Model Checking et LTL: Application et comparaison des automates de Büchi et de la méthode des tableaux

Contents

1 Introduction

Dans un monde dont la dépendance à divers technologies est très importante, il est important de s'assurer du bon fonctionnement de ces technologies. C'est ainsi que la vérification de modèle rentre en jeu pour répondre à ces problèmatiques. La vérification de problème requiert des algorithmes permettant justement de faire ces vérifications... C'est donc dans le cadre de la logique temporelle linéaire appliqué à la vérification de modèle, que nous allons examiner les deux méthodes décrites plus tard.

2 Préliminaires

2.1 Model Checking et LTL

Definition 1 (Formule de la LTL): On définit F l'ensemble des formules de la LTL inductivement par:

- $p \in AP \implies p \in F$
- si ψ et ϕ sont des formules de LTL alors $\neg\psi, \phi \vee \psi, X\psi, \phi U \psi$ sont des formules de LTL

AP un ensemble fini de variables propositionnelles.

Definition 2 (Opérateurs X et U): On les définit par:

- $X\phi$: ϕ doit être satisfaite dans l'état suivant (neXt)
- $\psi U \phi$: ψ doit être satisfaite dans tous les états jusqu'à un état où ϕ est satisfait (Until)

Definition 3 (Monde): On définit un tel objet comme $\omega := w_0, w_1, \ldots$ une suite infinie d'état. On écrira $\omega^i := w_i, w_{i+1}, \ldots$ un suffixe de ω.

Soit $v : \omega \times F \to \{T, F\}$ une fonction de valuation.

Definition 4 (Satisfaction d'un monde): En LTL, on définit $\omega \models f$ via:

- $\omega \models a \Leftrightarrow v(w_0, a) = T$ si a atomique
- $\omega \models \neg f$ si $\neg(\omega \models f)$
- $\omega \models f \vee g$ si $\omega \models f$ ou $\omega \models g$
- $\omega \models \mathbf{X}f$ si $\omega^1 \models f$
- $\omega \models f\mathbf{U}g$ si $\omega \models g$ ou $\omega \models f \wedge \mathbf{X}(f\mathbf{U}g)$

On peut trouver plusieurs applications à la LTL:

- Preuve de programme concurrentiel
- Raisonnement sur des circuits intégrés
- Raisonnement sur les protocoles de communications

Toutes ces applications s'inscrivent dans le cadre de la vérification de modèles qui est une méthode permettant de montrer la correction de systèmes informatiques complexes.

Definition 5 (Verification de modèle (Model Checking)): Technique de vérification qui explore tout les états possible d'un système de manière force brute.

- Ce n'est pas avec la LTL que nous allons faire des gros exemples de model checking (Un model checker de base peut s'occuper de 10^9 états environ, allant jusqu'à 10^{476} pour les meilleurs!).
- Ainsi un model checker typique pourrait utiliser entre autres la LTL pour s'occuper en particulier des deadlocks. C'est ainsi qu'on pourrait utiliser la méthode des tableaux pour prouver une formule qui montre que des algorithmes vont bien s'executer de manière à ne pas avoir d'inter-dépendance.
- Ainsi nous allons examiner un exemple dans lequel nous allons appliquer la méthode des tableaux pour prouver le bon fonctionnement du système choisit!

2.2 Méthode des tableaux

Definition 6 (Méthode des tableaux): Algorithme pour prouver qu'une assertion ϕ ayant pour hypothèse (H_n) soit satisfiable

On supposera que aucune hypothèse n'est faite, on peut facilement adapter l'étude que l'on va faire lors d'ajout d'hypothèse.

- On place ϕ et ses hypothèses dans la racine.
- On applique des règles (R_x) à chaque formule en bout d'arbre qui sont developpables
- Si on trouve des contradictions (des *cycles*) dans toutes les branches de l'arbre (branches fermées), l'arbre est fermé donc la formule est insatisfiable.

$$\begin{array}{ccc} & \phi \vee \psi & \\ \swarrow & & \searrow \\ \phi & & \psi \end{array} \qquad \begin{array}{ccc} & \neg(\phi \wedge \psi) & \\ \swarrow & & \searrow \\ \neg\phi & & \neg\psi \end{array} \qquad \begin{array}{c} \neg\neg\phi \\ | \\ \phi \end{array} \qquad \begin{array}{c} \phi \wedge \psi \\ | \\ \phi, \psi \end{array} \qquad \begin{array}{c} \neg(\phi \vee \psi) \\ \\ \neg\phi, \neg\psi \end{array}$$

Les règles définis précedemment sont dites Smullyan-Style

Definition 7 (Branche fermée): Une branche est fermée si elle contient ϕ et $\neg\phi$

Une formule est insatisfiable ssi son arbre associé est dit fermé ssi toutes les branches le sont.
On peut utiliser la méthode des tableaux pour montrer qu'une formule est une tautologie:

1. On place $\neg\phi$ et ses hypothèses dans la racine.
2. On applique des règles (R_x) à chaque formule en bout d'arbre qui sont developpables

3. Si on trouve a et $\neg a$ dans les branches de l'arbre (des *cycles*), alors ϕ est une tautologie

On pourra donc aussi adapter nos recherches pour la recherche de tautologie.

- On ajoute les quantificateurs $\exists$ et $\forall$
- Un ensemble de fonctions de symboles $\mathcal{F}$ qui a des symboles associe un symbole
- Un ensemble de relation $\mathcal{R}$ qui a des symboles associe un booléen.

On note l'arité le nombre d'argument d'une fonction et X les variables.

> Definition 8: On définit les termes $\mathcal{T}(\mathcal{F}, X)$ par induction:
>
> - Tout $x \in X$ est un terme.
> - Les constantes sont des termes (symbole d'arité 0).
> - $f(t_1, \ldots, t_n)$ est un terme si f est un symbole d'arité n et $t_1, \ldots, t_n$ sont des termes.

On ajoute quatres règles pour le 1er ordre (?):

$$\begin{array}{cccc} \forall x, P(x) & \exists x, P(x) & \neg(\forall x, P(x)) & \neg(\exists x, P(x)) \\ | & | & | & | \\ P(t) & P(c) & \exists x, \neg P(x) & \forall x, \neg P(x) \end{array}$$

où t et c est une variable fixe quelquonque.
Attention: Pour la règle $\forall$, on peut choisir t, pour la règle $\exists$, on prend une variable fraiche c!
On définit deux nouvelles règles pour la méthode des tableaux en LTL:

$$\begin{array}{c} \neg \mathbf{X} f \\ | \\ \mathbf{X} \neg f \end{array} \qquad \begin{array}{ccc} & f \mathbf{U} g & \\ / & & \backslash \\ g & & f,\ \mathbf{X}(f \mathbf{U} g) \end{array} \qquad \begin{array}{ccc} & \neg(f \mathbf{U} g) & \\ / & & \backslash \\ g & & \neg g,\ \neg f \vee \neg \mathbf{X}(f \mathbf{U} g) \end{array}$$

> Definition 9 (Formule élementaire): f est élementaire ssi il respecte un de ses 2 points:
>
> - C'est une formule atomique (ou la négation d'une formule atomique)
> - Il a comme connecteur logique "principale" $\mathbf{X}$ (des X-formules)

Si un noeud contient uniquement des formules elementaires, alors on créé un fils du noeud contenant toutes les **X**-formules sans leur connecteur logique principal **X**

2.3 Automate de Buchi

> Definition 10 (BA): Un automate de Büchi est un 5-uplets (S, I, T, F, Σ) tel que
>
> - S est un ensemble fini d'état
> - $I \subseteq S$ un ensemble d'état initiaux
> - $F \subseteq S$ un ensemble d'état finaux
> - Σ Un ensemble fini de symboles appelé alphabet
> - $T : \{S \times \Sigma\} \mapsto \mathcal{P}(S)$ Une fonction de transition.
>
> Cet automate particulier accepte des séquences infinis (donc des mots infinis) ssi le mot passe par un état acceptant une infinité de fois.

C'est un outil utilisé dans le cadre de la vérification de modèle en LTL.
On va utiliser l'algorithme de Gerth pour transformer notre formule LTL en Automate de Büchi Généralisé (GBA)

Definition 11 (GBA): Un GBA est un automate de Büchi avec la seule particularité que F est un ensemble d'ensemble d'état acceptant appelé **condition d'acceptation**.
Un GBA accepte un mot ssi il est passé au moins une fois par un état dans chaque ensemble d'état acceptant.

Nous nous ramenerons à un Automate de Büchi via un autre algorithme pour simplifier.

Un ensemble multiple d'état acceptant peut être traduit en un ensemble en construisant un automate par une "counting construction". C'est à dire si $A = (S, I, \{F_1, \ldots, F_n\}, \Sigma, T)$, alors il est equivalent à $A' = (S', I', F, \Sigma, T')$ telle que

- $Q' = Q \times \{1, ..., n\}$
- $I' = I \times \{1\}$
- $F' = F_1 \times \{1\}$
- $\Delta' = \{((q, i), a, (q', j))|(q, a, q') \in \Delta$ et si $q \in F_i$, $j = i + 1 \mod n$ sinon $j = i\}$

qui est bien un automate de Büchi.

2.4 Résumé

Pour résumé, nous devons:

1. Implémentation de la méthode des tableaux en LTL
2. Implémentation des automates de Büchi
   - Implémentation des automates de Büchi
   - Implémentation de l'algorithme de Gerth
   - Implémentation de la traduction entre GBA et BA
3. Comparaison dans un exemple du model checking en terme de rapidité et compléxité (théorique et pratique)

Ensuite, nous mettrons cela en application sur la vérification du bon fonctionnement d'un feu rouge. Nous donnerons une formule logique définissant son clignotement et nous vérifirons via la méthode des tableaux que la formule est correcte selon ce qu'on attend d'elle. Et nous ferons de même avec l'automate de Buchi.

3 Implémentation

3.1 Méthode des tableaux

Adapter une ancienne implémentation (en rajoutant les elements de LTL)

3.2 Automates de Büchi

3.2.1 Implémentation de base

...

3.2.2 Algorithme de Gerth

...

3.2.3 Traduction GBA > BA

...

4 Application

4.1 A la méthode des tableaux

...

4.2 Aux automates de Büchi

...

4.3 Comparaison pratique

...

4.4 Comparaison théorique

...

5 Conclusion et Bibliographie

5.1 Conclusion

...

5.2 Bibliographie

1. Logique: fondements et applications (Dunod) de Pierre Le Barbenchon, Sophie Pinchinat, François Schwarzentruber
2. Mathematical Logic: Tableaux Reasoning for Propositional Logic de Chiara Ghidini
3. The tableau method for temporal logic: an overview - Pierre WOLPER
4. Principles of Model Checking - Christel Baier et Joost-Pieter Katoen

(1) Notation Logique (2) Méthode des tableaux LP (3) Méthode des tableaux LTL (4) Model Checking
Noté Bibliographie pour Buchi