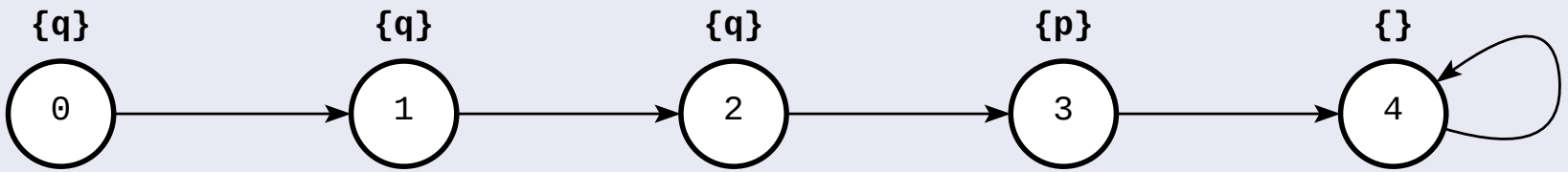


Exemple

1. $w_0 \models q$ $\quad q \in \nu(0)$
2. $w_0 \models \bigcirc q$ $\quad q \in \nu(1)$
3. $w_0 \models \bigcirc \bigcirc q$ $\quad q \in \nu(2)$
4. $w_0 \models \Diamond q$ $\quad q \in \nu(0)$ *Le futur commence au présent*
5. $w_0 \models \Diamond \neg q$ $\quad q \notin \nu(3)$
6. $w_0 \models q \sqcup p$ $\quad p \in \nu(3)$ et $q \in \nu(i) : \forall i \in [0, 3[$

Exercices

Le modèle



Exo #1

	0	1	2	3	4	5	...
$q \sqcup p$							
$\Diamond\Box\neg q$							
$\Diamond(q \sqcup p)$							
$\Diamond\neg(q \sqcup p)$							
$\neg\Box(q \sqcup p)$							
$\neg\Box\neg(q \sqcup p)$							

Extensions/Abbréviations

$\sqcup_W$: Until *faible*

$w_k \models A \ \sqcup_W \ B$ ssi (1) ou (2)

(1) $\forall i \geq k : M, w_i \models A$

(2) $\exists j \geq k : M, w_j \models B$ et $\forall i \in [k, j[\ \ M, w_i \models A$

Prec : *Précédence*

$A \ Prec \ B$: *A précède B*

$w_k \models A \ Prec \ B$ ssi $\forall j \geq k$

$(w_j \models B) \Rightarrow (\exists i \in [k, j] : w_i \models A)$

IO Infiniment souvent

$IO \ A$: A possède une infinité d'occurences

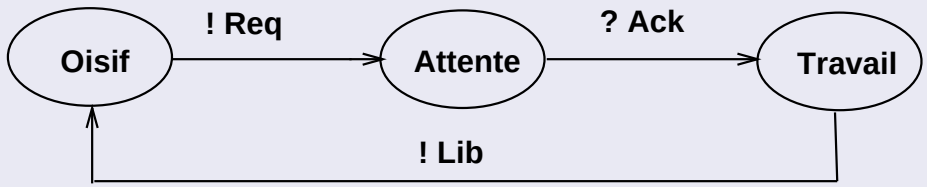
Exo #2 : Sucre ou non ?

Peut-on exprimer les modalités précédentes en LTL ?

Exemple de Spécification (1/3)

Problème d'allocation de ressources en exclusion mutuelle

Processus régis par le comportement ci-contre



Propriétés Attendues : Safety/Liveness, ...

1. Exclusion Mutuelle
2. Absence de Famine
3. Temps d'utilisation de la ressource borné
4. La ressource n'est accordée qu'aux processus l'ayant demandée

Exemple de Spécification : "Méthode"

Précision/Interprétation-Reformulation/Traduction

1. Préciser (faire repréciser) si besoin les exigences/propriétés.
2. Les exprimer en langue naturelle :
   1. avec les élèments* du modèle (états/événements/...)
   2. dans une forme a priori compatible avec la logique qui permettra de les formaliser (souvent plusieurs passes sont nécessaires)
3. Les traduire formellement dans la logique utilisée

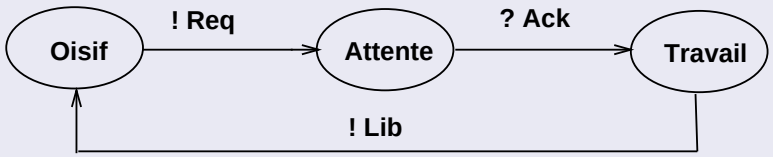
Le modèle à disposition ...

Vue Processus :

– Etats

– Communications

– Evénements



Risques et Précautions

Interpréter/Traduire : Opérations toujours risquées, ne pas hésiter donc à échanger/partager/discuter ...

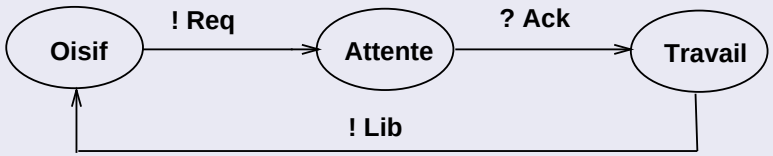
∗ Le modèle doit permettre une expression *aisée* de la propriété

TP#1 Systèmes Concurrents de 4IR : Comment vérifier l'utilisation en exclusion mutuelle de chaque ressource alors que le modèle est focalisé sur les processus. D'ailleurs, la propriété n'était pas garantie !

Exemple de Spécification : Reformulation

Le modèle à disposition . . .

Vue Processus :
– Etats/Evénements
– Communications



Précision/Interprétation-Reformulation

1. Exclusion Mutuelle
   Jamais plus de deux processus en train de travailler en même temps
2. Absence de Famine
   1. *Un processus ayant demandé à travailler accèdera en temps fini à la ressource*
      (version 1/2 événementielle - vue plus tard SE-LTL - TP)
   2. *Un processus en* `attente` *passera au bout d'un temps fini dans l'état* `travail` (*toujours hybride*)
   3. *Un processus en* `attente` *sera au bout d'un temps fini dans l'état* `travail`

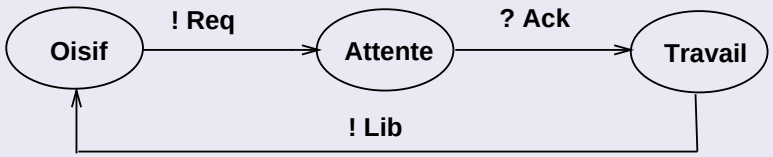
.../...

Quizz : Exprimer de façon *événementielle* la propriété d'exclusion mutuelle

Exemple de Spécification : Reformulation (suite)

Le modèle à disposition ...

Vue Processus :
– Etats/Evénements
– Communications



Précision/Interprétation-Reformulation

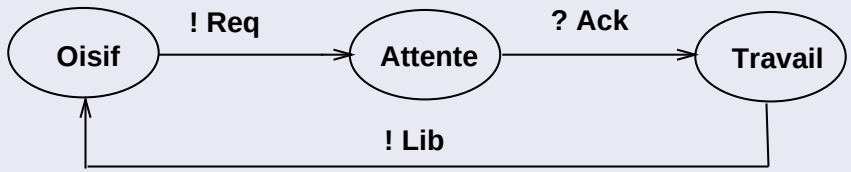
3. Temps d'utilisation de la ressource borné
   Aucune notion de ressource dans le modèle
   Adopter la vue processus :
   1. *Un processus ne reste pas indéfiniment dans l'état* `travail`
      A rapprocher de l'énoncé 2.3 du slide précédent
   2. *Un processus dans l'état travail sera au bout d' un temps fini dans un état non* `travail`
      Eviter de sur-spécifier !
   3. *Un processus dans l'état travail sera au bout d' un temps fini dans l'état* `oisif`

.../...

Exemple de Spécification : Reformulation (suite)

Le modèle à disposition ...

Vue Processus :
- Etats/Evénements
- Communications



Précision/Interprétation-Reformulation

1. La ressource n'est accordée qu'aux processus l'ayant demandée

   En adoptant la vue processus :
   - *Un processus ne travaille que s'il l'a expressément demandé*
   - *Un processus est dans l'état* `travail` *uniquement s'il a été auparavant dans l'état* `attente`

   PB : Enoncé formulé au passé alors que la logique ne prend en compte que le futur ..... mais $\exists$ une notion de précédence on va bien y arriver !
   - *Un processus est dans l'état* `travail` *s'il a été auparavant dans l'état* `attente`
   - Pour un process donné : l'état `attente` précède toujours l'état `travail`

Exemple de Spécification : Reformulation (Fin)

Récapitulatif

1. Exclusion Mutuelle
   Jamais plus de deux processus en train de travailler en même temps
2. Absence de Famine
   Un processus en `attente` *sera au bout d'un temps fini dans l'état* `travail`
3. Temps d'utilisation de la ressource borné
   Un processus dans l'état travail sera au bout d' un temps fini dans un état non `travail`

Notations - Simplification

On note A_i pour le processus i est en attente, O_i oisif et T_i travail
On suppose que l'on a deux processus 1 & 2

Traduction

1. Exclusion Mutuelle
   Jamais plus de deux processus en train de travailler en même temps
   $\Box\neg(T_1 \wedge T_2)$
   $\bigwedge_{i,j\in[1,k]:i\neq j} \Box\neg(T_i \wedge T_j)$ (cas général)
2. Absence de Famine
   Un processus en `attente` *sera au bout d'un temps fini dans l'état* `travail` $\Box(A_i \Rightarrow \Diamond T_i)$
   Inutile de surspécifier $\Box(A_i \Rightarrow (A_i \sqcup T_i))$
3. Temps d'utilisation de la ressource borné
   Un processus dans l'état travail sera au bout d'un temps fini dans un état non `travail` $\Box(T_i \Rightarrow \Diamond\neg T_i)$
4. La ressource n'est accordée qu'aux processus l'ayant demandée
   Pour un process donné : l'état `attente` *précède toujours l'état* `travail` $\Box(\neg T_i \Rightarrow A_i \ Prec \ T_i)$

Exo #3

En vous inspirant de la traduction de P#4, traduisez P#5
Les ressources sont accordées dans l'ordre où elles ont été demandées

De la modélisation à la vérification (LTL+CTL)

Du modèle à la structure de Kripke (SK)

Système concurrents/réactif le plus souvent non déterministe
Leur comportement est représenté par un "graphe" **"étiqueté"**

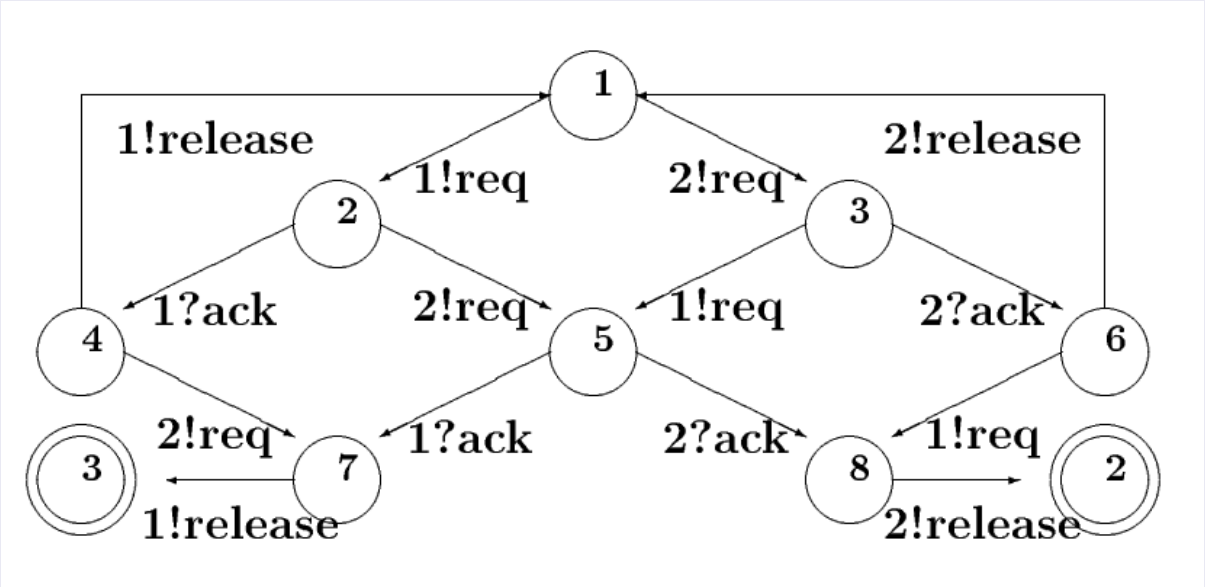
Structures de Kripke "standard" étiquettent les mondes (seulement)
Structures de Kripke "Etiquetées" étiquettent les sommets et les arcs

Formalisme	Comportement	Etiquetage
RdP	Graphe de marquages	arcs/sommets
Alg Proc	Labelled Trans. Syst.	communications
Réseaux d'automates Uppaal	Graphes	communications/ sommets

TP :

State/Event-LTL (SE-LTL) permet de prendre en compte les étiquettes sur les arcs et les sommets

SK : SKE associée à un Système d'allocation de ressources



Valuation des noeuds : ν

$\nu(1) = \{O_1, O_2\}$
$\nu(2) = \{A_1, O_2\}$
$\nu(3) = \{O_1, A_2\}$
$\nu(4) = \{T_1, O_2\}$
$\nu(5) = \{A_1, A_2\}$
$\nu(6) = \{O_1, T_2\}$
$\nu(7) = \{T_1, A_2\}$
$\nu(8) = \{A_1, T_2\}$

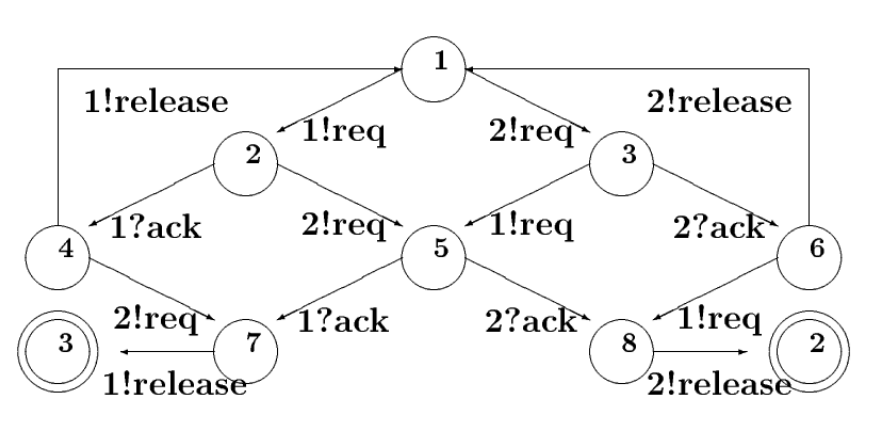
nb

Dualité Etats/Evénements : ici redondante
Infinité de séquences à traiter

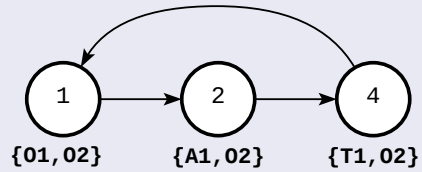
Rappel des Propriétés à vérifier ($\forall \imath \in [1,2]$)

- $P1 : \Box\neg(T_1 \wedge T_2)$
- $P2_i : \Box(A_i \Rightarrow \Diamond T_i)$
- $P3_i : \Box(T_i \Rightarrow \Diamond\neg T_i)$
- $P4_i : \Box(\neg T_i \Rightarrow A_i \ Prec \ T_i)$

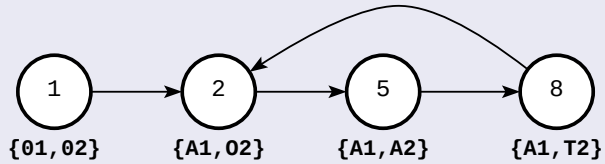
Le modèle et deux de ses traces (choisies arbitrairement parmi ∞)



Tr_0



Tr_1



Vérification sur les traces Tr_0 et Tr_1

$\models$	$P1$	$P2_1$	$P2_2$	$P3_1$	$P3_2$	$P4_1$
Tr_0	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
Tr_1	$\top$	$\bot$	$\top$	$\top$	$\top$	$\bot$

Bilan : $P2_i, P5_i$ sont contingentes

Intuition : $P1, P3_i, P4_i$ vraies pour toutes les traces possibles

Extension de $\models$ à des structures de KripKe générales

Extension de $\models$

Pour le moment, on a défini la sémantique de LTL ($\models$) sur des traces.
On étend $\models$ à une structure de Kripke générale de la façon suivante :

$$Sk \models \phi \quad ssi \quad \mathcal{E} \models \phi \quad \forall \mathcal{E} \in Tr(SK) \qquad \text{(i.e. } L(SK) \subset L(\phi)\text{)}$$

Vérification Traces Vs SK

$\models$	$P1$	$P2_1$	$P2_2$	$P3_1$	$P3_2$	$P4_1$
Tr_0	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
Tr_1	$\top$	$\bot$	$\top$	$\top$	$\top$	$\bot$
SK	?	$\bot$	?	?	?	$\bot$

On ne peut pas conclure à ce niveau sur $P1, P2_2, P3_1$ & $P3_2$

En fait on a bien

$\models$	$P1$	$P2_1$	$P2_2$	$P3_1$	$P3_2$	$P4_1$
SK	$\top$	$\bot$	$\bot$	$\top$	$\top$	$\bot$

Attention à la négation ! !

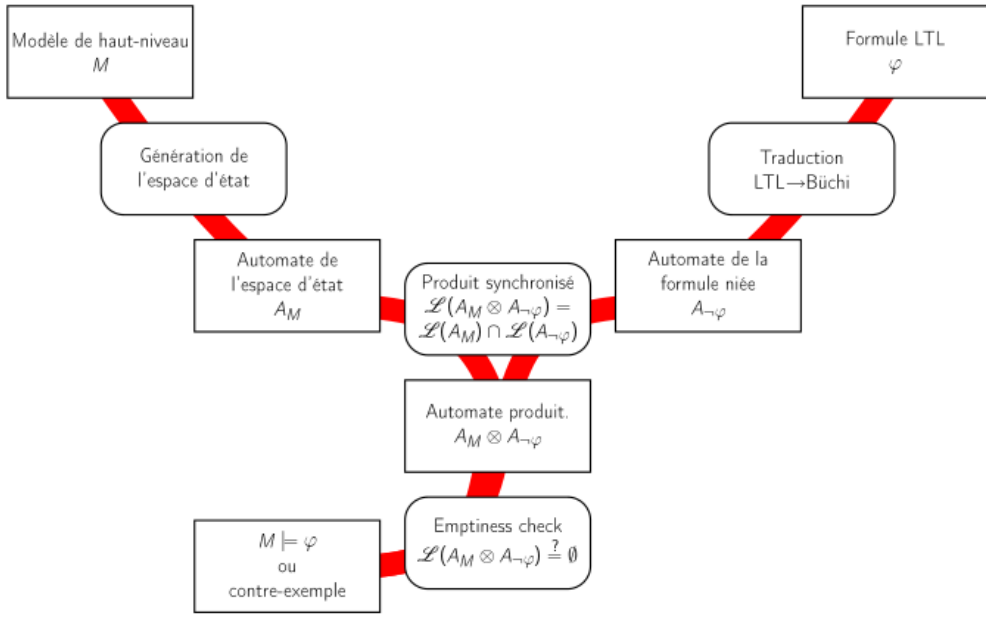
$SK \not\models \phi$ & $SK \not\models \neg\phi$ n'est pas contradictoire

Ainsi on a $SK \not\models P2_1$ à cause de Tr_1

et $SK \not\models \neg P2_1$ à cause de Tr_0

Principe du Model-Checking LTL (suite)

© A. Duret-Lutz

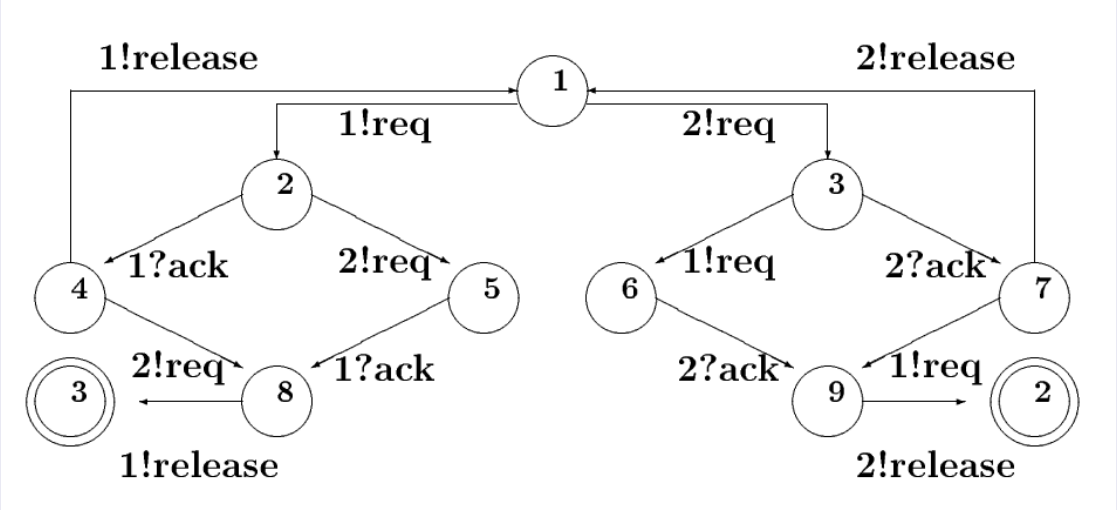


Principe

Idée générale : $SK \models \phi$ ssi $L(SK) \subset L(\phi)$ ou encore $L(SK) \cap \overline{L(\phi)} = \emptyset$
Pour obtenir $L(\phi)$, on construit son automate (de Buchi) accepteur (*)
Opérationnellement, on "teste le vide" du produit synchrone $SK \otimes B_{\neg\phi}$
Une séquence "acceptante" invalide la propriété et forme un contre-exemple

Complexité : $\mathcal{O}(|SK|.2^{|\phi|})$
Possibilité de construire à la volée l'espace d'états et le produit synchrone
(*) Par ex : `LTL2BA` permet d'obtenir un automate de Buchi via une formule LTL

SK' : Système d'allocation de ressources (version révisée)



Valuation des noeuds : ν

$\nu(1) = \{O_1, O_2\}$ $\quad$ $\nu(4) = \{T_1, O_2\}$ $\quad$ $\nu(7) = \{T_1, A_2\}$

$\nu(2) = \{A_1, O_2\}$ $\quad$ $\nu(5) = \{A_1, A_2\}$ $\quad$ $\nu(8) = \{T_1, A_2\}$

$\nu(3) = \{O_1, A_2\}$ $\quad$ $\nu(6) = \{A_1, A_2\}$ $\quad$ $\nu(9) = \{A_1, T_2\}$

Exo #4

1) Quelle est la principale différence entre SK et SK'

2) Renseigner le tableau suivant :

$\models$	$P1$	$P2_1$	$P2_2$	$P3_1$	$P3_2$	$P4_1$	$P4_2$	$P5_1$	$P5_2$
SK'									

SE-LTL

State/Event LTL (Shaki/Clarke/etc, CMU 2004)

SELT : Model-Checker LTL de Tina

Construction de l'automate de Buchi via LTL2BA (LIAFA-LSV)

Expressions SELT

Modalités : $\bigcirc$ (next), $\Box$ (always), $\Diamond$ (eventually), U (until)

propositions atomiques ou "entières" $\bigcirc P, \Box P \leq 4, ...$

propositions de transitions $\neg\Diamond\Box(Req1 \vee Ack1 \vee Lib1)$

Indication : Pensez aux Invariants de transitions dans les Petri !

Opérateurs arithmétiques et logico-arithmétiques :

$\Box(cs_1 + cs_2 + cs3 + cs4 \leq 1)$ *Exclusion mutuelle à n*

Mixité totale possible $\Box(t1 \Rightarrow (-(t3 \vee t4) \wedge (p2 \geq p3 + p4))\ U\ t2)$

Définition possible de nouveaux opérateurs

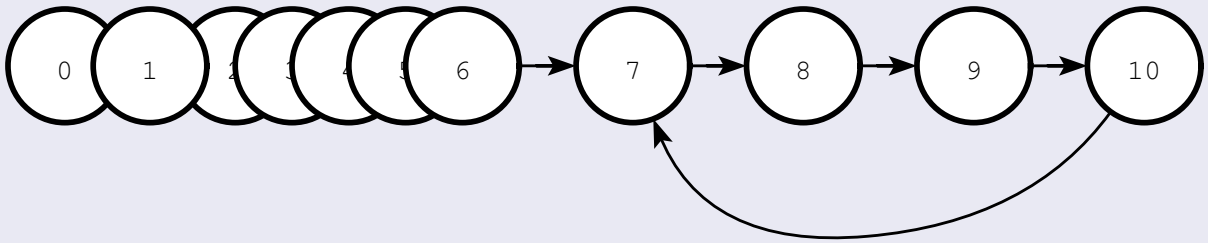
prefix F p = <> p;

infix Q responds P = G (P => F Q);

Model-checker SELT de Tina 2/3

Contre-Exemple abstrait

$-[]\ (wait\_1\ =>\ <>\ \ cs\_1);$
$FALSE$
$state0:\ \ idle\_1\ idle\_2\ idle\_3\ idle\_4\ token\_1$
$-\ 1\_ask...(preservingT)->$
$state25:\ \ after\_2\ idle\_3\ wait\_1\ wait\_2\ wait\_4$
$-\ 3\_ask...(preserving\ -\ cs\_1/\ \ wait\_1)->$
$*\ [accepting]state26:\ after\_2\ wait\_1\ wait\_2\ wait\_3\ wait\_4$
$-\ 3\_entry...(preserving\ -\ cs\_1)->$
$state26:\ \ after\_2\ wait\_1\ wait\_2\ wait\_3\ wait\_4\ state0:\ p1p2*2$



Possibilité d'obtenir le contre-exemple complet

Possibilité de le rejouer sur le stepper (y compris en temporisé)

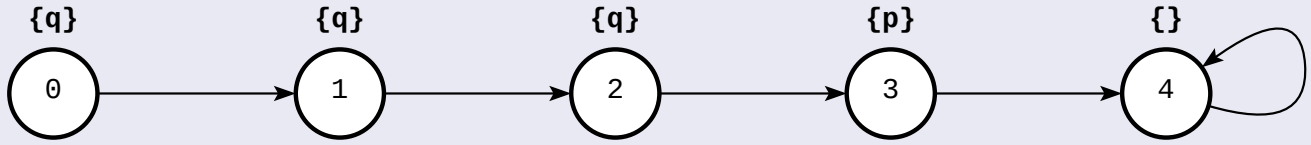
Mode “ligne de commande”

```
selt [-h | -help]
selt ktzfile [-f formula | formulafile] [-prelude file]
[-q | -v] [-b | -c | -p | -s | -g] [outfile]
```

FLAGS	WHAT
-h | -help	this mode
ktzfile	transition system input file
-f formula	pass formula to be checked as a string
formulafile	pass formula as a file (stdin if absent or -)
-prelude file	loads command file on entry
outfile	result file or output mode :
-b	just prints truth value
-c	prints summary of counter example
-p	prints counter example in full
-s	prints counter example loadable in stepper
-g	builds and prints full synchronized graph
-q	no banner nor times printed
-v	prints banner and exec times for commands

Solutions des exercices

Exo #1



$\models$	0	1	2	3	4	5	...
$q \sqcup p$	$\top$	$\top$	$\top$	$\top$	$\bot$	$\bot$	$\bot$
$\Diamond\Box\neg q$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
$\Diamond(q \sqcup p)$	$\top$	$\top$	$\top$	$\top$	$\bot$	$\bot$	$\bot$
$\Diamond\neg(q \sqcup p)$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
$\Box(q \sqcup p)$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$
$\neg\Box(q \sqcup p)$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$
$\Box\neg(q \sqcup p)$	$\bot$	$\bot$	$\bot$	$\bot$	$\top$	$\top$	$\top$
$\neg\Box\neg(q \sqcup p)$	$\top$	$\top$	$\top$	$\top$	$\bot$	$\bot$	$\bot$

Exo #2

$A \sqcup_W B \equiv \Box A \vee (A \sqcup B)$
$A \ Prec \ B \equiv (\neg B \sqcup_W A)$
$IO \ A \equiv \Box\Diamond A$

Exo #3

En vous inspirant de la traduction de P#4, traduisez P#5
Les ressources sont accordées dans l'ordre où elles ont été demandées
En posant P#5 $\equiv \Box(\phi \Rightarrow \psi)$, il reste à trouver :

a) $\phi \equiv$ "R_1 a été demandée avant R_2"

b) $\psi \equiv$ "R_1 sera obtenue avant R_2"

$\phi \equiv A_1 \wedge \neg(A_2 \vee T_2)$ Préférable à $A_1 \wedge O_2$

$\psi \equiv T_1 \ Prec \ T_2$ ou $\neg T_2 \sqcup_W T_1$ (cf exo #2)

Exo #4

Dans SK, l'état 5 - valué par {A1,A2} - est dupliqué en deux états - 5 et 6 - dans SK'. Dans ces états, les deux processus sont en attente mais on *a gardé en mémoire* que l'un avait demandé avant l'autre.
Dans SK', 5 (resp 6) correspond à un état où P1 (resp P2) est prioritaire,
Le contrôleur associé à SK' est donc équitable et la famine évitée.

$\models$	$P1$	$P2_1$	$P2_2$	$P3_1$	$P3_2$	$P4_1$
SK'	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$

Outline

Logique Temporelle Arborescente : introduction

CTL

Computation Tree Logic	Clarke - Emerson 1980
Conditional Tree Logic	Sifakis 1982

Plan

- Modèles d'Exécution
- Définition de CTL
- Spécification CTL du Système d'Allocation de ressources
- Algorithmes de Model Checking
  - Exploration avant (Clarke - Emerson)
  - Caractérisation par point fixe (Sifakis)

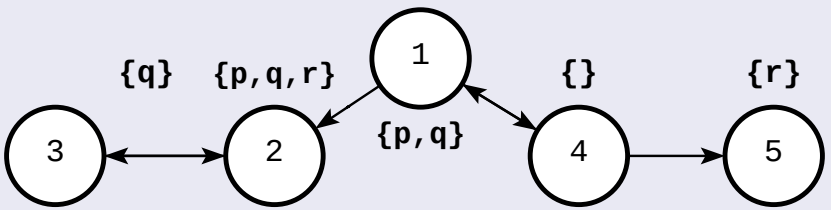
Modèle d'exécution

Structure de Kripke : $\mathcal{M} =< W, R, \nu >$

Chemins d'Exécution

- Une chemin de R est dit *terminal* ssi il ne peut pas être prolongé : $(s_0, s_1, \ldots s_n)$ est un chemin et $\nexists s \in S : s_n \ R \ s$
- Une chemin de R est dit *maximal* s'il est soit infini soit terminal.

Exemples :



A partir de l'état 1

- Terminaux : $(1, 4, 5), ((1, 4)^n, 5)$
- Infinis : $(1, 4)^\omega, ((1, 4)^n, 1, (2, 3)^\omega)$
- Maximaux : Infinis $\bigcup$ Terminaux

Infinité de Chemins Terminaux et Infinis

Modèle d'exécution (suite)

Potentialité Vs Inévitabilité

- "Une propriété sera dite **potentielle** s'**il existe** un chemin sur lequel elle se réalise "
- "Une propriété est **inévitable** si **tout** chemin **maximal** comporte un état où elle se réalise "

Remarque

La notion de chemin maximal permet de travailler avec une relation R non forcément totale.
On ne sera pas obligé d'ajouter des boucles sur les états de blocage comme on l'a fait pour LTL

Définition de CTL

Ingrédients

Ensemble de variable propositionnelles $\mathcal{P}$
$+$ Connecteurs classiques (Logique Propositionnelle)
$+$ *Pot*, *Inev* 2 connecteurs binaires notés en **préfixe**

Syntaxe

Pour $p \in \mathcal{P}, f\&g \in CTL(\mathcal{P})$ on a – $p \in CTL(\mathcal{P}) \quad \forall p \in \mathcal{P}$ – Si $f \in CTL(\mathcal{P})$ Alors $\neg f \in CTL(\mathcal{P})$ – Si $f, g \in CTL(\mathcal{P})$ Alors
$f \wedge g, Pot\ f\ g, Inev\ f\ g \in CTL(\mathcal{P})$

Variantes

Pot f g se note aussi f EU g (ou encore EF $[f$ U $g]$)
Inev f g se note aussi f AU g (ou encore AF $[f$ U $g]$)

Pot/Inev (Sifakis), EU/AU (Clarke-Emerson), EF/AF semble s'imposer
Souvenirs $E <> / A <>$ (Uppaal)

Sémantique de CTL

Définition de $\models$

Pour $p \in \mathcal{P}$ & $f, g \in CTL(\mathcal{P})$

- $\mathcal{M}, \omega_0 \models p$ ssi $p \in \nu(\omega_0)$ $(\forall p \in \mathcal{P})$
- $\mathcal{M}, \omega_0 \models \neg f$ ssi Non $(\mathcal{M}, \omega_0 \models f)$
- $\mathcal{M}, \omega_0 \models f \wedge g$ ssi $\mathcal{M}, \omega_0 \models f$ et $\mathcal{M}, \omega_0 \models g$
- $\mathcal{M}, \omega_0 \models Pot\ f\ g$ ssi $\exists$ un chemin d'origine ω_0 $(\omega_0, \omega_1, \ldots \omega_n)$, $\exists k \geq 0 : \mathcal{M}, \omega_k \models g$ et $\forall j \in [0, k[: \mathcal{M}, \omega_j \models f$
- $\mathcal{M}, \omega_0 \models Inev\ f\ g$ ssi $\forall$ chemin **maximal** d'origine ω_0 $(\omega_0, \omega_1, \ldots \omega_n, \ldots)$, $\exists k \geq 0 : \mathcal{M}, \omega_k \models g$ et $\forall j \in [0, k[: \mathcal{M}, \omega_j \models f$

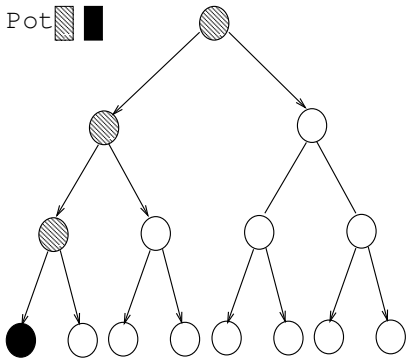
LTL Vs CTL : 2 logiques d'expressivité incomparables

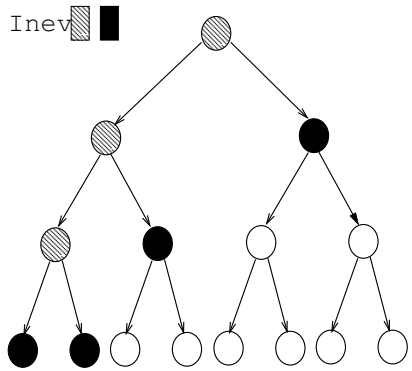
Rappel : $w_k \models f \sqcup g$ ssi $\exists j \geq k : M, w_j \models g$ et $\forall i \in [k, j[\ \ M, w_i \models f$

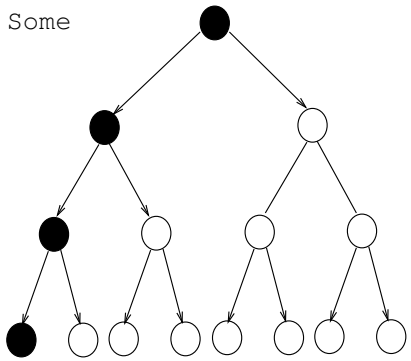
$\sqcup$ analogue de *Inev* au sens que si $f \sqcup g$ alors g sera forcément vraie sur la trace considérée

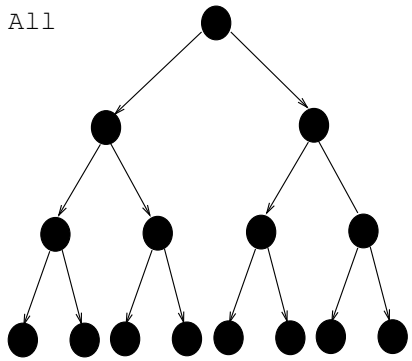
Pot, quant à lui, n'est pas exprimable en LTL mais l'équité $(\Box\Diamond f)$, exprimable en LTL, n'est pas exprimable en CTL

Intuition des modalités









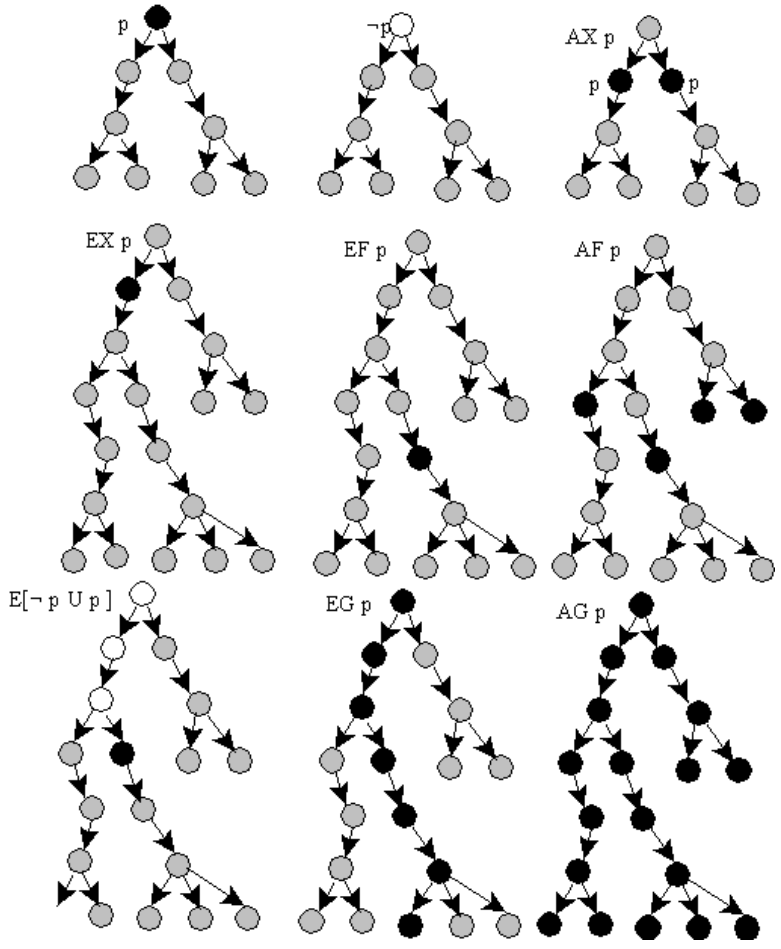
Abbréviations

$$Pot\ A \equiv Pot\ True\ A \qquad All\ A \equiv \neg\ Pot \neg\ A$$
$$Inev\ A \equiv Inev\ True\ A \qquad Some\ A \equiv \neg\ Inev \neg\ A$$

Variantes de notations (Sifakis/Clarke-Emerson) – "Uppaal"

Pot	EF	$E <>$	Some	EG	$E[]$
Inev	AF	$A <>$	ALL	AG	$A[]$

Exemples



Next-time operators

EX, *AX* : versions CTL du next-time opérateur ◯ de LTL
A rapprocher de ◊ et de □ de la logique modale

Exemple de Spécification : même Spec mais en CTL

Rappel des Propriétés et de leur traduction LTL $(\forall \imath \in [1,2])$

- Exclusion Mutuelle
  $P1 : \Box\neg(T_1 \wedge T_2)$
- Absence de Famine
  $P2_i : \Box(A_i \Rightarrow \Diamond T_i)$
- Temps d'utilisation borné
  $P3_i : \Box(T_i \Rightarrow \Diamond\neg T_i)$
- Ressource accordée uniquement si elle a été demandée
  $P4_i : \Box(\neg T_i \Rightarrow A_i \ Prec \ T_i)$

P1 : Simple propriété d'Accessibilté $P1 : AG\neg(T_1 \wedge T_2)$

P2, P3 : Même forme ("Leadsto")

$P2_i : AG\ (A_i \Rightarrow AF\ T_i)$ $\quad$ $P3_i : AG\ (T_i \Rightarrow AF\ \neg T_i)$
Quid de $AG\ (\phi \Rightarrow EF\ \psi)$?

Exo #1 : Exprimer P4 en CTL

Pas de *Prec* ni de $\sqcup_W$ mais on dispose de Pot !
Indication : Pensez à $Pot\ \phi\ \psi \ldots Pot\ \neg\phi\ \psi \ldots \neg Pot\ \neg\phi\ \psi$
$P4_i$:

Algorithme de Model-Checking "Avant" (Clarke-Emerson)

Expression de *Pot* et *Inev* à l'aide de *Ex* et *Ax*

Récursive

$Pot\ A\ B\ "\equiv"\ B \vee (A \wedge Ex\ Pot\ A\ B)$

$Inev\ A\ B\ "\equiv"\ B \vee (A \wedge Ax\ Inev\ A\ B)$

Dérécursivée

$Pot^0\ A\ B \equiv B$

$Pot^k\ A\ B \equiv B \vee (A \wedge Ex\ Pot^{k-1}\ A\ B)$

$Inev^0\ A\ B \equiv B$

$Inev^k\ A\ B \equiv B \vee (A \wedge Ax\ Inev^{k-1}\ A\ B)$

Propriétés (dans tout modèle fini)

$M, w \models Pot\ A\ B$ ssi $\exists k : M, w \models Pot^k\ A\ B$

$M, w \models Inev\ A\ B$ ssi $\exists k : M, w \models Inev^k\ A\ B$

Il suffit de prendre $k = Card(S) + 1$

Complexité Modale $\mathcal{C} : CTL(\mathcal{P}) \mapsto \mathbb{N}$ définie par :

$\mathcal{C}(p) = 0 \quad \forall p \in \mathcal{P}$
$\mathcal{C}(\neg f) = \mathcal{C}(f)$
$\mathcal{C}(f \wedge g) = Max(\mathcal{C}(f), \mathcal{C}(g))$
$\mathcal{C}(\wr\ f\ g) = Max(\mathcal{C}(f), \mathcal{C}(g)) + 1$

Sous-formules positives

$Sub^+ : CTL(\mathcal{P}) \mapsto \mathcal{P}(CTL(\mathcal{P}))$ définie par :
$Sub^+(p) = \{p\} \quad \forall p \in \mathcal{P}$
$Sub^+(f \wedge g) = \{f \# g\} \bigcup Sub^+(f) \bigcup Sub^+(g)$
$Sub^+(\wr\ f\ g) = \{\wr\ f\ g\} \bigcup Sub^+(f) \bigcup Sub^+(g)$
$Sub^+(\neg\ f) = Sub^+(f)$

Stratification de Sub^+

$$Sub^+(f) = \bigcup_{n \geq 0} {Sub^+}_n(f) \quad \text{où} \quad {Sub^+}_n(f) = Sub^+(f) \cap \mathcal{C}^{-1}(n)$$

Algorithme "en avant"

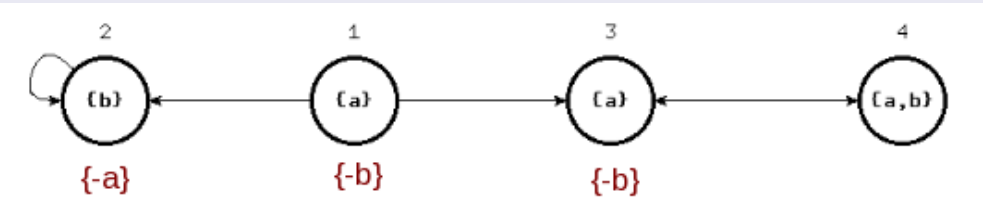
For i $\in [1, \mathcal{C}(f_0)]$ do
$F \leftarrow Sub^+{}_i(f_0)$
$\forall s \in S$ do
$\forall f \in F$ do
If $f = Pot\ A\ B$ and $B \in set(s)$
Or $f = Inev\ A\ B$ and $B \in set(s)$
Or $f = A \wedge B$ and $(A \in set(s)$ and $B \in set(s))$
Then $set(s) \leftarrow set(s) \cup \{f\}$
end
end
For j $\in [0, Card(S)]$ do
$\forall s \in S$ do
$\forall f \in F$ do
If $f = Pot\ A\ B$ and $A \in set(s)$ and $\exists t : s\ R\ t\ \&\ f \in set(t)$
Or $f = Inev\ A\ B$ and $A \in set(s)$
and $(\exists t : s\ R\ t)\ \&\ (\forall t : s\ R\ t \Rightarrow f \in set(t))$
Then $set(s) \leftarrow set(s) \cup \{f\}$
end
end
end
$\forall s \in S$ do
$\forall f \in F$ do If $f \notin set(s)$ then $set(s) \leftarrow set(s) \cup \{\neg\ f\}$
end end end

Exemple d'exécution

La formule

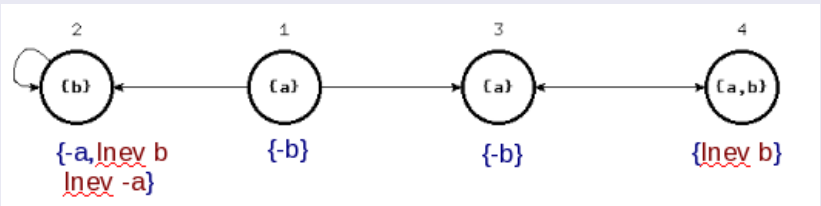
$f = InevB \wedge SomeA\ (\equiv Inev\ B \wedge \neg Inev\ \neg\ A\ )$
$Sub^{+}(f) = \{A, B, Inev\ B, Inev\ \neg\ A, f\}$

La structure



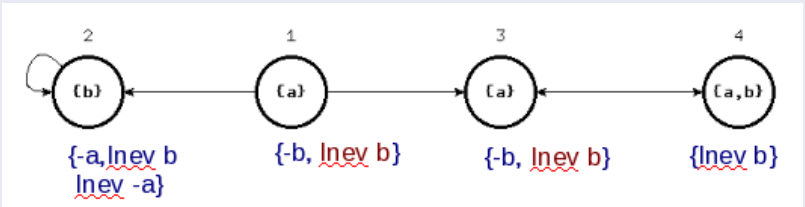
Ordre de parcours des états : $1 \mapsto 2 \mapsto 3 \mapsto 4$

$i = 2$, Phase A $(Inev\ B, Inev\ \neg\ A)$

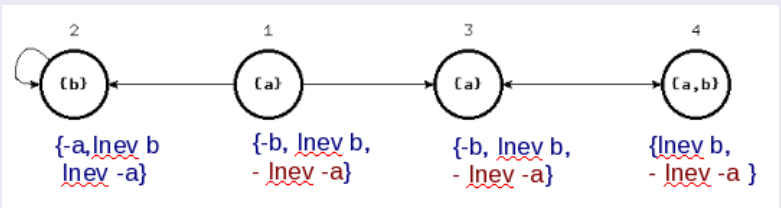


Exemple d'exécution (suite)

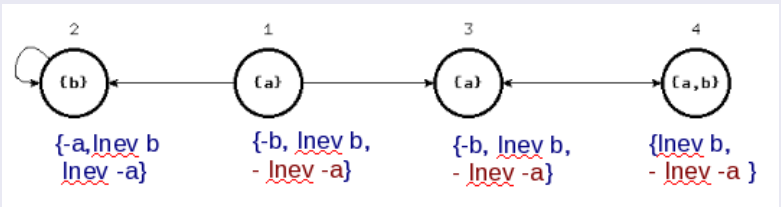
$i = 2$, Phase B ($Inev\ B, Inev\ \neg\ A$)



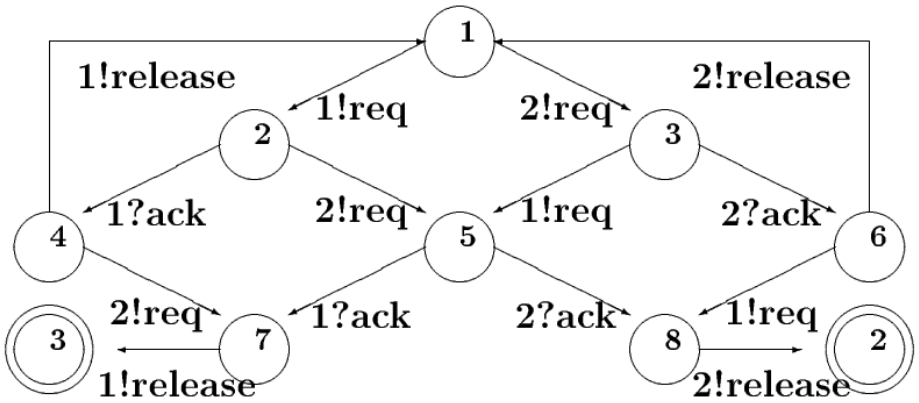
$i = 2$, Phase C ($Inev\ B, Inev\ \neg\ A$)



$i = 3$, Phase C (f)



Pb Allocation de Ressources - version non équitable



La formule

$f = A_1 \Rightarrow Pot\ T_1$

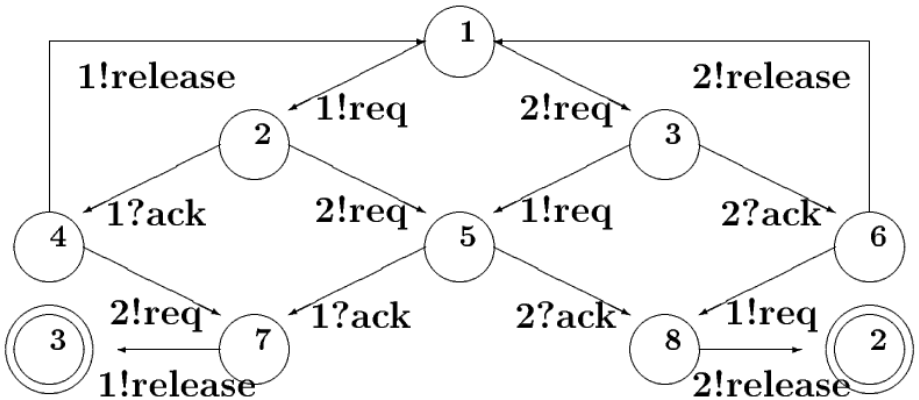
$Sub^+(f) = \{A_1, T_1, Pot\ T_1, f\}$

Ordre de parcours : $1 \mapsto 2 \mapsto 3 \ldots \mapsto 8$

Evaluation

	I	A	B_1	B_2	B	C	f
1				$Pot\ T_1$ (2)	””	””	ok
2	A_1		$Pot\ T_1$ (4)	””	””	””	ok
3				$Pot\ T_1$ (5)	””	””	ok
4	T_1	$Pot\ T_1$	””	””	””	””	ok
5	A_1		$Pot\ T_1$ (7)	””	””	””	ok
6				$Pot\ T_1$ (8,1)	””	””	ok
7	T_1	$Pot\ T_1$	””	””	””	””	ok
8	A_1		$Pot\ T_1$(2)	””	””	””	ok

Pb Allocation de Ressources - version non équitable



La formule

$f = A_1 \Rightarrow Inev\ T_1$

$Sub^+(f) =$

$\{A_1, T_1, Inev\ T_1, f\}$

Ordre de parcours : $1 \mapsto 2 \mapsto 3 \ldots \mapsto 8$

Evaluation

	I	A	B_1	B_2	B	C	f
1							
2							
3							
4							
5							
6							
7							
8							

Algorithme d'évaluation par Point Fixe (Sifakis)

Principe :

Considérer une application qui associe à toute formule de CTL sa *théorie* (dans une structure de Kripke donnée)

$$TH : CTL \mapsto \mathcal{P}(S)$$
$$TH(f) = \{s \in S : M, s \models f\}$$

Note : A l'inverse, l'algorithme "en avant" associait à chaque état l'ensemble des sous-formules qu'il satisfaisait.

Pour l'aspect propositionnel

$$TH(p) = \{s \in S : p \in \nu(s)\}$$
$$TH(\neg F) = S \setminus TH(F)$$
$$TH(F \wedge G) = TH(F) \bigcap TH(G)$$
$$TH(F \vee G) = TH(F) \bigcup TH(G)$$
...

Pour les modalités, trouver f et f' :

$$TH(Pot\ F\ G) = f(TH(F), TH(G))?$$
$$TH(Inev\ F\ G) = f'(TH(F), TH(G))?$$

Algorithme d'évaluation par Points Fixes (En arrière)

Opérateurs Pre et $\overline{Pre} : \mathcal{P}(\mathcal{S}) \mapsto \mathcal{P}(\mathcal{S})$

$Pre(U) = \{t \in S : t \to s \text{ et } s \in U\}$
$\overline{Pre}(U) = \{t \in S : t \to s \text{ et } (t \to s \Rightarrow s \in U)\}$

Calcul de Pot et Inev comme points fixes de fonctions monotones

Rappel : Toute fonction monotone dans un treillis complet admet un point fixe
$\exists x_0 : f(x_0) = x_0$ (Croissante/plus petit - Décroissante/Plus grand)

Soit $f_{Pot} : \mathcal{P}(S) \mapsto \mathcal{P}(S)$
$f_{Pot}(x) = x \bigcup (TH(F) \bigcap Pre(x))$

Soit $f_{Inev} : \mathcal{P}(S) \mapsto \mathcal{P}(S)$
$f_{Inev}(x) = x \bigcup (TH(F) \bigcap \overline{Pre}(x))$

$\mathcal{P}(S)$ est un treillis complet, f_{Pot}, f_{Inev} croissantes donc admettent un +petit pf
$TH(Pot\ F\ G)$ = Plus petit point fixe de f_{Pot} $\quad \nu X.f_{Pot}(X)$
$TH(Inev\ F\ G)$ = Plus petit point fixe de f_{Inev} $\quad \nu X.f_{Inev}(X)$

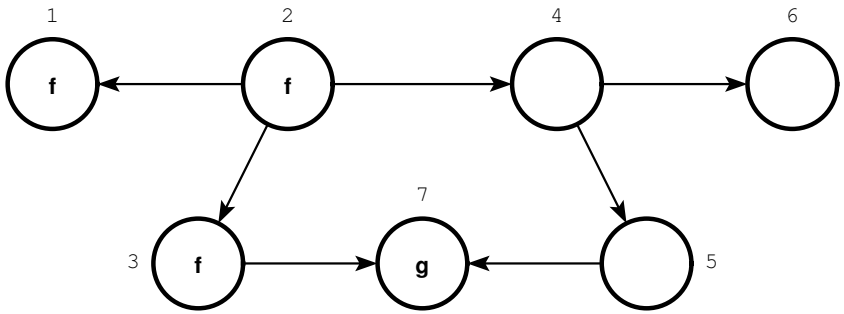
Calcul de $TH(Pot\ F\ G)$

$TH(Pot\ F\ G) = \bigcup_{n \geq 0} X_n$
où $X_0 = TH(G)$ et
$X_k = (TH(F) \bigcap Pre(X_{k-1})) \bigcup X_{k-1}$

Calcul de $TH(Inev\ F\ G)$

$TH(Inev\ F\ G) = \bigcup_{n \geq 0} Y_n$
où $Y_0 = TH(G)$ et
$Y_k = (TH(F) \bigcap \overline{Pre}(Y_{k-1})) \bigcup Y_{k-1}$

Exemple d'évaluation de $TH(Pot\ f\ g)$



Préliminaires

$TH(g) = \{7\}$
$TH(f) = \{1, 2, 3\}$
$X_0 = TH(g) = \{7\}$

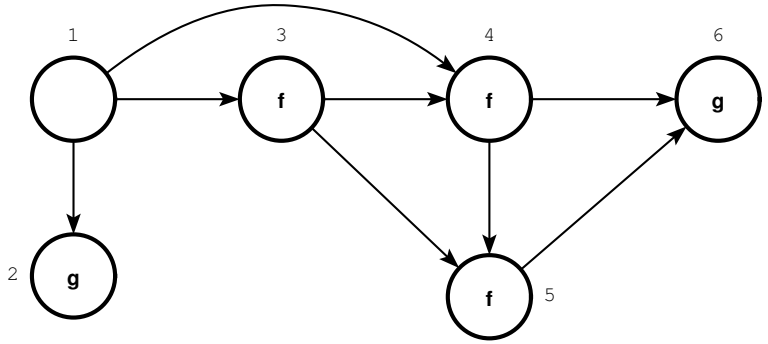
Itérations

$$
\begin{aligned}
X_1 &= (\{1,2,3\} \cap Pre(\{7\})) \cup \{7\} \\
&= (\{1,2,3\} \cap \{3,5\}) \cup \{7\} \\
&= \{3,7\} \\
X_2 &= (\{1,2,3\} \cap Pre(\{3,7\})) \cup \{3,7\} \\
&= (\{1,2,3\} \cap \{2,3,5\}) \cup \{3,7\} \\
&= \{2,3,7\} \\
X_3 &= (\{1,2,3\} \cap Pre(\{2,3,7\})) \cup \{2,3,7\} \\
&= (\{1,2,3\} \cap \{2,3,5\}) \cup \{2,3,7\} \\
&= \{2,3,7\}
\end{aligned}
$$

Point Fixe

$X_3 = X_2 \Rightarrow TH(Pot\ f\ g) = \{2,3,7\}$

Exemple d'évaluation de $TH(Inev\ f\ g)$



Préliminaires

$TH(g) = \{2, 6\}$
$TH(f) = \{3, 4, 5\}$
$X_0 = TH(g) = \{2, 6\}$

Itérations

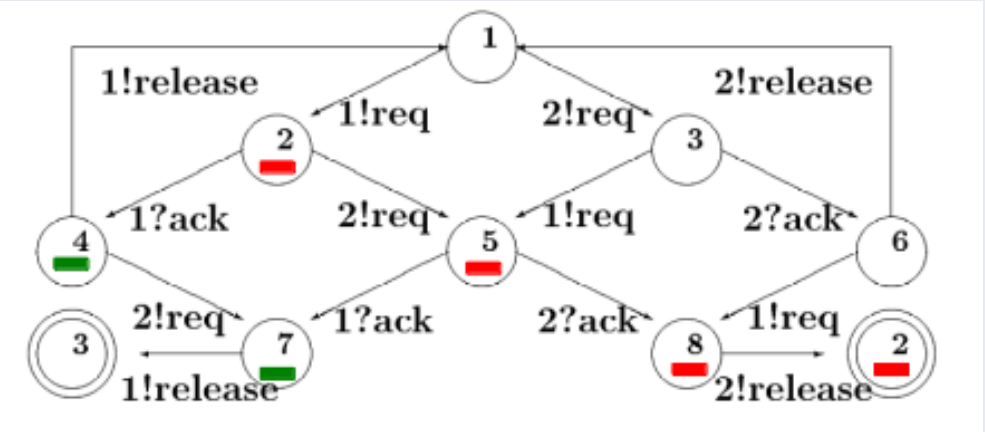
$$
\begin{aligned}
X_1 &= (\{3,4,5\} \cap \overline{Pre}(\{2,6\})) \cup \{2,6\} \\
&= (\{3,4,5\} \cap \{5\}) \cup \{2,6\} \\
&= \{2,5,6\} \\
X_2 &= (\{3,4,5\} \cap \overline{Pre}(\{2,5,6\})) \cup \{2,5,6\} \\
&= \{3,4,5\} \cap \{4,5\}) \cup \{2,5,6\} \\
&= \{2,4,5,6\} \\
X_3 &= (\{3,4,5\} \cap \overline{Pre}(\{2,4,5,6\})) \cup \{2,4,5,6\} \\
&= \{2,3,4,5,6\} \qquad \text{car } \overline{Pre}(\{2,4,5,6\}) = \{3,4,5\} \\
X_4 &= (\{3,4,5\} \cap \overline{Pre}(\{2,3,4,5,6\})) \cup \{2,4,5,6\} \\
&= \{2,3,4,5,6\} \qquad \text{car } \overline{Pre}(\{2,3,4,5,6\}) = \{1,3,4,5\} \text{ mais } 1 \not\models f
\end{aligned}
$$

Point Fixe

$X_4 = X_3 \Rightarrow TH(Inev\ f\ g) = \{2, 3, 4, 5, 6\}$

Retour sur le pb d'allocation de ressources (non équitable)

Modèle et Propriétés



A Evaluer

$P21'$: $All\ (A_1 \Rightarrow Pot\ T_1)$

Absence de Famine

$P2_1$: $All\ (A_1 \Rightarrow Inev\ T_1)$

Calcul de $TH(Pot\ F)$

$TH(Pot\ F) = \bigcup_{n\geq 0} X'_n$

où $X'_0 = TH(F)$ et

$X'_k = Pre(X'_{k-1}) \bigcup X'_{k-1}$

Calcul de $TH(Inev\ F)$

$TH(Inev\ F) = \bigcup_{n\geq 0} Y'_n$

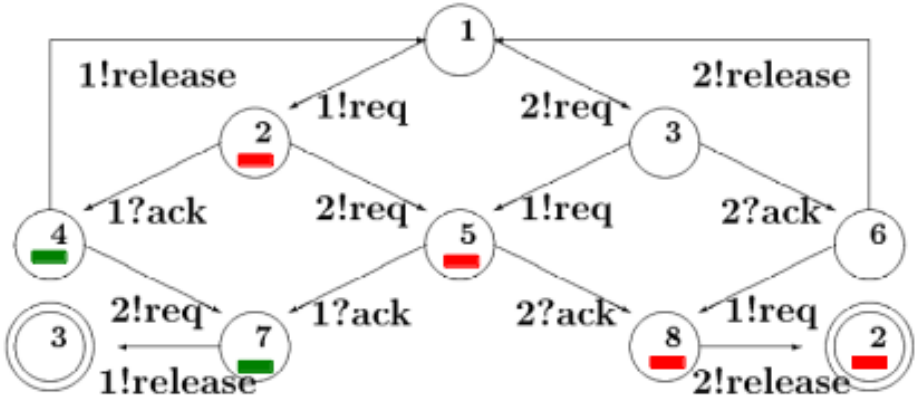
où $Y'_0 = TH(F)$ et

$Y'_k = \overline{Pre}(Y'_{k-1}) \bigcup Y'_{k-1}$

Etapes intermédiaires à réaliser

1) Calculer $TH(A_1)$, $TH(T_1)$, $TH(Pot\ T_1)$, $TH(Inev\ T_1)$
2) Appliquer $TH(\phi \Rightarrow \psi) = (S \setminus TH(\phi)) \cup TH(\psi)$
   pour obtenir $TH(A_1 \Rightarrow Pot\ T_1)$ et $TH(A_1 \Rightarrow Inev\ T_1)$
3) Appliquer $TH(All\ \phi) = S \setminus TH(Pot\ \neg\phi)$

Retour sur le pb d'allocation de ressources (suite)



$P21'$: $All\ (A_1 \Rightarrow Pot\ T_1)$
Absence de Famine
$P2_1$: $All\ (A_1 \Rightarrow Inev\ T_1)$
$TH(A_1) = \{2, 5, 8\}$
$TH(T_1) = \{4, 7\}$

$TH(Pot\ T_1)$

$X'0 = \{4, 7)$
$X'_1 = \{2, 5, 8, 4, 7\}$
$X'_2 = \{1, 3, 6, 2, 5, 8, 4, 7\}$
$X'_2 = S$ donc $TH(Pot\ T_1) = S$

$TH(Inev\ T_1)$

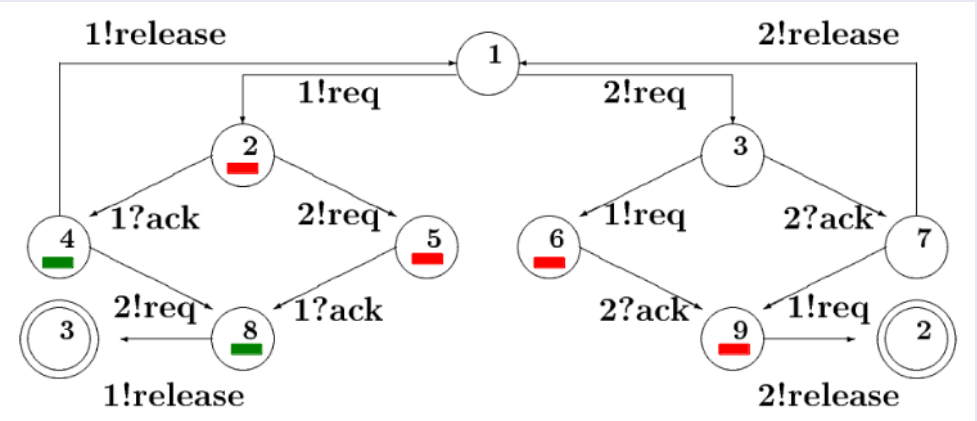
$Y'_0 = \{4, 7)$
$Y'_1 = \{4, 7\}$ car $\overline{Pre}(\{4, 7\}) = \emptyset$
$Y'_1 = Y'_0$ donc
$TH(Inev\ T_1) = \{4, 7\}$

Suite

$TH(A_1 \Rightarrow Pot\ T_1) = (S \setminus \{2, 5, 8\}) \wedge S = S$
$TH(\neg Pot\ \neg"S") = S$ donc $TH(All\ ((A_1 \Rightarrow Pot\ T_1)) = S$
$TH(A_1 \Rightarrow Inev\ T_1) = (S \setminus \{2, 5, 8\}) \wedge \{4, 7\} = \{1, 3, 6, 4, 7\}$
$TH(\neg Pot\ \neg"\{1, 3, 6, 4, 7\}")$, $TH(\neg Pot\ "\{2, 5, 8\}") = TH(\neg"S") = \emptyset$
et $TH(All\ (A_1 \Rightarrow Inev\ T_1)) = \emptyset$

Pb d'allocation de ressources (version équitable)

Exo #2



Evaluez $P2_1$

$P2_1 : AG\ (A_1 \Rightarrow Inev\ T_1)$

en utilisant l'algo "avant" et l'algo par point fixe.

Exo #3

Proposez une formule CTL permettant de caractériser les états origines d'une séquence où un des processus ($Proc_1$ par ex) est en situation de famine. Vous évaluerez ensuite cette formule sur les deux versions : équitable et non équitable.

Compléments

Définition de $\models$ sur SK

On a défini jusqu'à présent $Sk, w \models \phi$
2 extensions possibles pour étendre $\models$ aux SK

1. $Sk \models \phi$ ssi $Sk, s \models \phi$ $(\forall s \in S)$
   Dans ce cas, même attention pour la négation que pour LTL
2. Pour les Sk dotées d'un état initial s_0
   $Sk \models \phi$ ssi $Sk, s_0 \models \phi$

Suivant les model-checkers, on a accès aux différentes variantes.
Dommage que certains ne proposent que (2)

Complexité

CTL : $\mathcal{O}(|SK|.|\phi|)$ rappel LTL : $\mathcal{O}(|SK|.2^{|\phi|})$

Tps TINA-Muse

Muse : Model-checker pour le μ-calcul modal (Kozen)

Le μ-calcul est une extension de la logique temporelle basée sur la définition de points fixes imbriqués.
Pour ce TP, on utilisera uniquement la bibliothèque CTL fournie par Muse.

Bibliothèque CTL (ctl.mmc)

```
# Netx-Time Operators
prefix EX g = <T>g;
prefix AX g = [T]g;
# CTL. (All | Some) state along (Every | Some) path
prefix AF g = min x | g \/ (<T>T /\ [T]x);          ## Inev
prefix wAF g = min x | g \/ [T]x;                   ## "weak inev"
prefix EF g = min x | g \/ <T>x;                    ## Pot
prefix AG g = max x | g /\ [T]x;                    ## All
prefix EG g = max x | g /\ ([T]F \/ <T>x);          ## Some
infix f EU g = min x | g \/ (f /\ <T>x);
infix f AU g = min x | g \/ (f /\ <T>x /\ [T]x);
# Extras
op dead = - <T>T;  # deadlocks
op LIVE l = EF (<l>T); # Petri Net liveness
```

Muse & Pathto

Manuel Muse (extrait) (projects.laas.fr/tina//manuals/muse.html)

```
muse ktzfile [-f formula | formfile] [-prelude mmcfile]
[-q | -v] [-b | -c | -s] [-wp n] [outfile] [errorfile]
FLAGS          WHAT
ktzfile        transition system input file
-f formula     pass formula to be checked as a string
formulafile    pass formula as a file (stdin if absent or -)
-prelude file loads command file on entry
outfile        result file or -
errorfile      error file (stderr if - or absent)
output mode: -s sets | -c sets cardinality (def) | -b truth value
.../...
```

Manuel Pathto (extrait) (projects.laas.fr/tina//manuals/pathto.html)

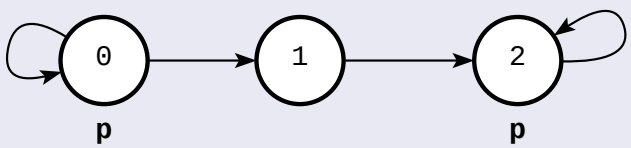
```
pathto tgt [-from src] [-p | -t | -s | -l]
[infile] [outfile] [errorfile]
Source and destination states:
tgt : target state  /  -from src source state (default 0)
Output format flags:
-p (state/transition)* | -t (transition)* - cf stepper
-s (state)* | -l Prints the length of a path from src to tgt.
```

Outline

LTL, CTL deux logiques complémentaires (incomparables)

CTL $\not\geq$ LTL

Equité en LTL : $\Box \Diamond \phi$
CTL ne permet pas d'exprimer l'équité !

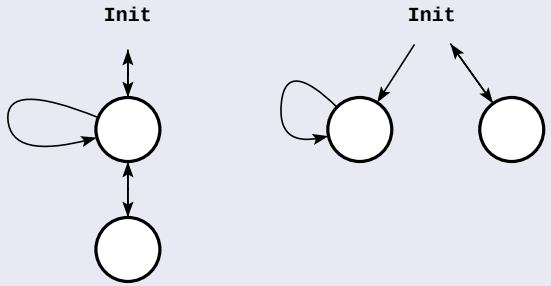


Autre exemple ci-contre

$SK \models \Diamond \Box p$ et $SK \not\models Inev\ All\ p$

LTL $\not\geq$ CTL

LTL ne permet pas d'exprimer *Pot*
voir ex ci-contre



$Sk_{\text{Gauche}} \models All\ Pot\ Init$ tandis que $Sk_{\text{Droite}} \not\models All\ Pot\ Init$
$L(Sk_{\text{Gauche}}) = L(Sk_{\text{Droite}})$ et sont donc indistinguables pour LTL

Autres logiques

CTL* combine CTL et LTL
μ-calcul TP `Muse` : moteur μ-calcul avec une interface CTL

LTL/CTL Vs Uppaal : $L_{Uppaal} \subset (LTL \cap CTL)$

Syntaxe L_{Uppaal}

$Prop ::= \ 'A[]'\ Expression \mid 'E <>'\ \ Expression \mid 'E[]'\ Expression$
$\mid 'A <>'\ \ Expression \mid Expression\ ' - - >'\ Expression$

Pas d'imbrication de formules !

Sémantique L_{Uppaal} : double quantification chemins/états

A/E	Chemins	$(\forall/\exists)$
$[]/<>$	Etats	$(\forall/\exists)$

Evaluation à la racine

$L_{Uppaal} \subset (LTL \cap CTL)$

Uppaal	*CTL*	$(LTL, \{True, False\})$
$A[]\phi$	*All* ϕ	$(\Box\phi, True)$
$A <> \phi$	*Inev* ϕ	$(\Diamond\phi, True)$
$E <> \phi$	*Pot* ϕ	$(\Box\neg\phi, False)$
$E[]\phi$	*Some* ϕ	$(\neg\Box\phi, False)$
$phi - - > \psi$	*All* $(\phi \Rightarrow$ *Inev* $\psi)$	$(\Box\ (\phi \Rightarrow \Diamond\ \psi), True)$

Patterns de Propriétés

Objectifs

Offrir une interface de plus haut niveau que la logique temporelle pour l'expression des propriétés les plus usuelles.
Leur donner une traduction dans les principales logiques pour en permettre la vérification par des model-checkers.

Patterns de M. Dwyer

patterns.projects.cis.ksu.edu/documentation/patterns
Patterns structurés en 8 "classes" dont :
Absence, Existence, Universality, Precedence, Response, ...
Traduction fournie en LTL, CTL, GIL, QRE, ACTL, μ-calcul

Exemples : Absence P is False

	CTL	*LTL*
Globally	$AII\ \neg P$	$\Box\ \neg P$
After Q	$AII\ (Q \Rightarrow AII\ \neg P)$	$\Box(Q \Rightarrow \Box\neg P)$

Suite en Tp ....

Procédures de Décision pour LTL et CTL

Propriété de Modèle Fini

Une logique $\mathcal{L}$ possède la *pmf* ssi $\forall \phi \in \mathcal{L}, \exists SK \ \textit{finie} \ \ tq \ SK \models \phi$
LTL et CTL ont la Propriété de Modèle Fini

Procédure de décision : Synthèse de contrôleur

Pour une logique possédant la *pmf*, une procédure de décision permet de construire un modèle fini pour toute formule satisfiable

$\mapsto$ **Synthèse**

nb : la version équitable de la SK du pb d'allocation de ressources peut-être synthétisée en utilisant la spécification que nous avons étudiée.
En général, basée sur la méthode des "tableaux". Complexité : $\mathcal{O}(2^{|\phi|})$

TWB : The Tableau Workbench

Implémente les procédures de décision notamment pour LTL, CTL, ...

test en ligne via twb.rsise.anu.edu.au

Algorithme de décision d'équivalence entre deux formules :
$\phi \equiv \psi$ ssi $(\phi \wedge \neg\psi)$ est non satisfiable

Quelques problèmes en cours

Intégration dans les process Industriels

- Avoir des algos et outils efficaces
- "Masquer" le recours aux techniques formelles :
  - Expression du comportement
    (AADL, SDL, ...) $\mapsto$ "RDP/Automates"
  - Expression des propriétés (?, PSL)
    Patterns, Propriétés "Métier", Observateurs, ...
  - Retour de vérification : Contre-exemple au niveau utilisateur

Passage à l'échelle : Comment contourner l'explosion combinatoire

- Compression : optimisation du stockage (ex Binary Decision Diagram)
- Réduction : Odres Partiels, Symétrie , Abstraction ...
  Pour une propriété ϕ donnée, construire $SK_{Red\phi} \subset SK$
  vérifiant $SK \models \phi$ *ssi* $SK_{Red\phi} \models \phi$

Autres

Systèmes infinis (temporisés, hybrides, ...) Systèmes Paramétrés
Log temporelles temporisées souvent indécidables

Outline

Sur les Tps

Sujets disponibles via
http ://homepages.laas.fr/francois/POLYS/ENSEIGNEMENT/MC2/

Boîte à outils TINA (Time Petri Net Analyser)

`selt` : Model-checker $SE - LTL$
`muse` : Model-checker μ-calcul (avec interface CTL pour les TPs)
http ://projects.laas.fr/tina/

Bibliothèques Patterns

http ://patterns.projects.cis.ksu.edu/

TWB

http ://twb.rsise.anu.edu.au

Quelques Outils

- LTL
  SPIN, Divine, TINA/Selt, ...
- CTL
  nusmv
- CTL*
  EMC, CWB
- μ-calcul
  MECV, CADP, TINA/Muse, ...
- Timed-TL
  (Kronos), UPPAAL, ROMEO, TINA, ...
- Probabilistic-TL
  Prism, ...
- ...

Bibliographie

A. Arnold *Systèmes de transitions finis et sémantique des processus communicants* – Masson Paris, 1992.

E. Audureau, L. Farinas del Cerro, P. Enjalbert *Logique Temporelle. Sémantique et Validation des programmes parallèles* Masson Paris, 1990.

E.M. Clarke, O. Grumberg, D.A. Peled *Model Checking* MIT Press, 2000

Ed. P. Schnoebelen *Systems and Software Verification. Model-Checking Techniques and Tools.* – Springer, 2001.

Ed. N. Navet *Systèmes temps réel 1 – techniques de description et de vérification*, Hermes 2006.

Ed. M.Diaz *Petri Nets : Fundamental Models, Verification And Applications* ISTE 2009

Autres :
Forum Méthodes Formelles : http ://projects.laas.fr/IFSE/FMF
Ecole temps-réel : etr2015.irisa.fr/
.../...

Table des Matières

Table des Matières