NuSMV/NuXMV

Symbolické model checkery

Jan Mrázek

20. 4. 2015

Obsah

Vlastnosti NuSMV

- Vznikl jako rozšíření SMV[1] (Symbolic Model Verifier)
- Implementuje symbolický model checking pomocí BDD pro konečně stavové modely
- Podporuje i bounded model checking pomocí externího SAT solveru
- Kontrola CTL i LTL vlastností
- Podpora pro kontrolu invariantů běhů
- Koncipován především jako backend pro další nástroje
- Poslední release starý 3 roky, aktivní vývoj se přesunul na NuXMV

[1] `http://www.cs.cmu.edu/~modelcheck/smv.html`

Ukázka vstupního jazyka NuSMV

short.smv

```
MODULE main
VAR
  request : {Tr, Fa};
  state : {ready, busy};
ASSIGN
  init(state) := ready;
  next(state) := case
                   state = ready & (request = Tr): busy;
                   TRUE : {ready,busy};
                 esac;
SPEC
  AG((request = Tr) -> AF state = busy)
```

Vstupní jazyk ve stručnosti

- Model se skládá z jednotlivých modulů uvozených klíčovým slovem `MODULE` následovaných názvem modulu.
- Celý model musí obsahovat právě jeden modul s názvem `main`, jehož iniciální stav se bere jako iniciální stav celého modelu.
- Modul může přebírat i formální parametry (uvádí se v závorce za jeho názvem). S parametrem se pracuje jako s proměnnou.
- Modul může obsahovat následující sekce:
  - `VAR` pro deklaraci proměnných modulu
  - `ASSIGN` pro deklaraci ohodnocení proměnných (a tím přechodové funkce)
  - `TRANS` pro deklaraci přechodové funkce
  - `SPEC` pro definici CTL vlastností modulu
  - `DEFINE` pro definici virtuálních proměných
- modul může obsahovat i LTL vlastnosti uvozené slovem `LTLSPEC`, případně invarianty uvozené slovem `INVARSPEC`.

Deklarace proměnných

Proměnná může být některý z následujících typů:

- `Boolean` s hodnotami `TRUE` a `FALSE`
- `Integer` celočíselný datový typ, implementací omezen na rozsah od -2^{31} do $2^{31} - 1$.
- `Enum` s výčtem prvků. Lze vytvořit i enum z číselné řady.
- `Word` bitový vektor o dané šířce v doplňkovém kódu. Konstanty je třeba zapisovat ve tvaru `0(s/u)d(sirka)_(hodnota)`, např. `0sd32_57000`

```
VAR
  alive : boolean;
  rank : integer;
  state : {ready, busy, waiting};
  evaulation : 1..42;
  y : signed word[32];
```

Sekce ASSIGN – ohodnocení proměnných

- Pomocí `init(název_proměnné)` lze definovat počáteční hodnotu.
- Pomocí `next(název_proměnné)` lze určit nové ohodnocení proměnné. Kombinuje se s příkazem case.
- case vyhodnotí stráže a ze stráží vyhodnocených na true nederterministicky vybere.
- Proměnné bez `init` nabývají nederterministicky všechny své přípustné hodnoty
- Proměnné lze přiřadit kromě jedné hodnoty i množinu hodnot – nedeterministicky nabude jedné z hodnot v množině

```
ASSIGN
  init(state) := ready;
  next(state) := case
                   state = ready & (request = Tr): busy;
                   TRUE : {ready,busy};
                 esac;
```

Definice vlastností a invariantů

- CTL specifikace patří do sekce `SPEC`
- LTL specifikace se uvozují slovem `LTLSPEC`
- Invarianty se uvozují slovem `INVARSPEC`. Musí se jednat o jednoduché formule, které mohou využívat pouze operátor next.
- Pozor na mezery za operátory! Jsou vyžadovány
- `->` značí implikaci, `<->` značí ekvivalenci

```
SPEC
  AG((request = Tr) -> AF state = busy)
LTLSPEC G (proc1.state = entering -> F proc1.state = critical)
LTLSPEC G ! (proc1.state = critical & proc2.state = critical)
INVARSPEC y in (0..12)
```

Procesy

- Více modulů může být spuštěno naráz a můžou tak tvořit paralelní přechodový systém.
- Tohoto chování lze docílit deklarací proměnné typu `process`.
- Běžně se berou v potaz všechny běhy – včetně těch, kdy žádný či některý process neběží.
- Přidáním `FAIRNESS running` do modulu zajistíme, že se vezmou v potaz pouze ty běhy, kde process běží nekonečněkrát často.
- Procesy byly označeny za depreciated. NuXMV podporuje pouze synchronní modely.

```
MODULE main
VAR
  shared : boolean;
  proc0 : process observer;
  proc1 : process user(semaphore);
  proc2 : process user(semaphore);
ASSIGN
```

Začínáme s NuSMV prakticky

- Domovská stránka projektu `http://nusmv.fbk.eu/`. Zde je možné po vyplnění několika údajů stáhnout předkompilované binárky pro Windows a 64-bit Linux. Také je možné stáhnout zdrojové kódy a NuSMV si zkompilovat na svém stroji (přeji hodně štěstí a pevné nervy).
- Spouští se příkazem `nusmv název_modelu.smv`. V tomto režimu zkontroluje model vůči specifikacím uvedným v definici modelu
- Spuštěním `nusmv -int název_modelu.smv` se spustí v interaktivním módu, kdy je možné prozkoumávat stavový prostor a dotazovat se na vlastnosti modulu.
- K dispozici tutoriál `http://nusmv.fbk.eu/NuSMV/tutorial/v25/tutorial.pdf`
- A také uživatelská příručka `http://nusmv.fbk.eu/NuSMV/userman/v25/nusmv.pdf`

Několik užitečných příkazů v interaktivním módu

- `help příkaz` zobrazí nápovědu k příkazu, pokud je správně nainstalována
- `go` zinicializuje celý systém pro verifikaci
- `pick_state` označí specifikovaný stav jako aktivní, volbou `-r` vybere náhodný stav, běžně se specifikuje jméno či jiná omezení (např. ohodnocení proměnné)
- `simulate` provede simulaci jednoho konkrétní běhu. Je možné specifikovat jeho délku a strategii výběru následníka
- `show_traces` zobrazí stavy na daném běhu (specifikuje se číslo běhu)
- `check_ctlspec` zkontroluje zadanou CTL vlastnost modelu a případně uvede protipříklad
- `check_ltlspec` obdobně jako předchozí příkaz, jen pro LTL vlastnosti
- `check_invar` zkontroluje platonost invariantu v modelu
- NuSMV se ukončuje příkazem `quit`

Jak na BMC v NuSMV

- Při spouštění
  - Spuštěním s přepínačem `-bmc` začne NuSMV fungovat v módu BMC
  - Přepínačem `--bmc_length délka` lze nastavit počet iterací BMC
- V interaktivním módu
  - Příkazu `go_bmc` přebně do BMC
  - Počet iterací lze nastavit pomocí příkazu `bmc_length`. Defaultní je 10

Rozcvička

- Soubor short.smv
- Úkol:
  - Načtěte soubor do NuSMV
  - Prozkoumejte ručně část stavového prostoru
  - Označte iniciální stav
  - Zkontrolujte vlastnost `AG((request = Tr) -> AF state = busy)`

Binární počítadlo

- Soubor binary_counter.smv
- Úkol:
  - Prohlédněte si zdrojový soubor
  - Načtěte soubor do NuSMV
  - Ověřte, zda počítadlo správné přetéká (`111 -> 000`) a chybu případně opravte.

Semafor

- Soubor semaphore.smv
- Úkol:
  - Načtěte soubor do NuSMV
  - Ověřte, že nemůže dojít k deadlocku a že se do kritické sekce naráz nemůže dostat naráz více než jeden proces.
  - Případné chyby zkuste opravit.

Numerický model

- Soubor numeric.smv
- Úkol:
  - Načtěte soubor do NuSMV
  - Ověřte, že nemůže platit $y = 5 \implies X(y = 7)$
  - Zkuste ověřit invariant $y \in < 0; 12 >$. K jakému jste dospěli výsledku?
  - Zkuste ověřit invariant $y \in < 0; 7 >$. Jaký je výsledek tentokrát?
- Dodatečný úkol: Vyzkoušejte i v BMC

Na co je NuSMV krátký

- Jaká omezení NuSMV jste pozorovali?
- Jaké typy modelů mu dělají problémy?

Představení NuXMV

- Symbolický verifikační nástroj pro synchronní konečně a nekonečně stavové systémy
- Nástupce NuSMV, postaven na nových algoritmech
- Konečně stavové systémy jsou verifikovány pomocí algoritmů založených na SAT
- Nekonečně stavové systémy jsou verifikovány pomocí algoritmů založených na SMT
- Bodporuje i bounded model checking a ověřování invariantů (pokročilejšími metodami než NuSMV)
- V modelech podporuje racionální čísla
- Stejný modelovací jazyk jako NuSMV (pouze drobná rozšíření)

Začínáme s NuXMV prakticky

- Domovská stránka projektu `https://nuxmv.fbk.eu/`. Opět po vyplnění několika málo údajů je možné stáhnout předkompilované binárky.
- Ovládání v podstatě totožné s NuSMV.
- Uživatelská příručka k nalezení na `https://es.fbk.eu/tools/nuxmv/downloads/nuxmv-user-manual.pdf`

Příklady

- Zkuste analyzovat modely, se kterými si NuSMV neporadilo.
- Pozorujete nějaké zlepšení?