

IV113 Validace a verifikace

Ověřování modelu s využitím řešičů SAT a SMT
(Bounded Model Checking)

Jiří Barnat

Problém splnitelnosti – SAT

- Nalezení valuace boolovských proměnných formule výrokové logiky takové, že formule je v této valuaci pravdivá.

Satisfiability Modulo Theory – SMT

- Problém rozhodnout splnitelnost formule prvořákové logiky s rovnostmi, predikáty a funkčními symboly kódující jednu či více zvolených teorií.

Typické teorie SMT

- Aritmetika neomezených celých a desetinných čísel.
- Aritmetika celých čísel omezené velikosti (bitové vektory).
- Teorie datových struktur (seznamy, pole, ...).

ZZZ aka **Z3**

- Nástroj vyvíjený v Microsoft Research.
- WWW interface — <http://www.rise4fun.com/Z3>
- Binární API pro použití v jiných aplikacích.

SMT-LIB

- Standardizace jazyka pro zadávání SMT dotazů.
- Volně dostupná knihovna s implementací SMT.

Pozorování

- Formule je platná právě když její negace není splnitelná.

Důsledek

- Řešiče SAT a SMT lze využít jako nástroje pro dokazování platnosti formulovaných tvrzení.

Syntéza modelu

- Řešiče SAT nejen rozhodují splnitelnost formulí, ale v případě splnitelnosti vrací požadovanou valuaci proměnných, pro niž je formule pravdivá.
- Na rozdíl od dokazovacích nástrojů tak poskytují "protipříklad" v případě neplatnosti dokazovaného tvrzení.

Ověřování safety vlastností redukcí na problém SAT

Hypotéza

- Je-li v systému chyba, pro její reprodukci stačí malý počet kontrolovaných kroků systému.

Myšlenka metody

- Používáme-li metodu ověřování modelu pro detekci chyb, je smysluplné zkoumat, zda k porušení specifikace dojde během prvních k kroků systému.

Literatura

- Armin Biere, Alessandro Cimatti, Edmund M. Clarke, Yunshan Zhu: Symbolic Model Checking without BDDs. TACAS 1999: 193-207, LNCS 1579.
- Henry A. Kautz, Bart Selman: Planning as Satisfiability. Proceedings of the 10th European conference on Artificial intelligence (ECAI'92): 359-363, 1992, Kluwer.

Předpoklady

- Množinu prefixů délky k všech běhů Kripkeho struktury M lze kódovat boolovskou formulí $[M]^k$.
- Porušení vlastnosti typu *safety*, které se projeví po provedení k kroků systému, lze kódovat formulí $[\neg\varphi]^k$.

Redukce na problém SAT

- Ověřuje se splnitelnost formule $[M]^k \wedge [\neg\varphi]^k$.
- Splnitelnost indikuje existenci protipříkladu délky k .
- Nesplnitelnost formule prokazuje neexistenci protipříkladu délky k .

Předpoklady

- Mějme Kripkeho strukturu $M = (S, T, I)$ s iniciálním stavem $s_0 \in S$.
- Libovolný stav $s \in S$ lze reprezentovat jako bitový vektor délky n , tj. stav $s = \langle a_0, a_1, \dots, a_{n-1} \rangle$.

Kódování M skrze boolovské formule

- $Init(s)$ – formule, která je splnitelná právě pro takovou valuaci proměnných a_1, a_2, \dots, a_n , které popisují stav s_0 .
- $Trans(s, s')$ – formule, která je splnitelná pro stavové vektory s, s' , právě tehdy když valuace proměnných $a_1, a_2, \dots, a_n, a'_1, a'_2, \dots, a'_n$ popisuje stavy, mezi kterými existuje přechod, tj. $(s, s') \in T$.

Popis běhů systému délky k

- Běh délky k je tvořen $k + 1$ stavy s_0, s_1, \dots, s_k .
- Množina všech běhů délky k struktury M je označena jako $[M]^k$ a je popsána následující formulí:

$$[M]^k \equiv \text{Init}(s_0) \wedge \bigwedge_{i=1}^k \text{Trans}(s_{i-1}, s_i)$$

Příklad $[M]^3 \wedge [\neg\varphi]^3$

- $\text{Init}(s_0) \wedge \text{Trans}(s_0, s_1) \wedge \text{Trans}(s_1, s_2) \wedge \text{Trans}(s_2, s_3) \wedge \neg\varphi(s_3)$

Úplnost metody BMC

Problém – nedetekované porušení vlastnosti typu safety

- Porušení invariantu je cestou délky k nedosažitelné.
- Cesty kratší než k nejsou v $[M]^k$ kódovány.

Ohraničení k shora

- Pokud $k \geq d$, kde d je průměr grafu, všechna místa možného porušení invariantu jsou pokryta.
- Průměr grafu lze omezit konstantou 2^n , kde n je počet bitů stavového vektoru.

Řešení problému

- Realizace procedury BMC postupně pro $k \in [0, d]$.

Fakta

- Určení konstanty d uživatelem je nereálné.
- Bezpečné horní odhady jsou velmi vzdálené realitě.
- Chtěli bychom, aby samotná procedura verifikace detekovala, zda má smysl nadále zvyšovat k .

Kostra algoritmu pro úplný BMC

$k = 0$

while (true) **do**

if (existuje protipříklad délky k)

then return "Invalid"

if (neexistuje protipříklad délky větší než k)

then return "Valid"

$k = k + 1$

od

Předpoklady

- Kripkeho struktura $M = (S, T, I)$.
- Stavy popsány bitovými vektory fixní délky.
- $Trans$ je SAT reprezentace binární relace T .

Cesta délky n

$$path(s_{[0..n]}) \equiv \bigwedge_{0 \leq i < n} Trans(s_i, s_{i+1})$$

Platnost tvrzení Q podél celé cesty

$$all.Q(s_{[0..n]})$$

Cesta bez cyklů

$$\text{loopFree}(s_{[0..n]}) \equiv \text{path}(s_{[0..n]}) \wedge \bigwedge_{0 \leq i < j \leq n} s_i \neq s_j$$

Existence cesty délky n z s_0 do s_n

$$\text{path}_n(s_0, s_n) \equiv \exists s_1 \dots s_{n-1}. \text{path}(s_{[0..n]})$$

Nejkratší cesta

$$\text{shortest}(s_{[0..n]}) \equiv \text{path}(s_{[0..n]}) \wedge \neg \left(\bigvee_{0 \leq i < n} \text{path}_i(s_0, s_n) \right)$$

Verifikace

- Chceme ukázat, že není dosažitelný stav z iniciální konfigurace, který by porušoval specifikaci φ , tedy chceme ukázat, že

$$\forall i. \forall s_0 \dots s_i. (Init(s_0) \wedge path(s_{[0..i]}) \implies \varphi(s_i))$$

Alternativně

- Chceme ukázat, že z chybového stavu není směrem zpět dosažitelný iniciální stav

$$\forall i. \forall s_0 \dots s_i. (\neg Init(s_0) \iff path(s_{[0..i]}) \wedge \neg \varphi(s_i))$$

Ekvivalentně

$$\forall i. \forall s_0 \dots s_i. \neg (Init(s_0) \wedge path(s_{[0..i]}) \wedge \neg \varphi(s_i))$$

Podmínka terminace v kostře algoritmu pro BMC

- Není delší acyklická cesta z počátečního stavu. tj. následující formule je nespelnitelná:

$$Init(s_0) \wedge loopFree(s_{[0..i+1]})$$

- **Platí i symetricky pro zpětnou dosažitelnost z chybových stavů.**

Řešení 1

- $\text{not SAT} \left(loopFree(s_{[0..i+1]}) \wedge Init(s_0) \right)$
 \vee
• $\text{not SAT} \left(loopFree(s_{[0..i+1]}) \wedge \neg\varphi(s_{i+1}) \right)$

Vyšší účinnost terminačního kritéria

- Při zpětné dosažitelnosti z $\neg\varphi$ stavů není třeba uvažovat cesty, které jdou přes další $\neg\varphi$ stavy.
- Symetricky platí i pro dopřednou variantu pro systémy, ve kterých je definováno více iniciálních stavů, tj. pro detekci úplnosti není třeba uvažovat cesty, které procházejí dalšími iniciálními stavy.

Řešení 2

- $$\text{not SAT} \left(\text{loopFree}(s_{[0..i+1]}) \wedge \text{Init}(s_0) \wedge \text{all.} \neg \text{Init}(s_{[1..i+1]}) \right)$$
$$\vee$$
$$\text{not SAT} \left(\text{loopFree}(s_{[0..i+1]}) \wedge \neg\varphi(s_{i+1}) \wedge \text{all.}\varphi(s_{[0..i]}) \right)$$

Pozorování

- Pro malé hodnoty k , dotazy na řešiče SAT nevedou k nalezení protipříkladu ani k ukončení výpočtu.
- Chceme proceduru BMC začít s k větším než 0.

Reformulace testu na protipříklad

- Původní test na existenci protipříkladu pro dané k

$$\text{SAT}\left(\text{Init}(s_0) \wedge \text{path}(s_{[0..k]}) \wedge \neg\varphi(s_k)\right)$$

je nutno reformulovat, abychom neminuli protipříklady, jež jsou kratší než výchozí hodnota k .

- Nový test na přítomnost protipříkladu:

$$\text{SAT}\left(\text{Init}(s_0) \wedge \text{path}(s_{[0..k]}) \wedge \neg \text{all}.\varphi(s_{[0..k]})\right)$$

Pozorování

- Testy lze reformulovat tak, aby připomínaly strukturu matematické indukce.
- TAUT je test na tautologii (nesplnitelnost negace).

Báze

- Test na přítomnost protipříkladu.

$$\text{SAT} \left(\neg \left(\text{Init}(s_0) \wedge \text{path}(s_{[0..i]}) \implies \text{all}.\varphi(s_{[0..i]}) \right) \right)$$

Indukční krok

- Test na úplnost.

$$\begin{aligned} & \text{TAUT} \left(\neg \text{Init}(s_0) \iff \text{all}.\neg \text{Init}(s_{[1..(i+1)]}) \wedge \text{loopFree}(s_{[0..i+1]}) \right) \\ & \vee \\ & \text{TAUT} \left(\text{loopFree}(s_{[0..i+1]}) \wedge \text{all}.\varphi(s_{[0..i]}) \implies \varphi(s_{i+1}) \right) \end{aligned}$$

Pozorování

- Průměr grafu (d) je délka nejdelší z nejkratších cest mezi každými dvěma vrcholy grafu.
- Acyklická cesta v grafu může být výrazně delší než je průměr grafu.

BMC s nejkratšími cestami

- Algoritmus BMC je korektní, pokud se místo *loopFree* použije *shortest*.
- Predikát *shortest* ale vyžaduje použití kvantifikátorů, nejedná se tedy o čistou aplikaci SAT.

Pro více detailů viz ...

- Mary Sheeran, Satnam Singh, and Gunnar Stålmarck: Checking Safety Properties Using Induction and a SAT-Solver, FMCAD 2000, 108-125, LNCS 1954, Springer.

Ověřování modelu LTL metodou BMC

Pozorování 1

- LTL je dobře definováno pouze pro nekonečné běhy.
- Pro vyhodnocování LTL na konečných cestách použijeme tříhodnotovou logiku (platí, neplatí, nelze říci).
- Platnost některých LTL formulí nelze rozhodnout na žádné konečné cestě (např. $GF a$).

Pozorování 2

- Cykly tvořené malým počtem stavů jsou procedurou BMC vždy rozbaleny do acyklické cesty délky k .
- Umožníme kódovat cesty, jež mají tvar lasa.
- Tzv. (k, l) -cyklické cesty.

(k,l) -cyklické běhy

- Běh $\pi = s_0 s_1 s_2 \dots$ Kripkeho struktury $M = (S, T, l, s_0)$ je (k, l) -cyklický pokud

$$\pi = (s_0 s_1 s_2 \dots s_{l-1})(s_l \dots s_k)^\omega,$$

kde $0 < l \leq k$ a $s_{l-1} = s_k$.

Pozorování

- Pokud π je (k, l) -cyklický, pak π je též $(k + 1, l + 1)$ -cyklický.
- Nahlížení konečné cesty délky jako (k, k) -cyklické je nekorektní (může vytvořit neexistující běh M).
- Každá cesta délky k je acyklická nebo je (k, l) -cyklická.

Sémantika LTL pro konečné prefixy

- Mějme π běh Kripkeho struktury M .
- Nechť je dáno k .
- $\pi = \pi^0$

$$\begin{aligned}\pi^i \models_{nl} X\varphi & \text{ iff } i < k \wedge \pi^{i+1} \models_{nl} \varphi \\ \pi^i \models_{nl} \varphi U \psi & \text{ iff } \exists j. i \leq j \leq k, \pi^j \models_{nl} \psi \text{ and} \\ & \forall m. i \leq m < j, \pi^m \models_{nl} \varphi\end{aligned}$$

Sémantika \models_k pro LTL při BMC

- Pro (k, l) -cyklické cesty platí, že $\pi \models_k \varphi \iff \pi \models \varphi$.
- Pro necyklické cesty platí, že $\pi \models_k \varphi \iff \pi^0 \models_{nl} \varphi$.
- $\models_k \implies \models_{k+1}$, \models_k aproximuje \models

Cíl

- Konstruujeme boolovskou formuli $[M, \varphi, k]$, která je splnitelná právě když Kripkeho struktura M má běh π takový, že $\pi \models_k \varphi$.
- $[M, \varphi, k] \equiv [M]^k \wedge [\varphi, k]$

Kódování

- $[M]^k$ kóduje všechny cesty délky k
- $[\varphi, k] \equiv \neg[\varphi, k]_0 \vee \bigvee_{l=1}^k l[\varphi, k]_0$
- $\neg[\varphi, k]_0$ kóduje, že cesta je acyklická a $\models_{nl} \varphi$
- $l[\varphi, k]_0$ kóduje, že cesta je (k, l) -cyklická a $\models \varphi$

Fragment LTL-X

- Redukce počtu přechodů (redukce velikosti zadání SAT).
- Podobné principy jako redukce částečným uspořádáním.

Pro zájemce

- Keijo Heljanko: Bounded Model Checking for Finite-State Systems
<http://users.ics.aalto.fi/kepa/qmc/slides-heljanko-2.pdf>
- Keijo Heljanko and Tommi Junttila: Advanced Tutorial on Bounded Model Checking
<http://users.ics.aalto.fi/kepa/acsd06-atpn06-bmc-tutorial/lecture1.pdf>

Shrnutí pro BMC

Obecné

- Redukce na standardní problém SAT, vývoj v oblasti řešičů SAT se projevuje i na BMC přístupu.
- Často vrací protipříklady minimální délky (ne vždy).
- Boolovské formule mohou být kompaktnější než OBDD reprezentace.

Verifikace HW

- Díky k-indukci velmi úspěšná metoda.

Verifikace SW

- Dle Software Verification Competition (SV-COMP) je aktuálně BMC (rozšířené o využití SMT) mezi nejlepšími metodami pro verifikaci (spíše falsifikaci) software.

Obecné

- Obecně neúplná metoda.
- Velké instance SAT jsou stále neřešitelné.

Verifikace SW

- Problematická analýza dynamických datových struktur.
- Problematická analýza cyklů.
- Neefektivní pro úplnou aritmetiku (částečně zvládá SMT).

Nástroje

- CBMC – BMC pro ANSI-C.
- ESBMC – využívá SMT, nadstavba CBMC.
- LLBMC – BMC nad LLVM bitkódem.

K zamyšlení ...

- Čím se liší moderní SMT-BMC od symbolické exekuce?
- Konceptně velmi podobné, rozlišení je v omezeném rozbalení cyklů a v jiném pořadí prohledávání (prořezaného) stromu symbolické exekuce.