Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Transformer M en un automate de Büchi

- Soit M=(Q,T,A, q_0, AP, l) une structure de Kripke. On construit un automate de Büchi B= (Q', Σ, q'0, T', F) tel que L(B)=⟦M⟧:
- Idée: on fait «basculer» les étiquettes des états vers les transitions + tous les états sont acceptants
  - $\Sigma=2^{AP}$
  - $Q'=T \cup \{q_0'\}$
  - F=Q'
  - Soit $t=(q_0,q)\in T$, alors $(q_0',l(q_0),t)\in T'$
  - Soient t=(q,q') et $t'=(q',q'')\in T$, alors $(t,l(q'),t')\in T'$

Exemple

- au tableau

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$.

Tester le vide de l'intersection

- Construire l'automate $A_M \otimes A_{\neg\varphi}$ tel que $L(A_M \otimes A_{\neg\varphi}) = L(A_M) \cap L(A_{\neg\varphi})$. (cf théorème)
- Rechercher s'il existe un mot accepté par $A_M \otimes A_{\neg\varphi}$. (cf théorème)

Model-Checking LTL: catching bugs with a lasso

Model-Checking LTL : approche par automates

- Donnée: Structure de Kripke M, formule LTL φ.
- Etapes de l'algorithme :
  - Transformer M en un automate A_M tel que $L(A_M)=[\![M]\!]$ ✔ $O(|M|)$
  - Transformer φ en un automate $A_{\neg\varphi}$ tel que $L(A_{\neg\varphi})=[\![\neg\varphi]\!]$ ✔ $O(2^{|\varphi|})$
  - Tester si $L(A_M)\cap L(A_{\neg\varphi})=\varnothing$. ✔ $O(|M|.2^{|\varphi|})$

Model-Checking LTL: techniques à la volée

- Pas nécessaire de construire l'automate produit en entier
- On construit pas à pas, et on s'arrête lorsqu'on trouve un cycle (=contre-exemple).

3.3 Inclure des notions d'équité

Exécutions équitables

- Chaque processus est activé infiniment souvent : $\wedge_i$ (GF enabled$_i$)
- Aucun processus ne reste infiniment dans la section critique : $\wedge_i \neg$(FGcritic$_i$)= $\wedge_i$GF($\neg$critic$_i$)

Contraintes d'équité

- Contrainte d'équité inconditionnelle : $GF\varphi$
- Contrainte d'équité forte : $GF\varphi \rightarrow GF\varphi'$
- Contrainte d'équité faible : $FG\varphi \rightarrow GF\varphi'$

Conditions d'équité

- Une condition d'équité est une conjonction de contraintes d'équité
- Une condition d'équité est une formule LTL!

Exécutions équitables

- Soit t une trace d'exécution d'une structure de Kripke M, *fair* une condition d'équité
- t est équitable si t,0⊨ *fair*

LTL équitable

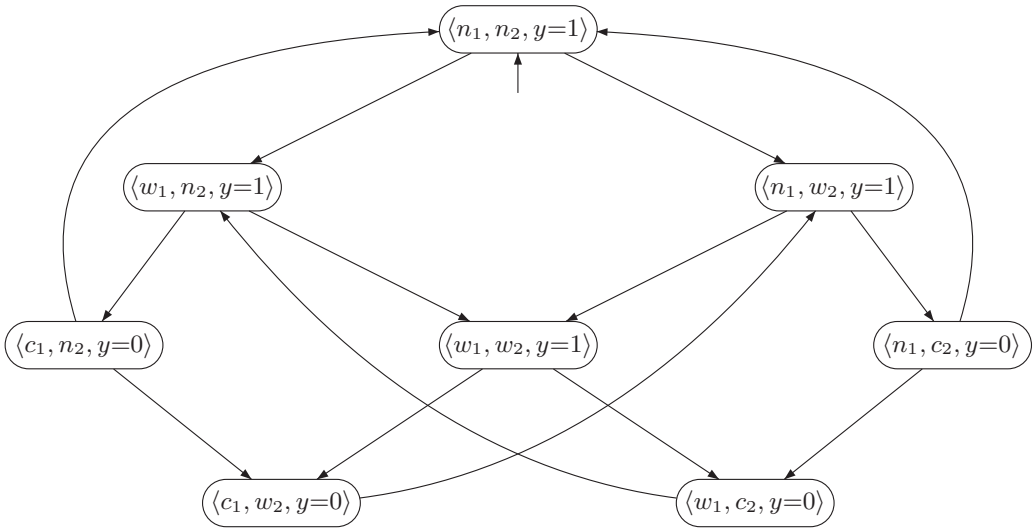
- Soit une structure de Kripke M, *fair* une condition d'équité et φ une formule LTL.
- $M \models_{fair} \varphi$ ssi $t,0 \models_{fair} \varphi$ pour toute trace initiale t de M ssi $t,0 \models \varphi$ pour toute trace initiale équitable de M.

Exemple

$GF(w_1 \wedge \neg c_2) \rightarrow GFc_1 \wedge GF(w_2 \wedge \neg c_1) \rightarrow GFc_2$

$\wedge$

$(FGn_1 \rightarrow GFw_1) \wedge (FGn_2 \rightarrow GFw_2)$



$M \models_{fair} GFc_1 \wedge GFc_2$

Model-Checking LTL équitable

Théorème : $M \models_{fair} \varphi$ ssi $M \models fair \rightarrow \varphi$.

CTL équitable

- Conditions d'équité ne peuvent pas s'écrire en CTL
- On voudrait dire $A(\mathit{fair} \rightarrow \varphi)$ ou $E(\mathit{fair} \wedge \varphi)$ mais ce sont des formules CTL*

CTL équitable

$\varphi ::= p \in AP \mid \neg\varphi \mid \varphi \vee \varphi \mid E_f X\varphi \mid A_f X\varphi \mid E_f \varphi U \varphi \mid A_f \varphi U \varphi$

$s \models p$ ssi $p \in l(s)$

$s \models \neg\varphi$ ssi $s \not\models \varphi$

$s \models \varphi_1 \vee \varphi_2$ ssi $s \models \varphi_1$ ou $s \models \varphi_2$

$s \models E_f X\varphi$ ssi il existe une exécution équitable $s_0 s_1 \ldots$ tel que $s_0 = s$, t.q. $s_1 \models \varphi$

$s \models A_f X\varphi$ ssi s', pour toute exécution équitable $s_0 s_1 \ldots$ telle que $s_0 = s$, $s_1 \models \varphi$

$s \models E_f \varphi_1 U \varphi_2$ ssi il existe une exécution équitable $s_0 s_1 \ldots s_k$ tel que $s_0 = s$, $s_k \models \varphi_2$ et pour tout $0 \leq i \leq k$, $s_i \models \varphi_1$.

$s \models A_f \varphi_1 U \varphi_2$ ssi pour toute exécution équitable $s_0 s_1 \ldots$ telle que $s_0 = s$, il existe k t.q. $s_k \models \varphi_2$ et pour tout $0 \leq i \leq k$, $s_i \models \varphi_1$.

Model-Checking de CTL équitable

- On suppose qu'on a étiqueté les états avec une nouvelle AP fair, qui indique s'il existe une exécution équitable partant de l'état
- $s \models E_f X\varphi$ ssi $s \models EX(\varphi \wedge fair)$
- $s \models A_f X\varphi$ ssi $s \models AX(\neg\ fair \vee \varphi)$
- $s \models E_f \varphi U \varphi'$ ssi $s \models E\varphi U(fair \wedge \varphi')$
- $s \models A_f \varphi U \varphi'$ ssi $s \models \neg E_f G \neg\varphi' \wedge \neg E_f(\neg\varphi' U(\neg\varphi \wedge \neg\varphi'))$
  ssi $s \models \neg E_f G \neg\varphi' \wedge \neg E(\neg\varphi' U(fair \wedge \neg\varphi \wedge \neg\varphi'))$

Model-Checking de CTL équitable

- 1er problème : Comment calculer fair?
- Rappel : s⊨fair ssi il existe une exécution équitable partant de s
- → Dépend de la condition d'équité!
- 2ème problème : calculer $E_f G\varphi$

Calculer fair : les composantes fortement connexes

Définition : Dans un graphe, une composante fortement connexe (SCC) est un sous-graphe maximal tel que pour toute paire de noeuds (s,s') s' est accessible depuis s, et s est accessible depuis s'

L'algorithme de Tarjan permet de calculer les SCC d'un graphe en temps linéaire.

Calculer fair : le cas inconditionnel

- On considère une condition d'équité de la forme GFψ, avec ψ formule CTL.
- On marque les états par ψ.
- On calcule les SCC de M par l'algorithme de Tarjan.
- Soit S' l'union des SCC qui intersectent S(ψ).
- fair est l'ensemble des états pouvant atteindre S'.
- (accessibilité se calcule en temps linéaire)

Calculer $E_f G\varphi$: le cas inconditionnel

- Effectuer mark(φ).
- Soit M(φ) la restriction de M aux états de S(φ).
- Calculer les SCC de M(φ) (algo de Tarjan).
- Soit S' l'union de SCC de M(φ) intersectant S(ψ), avec ψ la condition d'équité.
- M,s$\models E_f G\varphi$ ssi M,s$\models E\varphi U S'$ ssi M(φ),s$\models EFS'$.
- $\rightarrow$problème d'accessibilité.

Model-Checking de CTL équitable

- On suppose qu'on a étiqueté les états avec une nouvelle AP fair, qui indique s'il existe une exécution équitable partant de l'état ✔
- $s \models E_f X\varphi$ ssi $s \models EX(\varphi \wedge fair)$
- $s \models A_f X\varphi$ ssi $s \models AX(\neg\, fair \vee \varphi)$
- $s \models E_f \varphi U \varphi'$ ssi $s \models E\varphi U(fair \wedge \varphi')$
- $s \models A_f \varphi U \varphi'$ ssi $s \models \neg E_f G \neg\varphi' \wedge \neg E_f(\neg\varphi' U(\neg\varphi \wedge \neg\varphi'))$
  ssi $s \models \neg E_f G \neg\varphi' \wedge \neg E(\neg\varphi' U(fair \wedge \neg\varphi \wedge \neg\varphi'))$ ✔

Et aussi...

- Autres logiques temporelles : CTL*,mu-calcul... (plus expressives), ForSpec, PSL, Sugar... (industrie)
- Méthodes efficaces : méthodes symboliques, techniques de réduction (ordres partiels...)