

Teorie systémů [MI-TES]

Souhrn látky

leden 2014

Obsah

1	Základní pojmy	2
2	Základy matematické logiky	2
2.1	Ekvivalence a De Morganovy zákony	2
2.2	Volné a vázané proměnné	2
3	Automatové modely	3
3.1	Synchronizovaná paralelní kompozice systémů	3
3.2	Kaskádní kompozice systémů $S_1 \rightsquigarrow S_2$	4
4	Linear temporal logic (LTL)	4
4.1	Vlastnosti a cesty	4
4.2	Převod do základní logiky	5
5	Testování a ověřování modelů	5
5.1	Bounded Model Checking	5
5.2	Satisfiability (SAT)	6
5.2.1	Základní pravidla při vyšetřování formule	6
5.3	Davis-Putnam-Logemann-Loveland algoritmus (DPLL)	6
5.4	Neomezené testování modelů	6
5.4.1	Induktivní podmínky	6
6	Petriho sítě	7
6.1	Základní vlastnosti	7
6.2	Převod do automatu	7
7	Časované automaty	8
8	Probabilistické modely	8

1 Základní pojmy

Interpretace Definice nějaké vlastnosti, značí se $\mathcal{I}()$.

Arita Počet argumentů nebo operandů matematické funkce nebo operace.

Potenční množina Množina všech podmnožin, značí se $\mathcal{P}(x)$ a počet jejích prvků je $2^{|M|}$.

Uppaal Software.

2 Základy matematické logiky

2.1 Ekvivalence a De Morganovy zákony

- $A \wedge [B \vee C]$ je ekvivalentní s $[A \wedge B] \vee [A \wedge C]$
- $A \vee [B \wedge C]$ je ekvivalentní s $[A \vee B] \wedge [A \vee C]$
- $\neg[A \wedge B]$ je ekvivalentní s $\neg A \vee \neg B$
- $\neg[A \vee B]$ je ekvivalentní s $\neg A \wedge \neg B$
- $A \Rightarrow B$ je ekvivalentní s $\neg A \vee B$
- $A \Leftrightarrow B$ je ekvivalentní s $(A \Rightarrow B) \wedge (B \Rightarrow A)$

2.2 Volné a vázané proměnné

Definice

Výskyt standardní proměnné x ve formuli nazýváme vázaným, pokud při cestě od tohoto listu ke kořeni syntaktického stromu narazíme na vrchol označovaný buď $\forall x$ nebo $\exists x$. V opačném případě nazveme tento výskyt volným. Kvantifikátor $\forall x$ nebo $\exists x$ váže všechny výskyty proměnné x , které jsou v syntaktickém stromu pod tímto kvantifikátorem.⁷

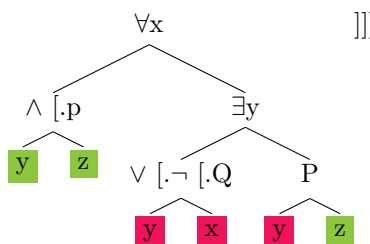
– J. Velebil

Příklad

Najděte volné a vázané proměnné ve výrazu

$$\forall x. [P(y, z) \wedge \exists y. [\neg Q(y, x) \vee P(y, z)]]$$

Sestavíme syntaktický strom:



3 Automatové modely

Formální definice

$$A = (S, S_0, I, O, R)$$

- S – množina stavů
- S_0 – počáteční stav
- I – množina vstupů
- O – množina výstupů
- R – přechodová funkce

Základní vlastnosti, které po automatovém modelu požadujeme.

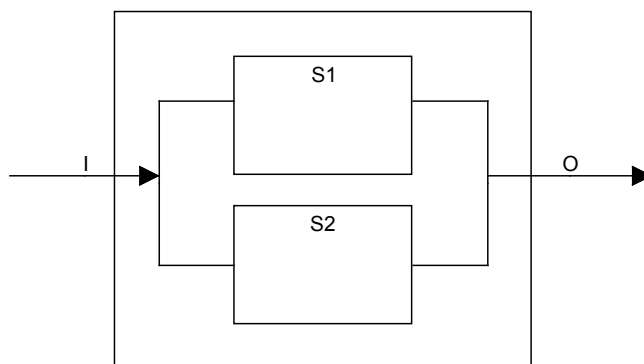
- Deterministický
- Receptivní – pro každý vstup existuje výstup
- S pamětí

Struktura přechodové funkce

$$\left(\underbrace{A}_{\text{Input}}, \underbrace{B}_{\text{odkud}}, \underbrace{C}_{\text{kam}}, \underbrace{D}_{\text{Output}} \right)$$

3.1 Synchronizovaná paralelní kompozice systémů

- „Hodiny“ jdou na každý přechod do každého z automatů, automat se tedy nachází vždy ve více stavech najednou.
- Pokud chceme výsledný systém zakreslit jako jeden automat, sestojíme *kartézský součin vstupů a výstupů*.

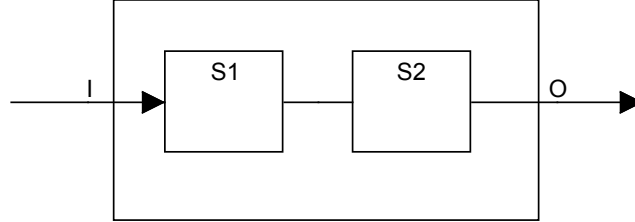


Obrázek 1: Synchronizovaná paralelní kompozice systémů

3.2 Kaskádní kompozice systémů $S_1 \rightsquigarrow S_2$

Kaskádní kompozici dvou automatů vytvoříme následovně:

- Vytvoříme kartézský součin všech stavů.
- Výstup jednoho automatu (S_1) je vstupem automatu druhého (S_2).



Obrázek 2: Kaskádní kompozice systémů

4 Linear temporal logic (LTL)

- Pokud chceme nějakou vlastnost dokazovat, snažíme se najít protipříklad. V případě automatového modelu např. hledáme cykly.

4.1 Vlastnosti a cesty

- První element cesty

$$\pi(0) \models p$$

• ○ ○ ○ ○ ...

- Další element cesty („neXt“)

$$\pi^1 \models p$$

○ • ○ ○ ○ ○ ...

- Časem („Future“)

$$\exists k \geq 0, \pi^k \models p$$

○ ○ ○ • • • ○ ○ ○ ...

- Vždy („Globally“)

$$\forall k \geq 0, \pi^k \models p$$

• • • • • ...

- Do té doby („Until“)

$$\exists i, \pi^i \models q \text{ a } \forall j < i, \pi^j \models p$$

p : • • • ○ ○ ○ ○ ○ ...

q : ○ ○ ○ • ○ ○ ○ ○ ...

- Pokud ještě („Release“)

p : • • • • ○ ○ ○ ○ ...

q : • • • • ! ○ ○ ○ ○ ...

Speciální případy:

- FGg : Časem bude nějaká vlastnost platit navždy, tzn. jakmile začne platit, nikdy už neskončí.
- Gfg : Vždy se lze časem vlastnosti g dočkat. Tento případ se od „samotného časem“ liší tak, že „samotné časem“ může a nemusí nastat pouze jednou.

Upozornění

Styl zápisu

$$red \rightarrow Ggreen$$

říká, že v prvním stavu musí platit vlastnost „red“.

4.2 Převod do základní logiky

$$\neg(Fp) = G\neg p$$

5 Testování a ověřování modelů

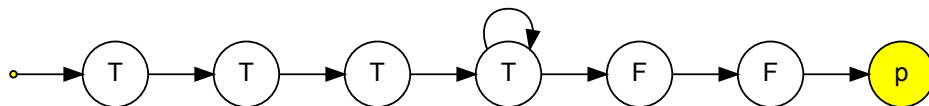
Pozor v této kapitole se používá namísto „automatového modelu“ nový termín „*přechodový systém*“, který se skládá z

- Množiny stavů S (stavový prostor),
- Neprázdne množiny $S_0 \subseteq S$ počátečních stavů,
- Přechodové relace $R \subseteq S \times S$ tak, že pro každé $s \in S$ existuje $s' \in S$ tak, že $(s, s') \in R$.

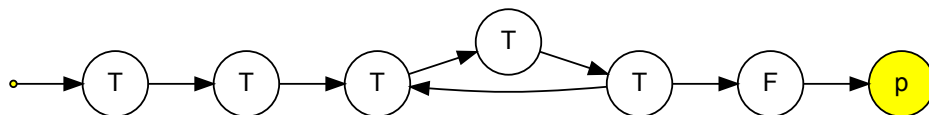
tl;dr: Přechodový systém je zjednodušený konečný automat – neřeší vstupy I ani výstupy O .

5.1 Bounded Model Checking

- Ověřujeme pro cestu určité délky
- Pro cesty délky 0 (ještě jsme nevstoupili do cyklu) platí **všechny vlastnosti triviálně**.
- Při dokazování „ F “ pomocí metody BNC je vlastnost pravdivá do té doby, **než narazíme na cyklus** (viz obrázky níže). Důvodem je, že „nevidíme“ do budoucnosti a dokud žádný cyklus nepotkáme, předpokládáme, že ho ani nepotkáme. Pozor – stav, ve ze kterého cyklus začíná se stále považuje za pravdivý.



Obrázek 3: BNC a dokazování „Future“



Obrázek 4: BNC a dokazování „Future“

5.2 Satisfiability (SAT)

- Ověřování splnitelnosti Booleovské formule.

5.2.1 Základní pravidla při vyšetřování formule

- **Simplifikace** – pokud je proměnná **TRUE**, můžeme škrtnout celou závorku (klausuli).
- **Eliminace** – pokud je proměnná **FALSE**, můžeme ji vyjmout.

5.3 Davis-Putnam-Logemann-Loveland algoritmus (DPLL)

Rozšíření SAT modelu o:

- **Unit Propagation** – pokud nějaká klauzule obsahuje jen jeden literál, ihned do tohoto literálu dosadíme hodnotu.
- **Pure Literal Elimination** – pokud se proměnná v celém výrazu vyskytuje buď jen pozitivně nebo jen negativně, dosadíme do této proměnné hned hodnotu, aby vyšla **TRUE**.

5.4 Neomezené testování modelů

Invariant Podmínka, která musí být splněna před a po každém průchodu. Musí tedy platit na cestě libovolné délky nebo-li musí platit na všech dosažitelných stavech.

Induktivní invariant Invariant, který splňuje indukční podmínky.

Triviální invariant Invariant je triviální, pokud obsahuje všechny stavy přechodového systému.

Existují dvě metody dokazování invariantu:

1. postupným průchodem automatem,
2. důkaz sporem.

5.4.1 Induktivní podmínky

1. Každý počáteční stav splňuje p , tj.

$$S_0 \subseteq \mathcal{I}(p) \iff \forall x. S_0(x) \Rightarrow V(x)$$

2. Důkaz, že pokud x splňuje p , a x' je výsledkem přechodu z x , pak x' také splňuje p , tj.

$$\{x' | x \in \mathcal{I}(p), (x, x') \in T\} \subseteq \mathcal{I}(p) \iff \forall x \forall x'. [V(x) \wedge T(x, x')] \Rightarrow V(x')$$

Negované indukční podmínky pro využití v důkazu sporem:

- $\neg [\forall x. S_0(x) \Rightarrow V(x)] \Leftrightarrow \boxed{\exists x. S_0(x) \wedge \neg V(x)}$
- $\neg [\forall x \forall x'. [V(x) \wedge T(x, x')] \Rightarrow V(x')] \Leftrightarrow \boxed{\exists x \exists x' [V(x) \wedge T(x, x') \wedge \neg V(x')]}$

6 Petriho sítě

6.1 Základní vlastnosti

Definice:

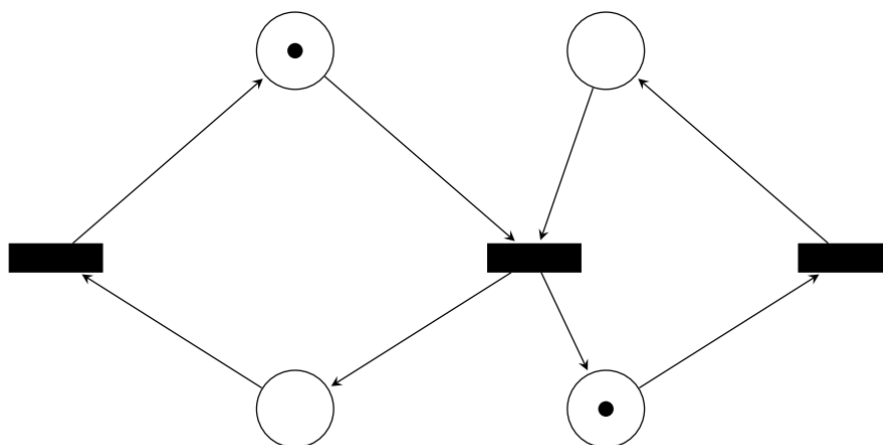
$$(P, T, F, w, M_0).$$

- P = places
- T = transitions
- F = hrany
- w - váhová funkce (implicitně je jedna jinak se ohodnocují hrany)
- M_0 - počáteční značení

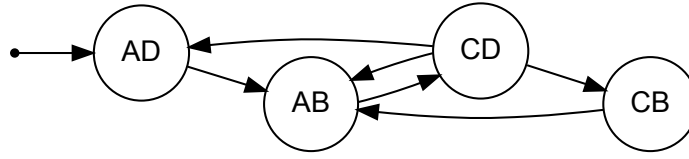
Petriho síť se skládá z míst, přechodů a hran.

- **Značení** = rozmístění tokenů.
- **Dosažitelnost** = značení M je dosažitelné právě když existuje posloupnost odpálení z M_0 do M .
- **(k-)Omezenost** = maximální počet značek v jednom místě (globálně).
- **Živost** = Přechod je živý, pokud z každého dosažitelného značení je možné tento přechod odpálit.
 - Petriho síť je živá, právě když je každý přechod živý.

6.2 Převod do automatu



Obrázek 5: Petriho síť, značení míst dle hodinových ručiček: A, B, C, D



Obrázek 6: Petriho síť převedená do konečného automatu

7 Časované automaty

Základní pojmy

Lokace Místo v automatu (stav).

Invariant Časové omezení v lokaci. Pozor nejedná se o tentýž pojem jako v případě induktivního invariantu.

Stav Konkrétní stav definovaný stavem a jeho časem.

Guard Časové omezení na přechodech.

Časovaný přechod Zůstáváme na jednom místě, ale čas plyne (navyšuje se).

Akční přechod Přechod do jiného místa.

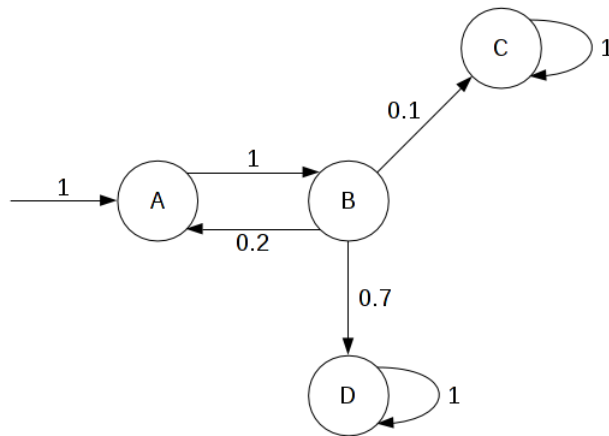
Formální definice časovaného automatu (pětice)

$$A = (S, S_0, X, \mathcal{I}, T).$$

- S je konečná množina lokací
- $S_0 \subseteq S$ je množina počáteční lokace
- X je konečná množina hodin
- $\mathcal{I} : S \rightarrow \mathcal{C}(X)$ jsou invarianty lokací
- $\mathcal{T} \subset S \times \mathcal{C}(X) \times \mathcal{P}(X) \times S$ (přechody, guard, reset)
- Pokud nemá stav invariant, můžeme v něm setrvat neomezeně dlouho

8 Probabilistické modely

- Přechody jsou ohodnoceny pravděpodobnostmi.
- Součet pravděpodobností v každém stavu (všech přechodů z toho stavu) se musí rovnat 1.
- Pokud chceme v probabilistickém modelu spočítat pravděpodobnost, že se dostaneme z místa A do místa B , musíme sestavit lineární soustavu rovnic.



Obrázek 7: Probabilistický model

Ve výše zobrazeném modelu chceme spočítat pravděpodobnost cesty z bodu A do bodu D . Sestavíme tedy pro každý stav rovnici s pravděpodobností cesty do cílového bodu D .

$$x_A = 1 * x_B \quad (1)$$

$$x_B = 0,2 * x_A + 0,1 * x_C + 0,7 * x_D \quad (2)$$

$$x_C = 0 \quad (3)$$

$$x_D = 1 \quad (4)$$

- (1) Ze stavu A vede jen jedna cesta.
- (2) Ze stavu B vedou tři cesty.
- (3) Ze stavu C se do stavu D nikdy nedostaneme, pravděpodobnost je tedy 0.
- (4) V D již jsme, pravděpodobnost je tedy 1.