M2 ISTR - Vérification et Validation

CTL and LTL model checking algorithms

Julien Brunel, ONERA
Julien.Brunel@onera.fr

Model checking de CTL

$$M \overset{?}{\models} \varphi$$

Idée
Pour chaque sous formule ϕ de φ, marquer les états du modèle qui satisfont ϕ.

Entrées : un modèle $M = (S, I, \rightarrow, V)$ et une formule φ

Résultat : l'ensemble des états qui satisfont φ

- p : tous les états s tels que $p \in V(s)$ sont marqués
- $\neg\phi$: tous les états qui ne sont pas marqués pour φ
- $\phi_1 \wedge \phi_2$: les états qui sont marqués pour ϕ_1 et pour ϕ_2
- $\mathbf{E}\mathrm{X}\ \phi$: les états dont un successeur est marqué pour ϕ

Model checking de CTL

E$[\phi_1 \ \mathbf{U} \ \phi_2]$

- marquer les états qui sont marqués pour ϕ_2
- ajouter les états qui sont marqués pour ϕ_1 et dont un successeur est marqué
- arrêter quand l'ensemble des états marqués n'augmente plus

A$[\phi_1 \ \mathbf{U} \ \phi_2]$

- marquer les états qui sont marqués pour ϕ_2
- ajouter les états qui sont marqués pour ϕ_1 et dont tous les successeurs sont marqués
- arrêter quand l'ensemble des états marqués n'augmente plus

Fonction de marquage

Fonction $\mathrm{SAT}(\varphi)$

$$
\begin{array}{lcl}
\mathrm{SAT}(p) & = & \{s \in S \;/\; p \in V(s)\} \\
\mathrm{SAT}(\neg\phi) & = & S - \mathrm{SAT}(\phi) \\
\mathrm{SAT}(\phi_1 \wedge \phi_2) & = & \mathrm{SAT}(\phi_1) \cap \mathrm{SAT}(\phi_2) \\
\mathrm{SAT}(\mathbf{E}\mathrm{X}\,\phi) & = & \{s \in S \;/\; \exists s'\; s \rightarrow s' \wedge s' \in \mathrm{SAT}(\phi)\}
\end{array}
$$

Fonction de marquage

Fonction $\mathrm{SAT}(\varphi)$

$$
\begin{array}{lcl}
\mathrm{SAT}(p) & = & \{s \in S \ / \ p \in V(s)\} \\
\mathrm{SAT}(\neg\phi) & = & S - \mathrm{SAT}(\phi) \\
\mathrm{SAT}(\phi_1 \wedge \phi_2) & = & \mathrm{SAT}(\phi_1) \cap \mathrm{SAT}(\phi_2) \\
\mathrm{SAT}(\mathbf{E}\mathrm{X}\ \phi) & = & \{s \in S \ / \ \exists s' \ s \rightarrow s' \wedge s' \in \mathrm{SAT}(\phi)\} \\
 & & \\
\mathrm{SAT}(\mathbf{E}[\phi_1 \ \mathrm{U} \ \phi_2]) & = & \mathrm{SAT}_{\mathtt{EU}}(\phi_1, \phi_2) \\
\mathrm{SAT}(\mathbf{A}[\phi_1 \ \mathrm{U} \ \phi_2]) & = & \mathrm{SAT}_{\mathtt{AU}}(\phi_1, \phi_2)
\end{array}
$$

$\mathrm{SAT}_{\mathtt{EU}}(\phi_1, \phi_2), \mathrm{SAT}_{\mathtt{AU}}(\phi_1, \phi_2)$

On se base sur les équivalences:

$$\mathbf{A}[\phi_1 \ \mathrm{U} \ \phi_2] \leftrightarrow \phi_2 \vee (\phi_1 \wedge \mathbf{A}\mathrm{X} \ \mathbf{A}[\phi_1 \ \mathrm{U} \ \phi_2])$$

$$\mathbf{E}[\phi_1 \ \mathrm{U} \ \phi_2] \leftrightarrow \phi_2 \vee (\phi_1 \wedge \mathbf{E}\mathrm{X} \ \mathbf{E}[\phi_1 \ \mathrm{U} \ \phi_2])$$

$\mathrm{SAT}_{\mathtt{EU}}(\phi_1,\phi_2), \mathrm{SAT}_{\mathtt{AU}}(\phi_1,\phi_2)$

$\mathrm{SAT}_{\mathtt{EU}}(\phi_1,\phi_2)$
$\mathtt{res} := \mathrm{SAT}(\phi_2)$;

$\mathtt{X} := S$;

while $\mathtt{X} \neq \textit{res}$ **do** $\{$

  $\mathtt{X} := \mathtt{res}$;

  $\mathtt{res} := \mathtt{res} \cup \quad (\mathrm{SAT}(\phi_1) \cap \{s \in S \ / \ \exists s' \ s \to s' \text{ and } s' \in \textit{res}\})$

$\}$

return $\mathtt{res}$;

$\mathrm{SAT}_{\mathtt{AU}}(\phi_1,\phi_2)$
$\mathtt{res} := \mathrm{SAT}(\phi_2)$;

$\mathtt{X} := S$;

while $\mathtt{X} \neq \textit{res}$ **do** $\{$

  $\mathtt{X} := \mathtt{res}$;

  $\mathtt{res} := \mathtt{res} \cup \quad (\mathrm{SAT}(\phi_1) \cap \{s \in S \ / \ \forall s' \text{ si } s \to s' \text{ alors } s' \in \textit{res}\})$

$\}$

return $\mathtt{res}$;

Exercice

Sur les modèles donnés en exemple, calculer les états satisfaisant
$\mathbf{A}[p \mathbin{\mathrm{U}} (p \wedge q)]$,
$\mathbf{E}[\neg q \mathbin{\mathrm{U}} p]$

Model checking de LTL

Étant donné une formule φ de LTL et un modèle $M = (S, I, \to, V)$, on veut déterminer si $M \models \varphi$

- Construction d'un automate (de Büchi) $A_{\neg\varphi}$ tel que A reconnaît exactement les traces qui satisfont $\neg\varphi$.
- Faire le produit $M \times A_{\neg\varphi}$ qui reconnaît les traces de M qui satisfont $\neg\varphi$
- Si $\mathcal{L}(M \times A_{\neg\varphi}) = \emptyset$ alors $M \models \varphi$, sinon $M \not\models \varphi$

Automates de Büchi

Definition
Un automate de Büchi est un tuple $A = (S, I, \rightarrow, F)$

- $S, I, \rightarrow$ définis comme d'habitude
- $F \subseteq S$ est un ensemble d'états d'acceptation

Une séquence infinie $\sigma = (s_0, s_1, \ldots)$ est acceptée par A si elle passe infiniment souvent par au moins un des états de F.

Automates de Büchi généralisés
Les automates de Büchi généralisé sont définis par une ensemble $F_1, F_2, \ldots, F_n$ d'ensembles d'états d'acceptation.

Une séquence infinie est acceptée si elle passe infiniment souvent par au moins un état de chaque ensemble F_i.

Construction d'un automate de Büchi

Idée :

- ramener les not ($\neg$) à l'intérieur des formules:
  $\neg X\phi \equiv X\neg\phi$, $\neg(\phi_1 \wedge \phi_2) \equiv \neg\phi_1 \vee \neg\phi_2$, $\neg(\phi_1 U \phi_2) \equiv \neg\phi_1 V \neg\phi_2$
- se servir des équivalences $\phi_1 U \phi_2 \equiv \phi_2 \vee (\phi_1 \wedge X(\phi_1 U \phi_2))$ et $\phi_1 V \phi_2 \equiv \phi_2 \wedge (\phi_1 \vee X(\phi_1 V \phi_2))$