# Teorie systémů [MI-TES]

Souhrn látky

leden 2014

---

## Obsah

# 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 označkovaný 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

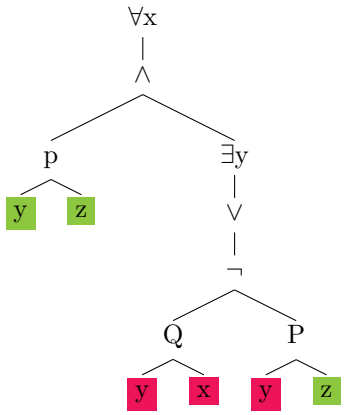
– J. Velebil

**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.

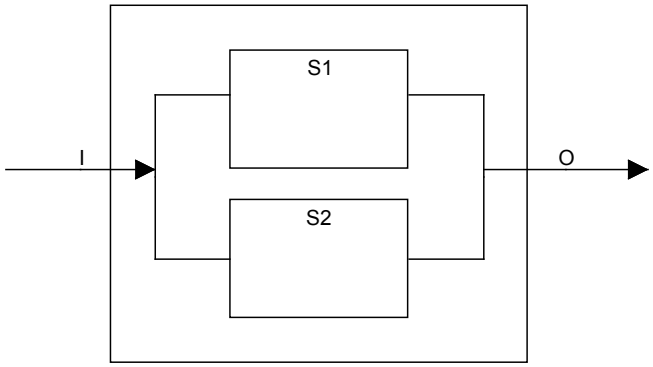
- Deterministický
- Receptivní – pro každý vstup existuje výstup
- S pamětí

**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, sestrojíme *kartézský součin vstupů a výstupů.*

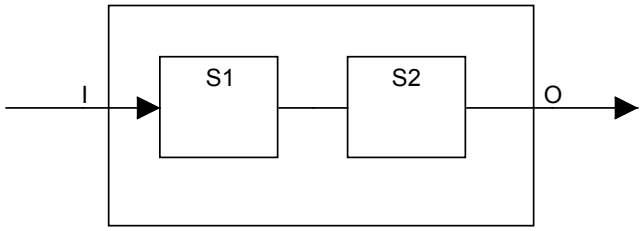



Obrázek 1: Synchronizovaná paralelní kompozice systémů

## 3.2 Kaskádní kompozice systémů $S_1 \rightsquigarrow S_2$

Kaskádní kompozici dvou automatů vytvoříme následovně:

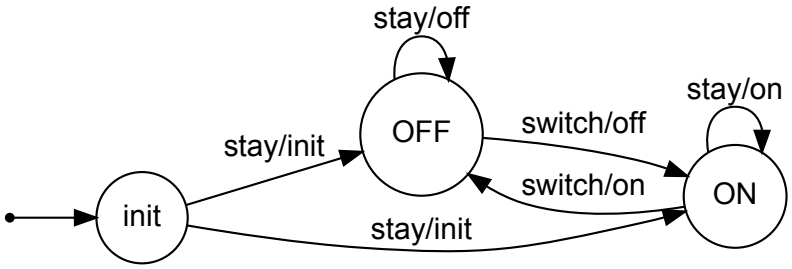
- Vytvoříme kartézský součin všech stavů.
- Výstup jednoho automatu ($S1$) je vstupem automatu druhého ($S2$).




Obrázek 2: Kaskádní kompozice systémů

## 3.3 Delay

- laicky zpožďuje vstup
- Delay $D_{S_0}$ pro množinu $S_0$, $(i, 0) \in D_{S_0} \Leftrightarrow$
  - $o(0) \in S_0$,
  - $\forall k \in N$, $o(k) = i(k-1)$
  - pro libovolné vstupní množiny $I$ a $O$ tak, že $I \subseteq O$, $S_0 \subseteq O$.




Obrázek 3: Ukázka delay

# 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

$$\pi\,(0) \vDash p$$

$$\bullet \circ \circ \circ \ldots$$

- Další element cesty („ne**X**t“)

$$\pi^1 \vDash p$$

$$\circ \bullet \circ \circ \circ \ldots$$

- Časem („**F**uture“)

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

$$\circ \circ \circ \bullet \bullet \bullet \circ \circ \circ \ldots$$

- Vždy („**G**lobally“)

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

$$\bullet \bullet \bullet \bullet \bullet \ldots$$

- Do té doby („**U**ntil“)

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

$$p : \bullet \bullet \bullet \circ \circ \circ \circ \circ \ldots$$
$$q : \circ \circ \circ \bullet \circ \circ \circ \circ \ldots$$

- Pokud ještě („**R**elease“)

$$p : \bullet \bullet \bullet \bullet \circ \circ \circ \circ \ldots$$
$$q : \bullet \bullet \bullet \underset{!}{\bullet} \circ \circ \circ \circ \ldots$$

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

$$red \rightarrow G\text{green}$$

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

## 4.2 Převod do predikátové logiky

$$\neg\left(\mathbf{F}p\right) = \mathbf{G}\neg p$$

$$\neg\left(\mathbf{G}p\right) = \mathbf{F}\neg p$$

$$\neg\left(\mathbf{X}p\right) = \mathbf{X}\neg p$$

$$\mathbf{F}p \Leftrightarrow \mathbf{TU}p$$

$$\mathbf{G}p \Leftrightarrow \mathbf{FR}p$$

$$\neg\left[p\mathbf{U}q\right] = \neg p\mathbf{R}\neg q$$

$$\neg\left[p\mathbf{R}q\right] = \neg p\mathbf{U}\neg q$$

**Příklad**

Převeďte LTL formuli do výrazu predikátové logiky (bez temporálních operátorů):

$$p \Rightarrow \mathbf{G}p \Leftrightarrow \neg\left[p \wedge \left(\mathbf{F}\neg g\right)\right].$$

$$\begin{array}{rcl} \neg p \vee \mathbf{G}p & \Leftrightarrow & \neg p \vee \neg\mathbf{F}\neg g \\ \underbrace{\pi\left(0\right) \nvDash p}_{p} \vee \underbrace{\left(\forall k \geq 0\right)\left(\pi^k \vDash q\right)}_{\mathbf{G}} & \Leftrightarrow & \pi\left(0\right) \nvDash p \vee \neg \underbrace{\left[\left(\exists k \geq 0\right) \neg\left(\pi^k \vDash q\right)\right]}_{\mathbf{F}\neg g} \end{array}$$

Nyní převedeme pravou stranu do výrokové logiky bez symbolu „$\vDash$“

$$\pi\left(0\right) \quad \nvDash \quad p \vee \neg\left[\left(\exists k \geq 0\right) \neg\left(\pi^k \vDash q\right)\right] \Leftrightarrow \boxed{\pi\left(0\right) \in \mathcal{I}\left(p\right) \vee \left(\forall k \geq 0\right)\left(\pi\left(k\right) \in \mathcal{I}\left(p\right)\right)}.$$

# 5 Testování a ověřování modelů

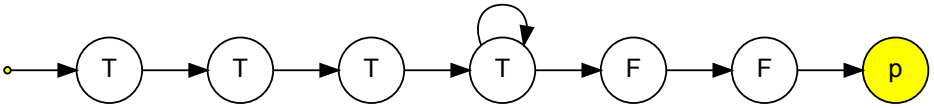
Pozor v této kapitole se používá namísto „automatového modelu“ nový termín „*přechodový systém*“, který se skládá z

- Množiny stavů $S$ (stavový prostor),
- Neprázdné 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 $(s,\, s') \in R$.

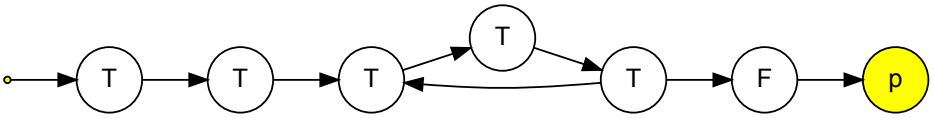
tl;dr: Přechodový systém je zjednodušený konečný automat – neřeší vstupy $I$ ani výstupy $O$.

## 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 4: BNC a dokazování „Future“




Obrázek 5: 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 škrtnou celou závorku (klauzuli).
- **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 induktivní podmínky.

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

Existují dvě metody dokazování invariantu:

1. postupným průchod automatem,
2. důkaz sporem.

### 5.4.1 Induktivní podmínky

1. Každý počáteční stav splňuje $p$, tj.

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

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

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

Negované induktivní 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:

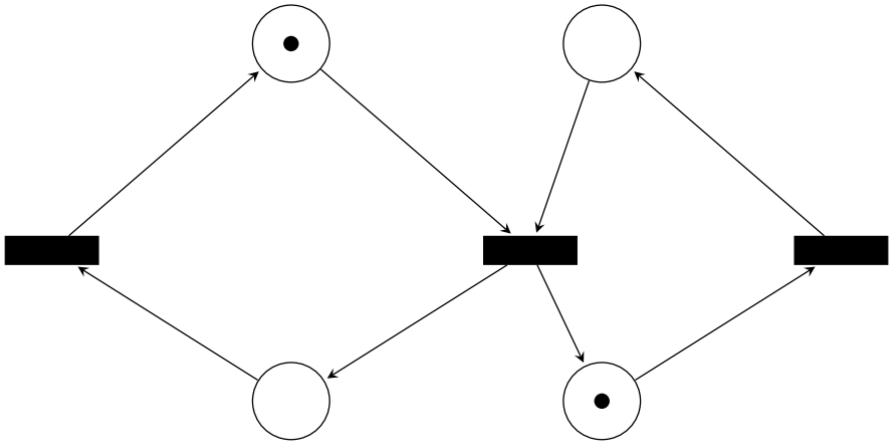
$$(P,\, T,\, F,\, w,\, M_0)\,.$$

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

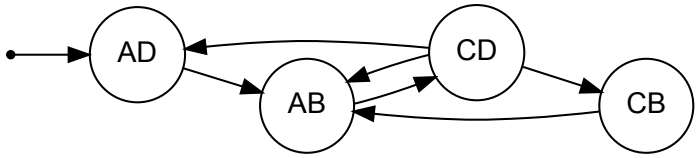
Petriho síť se skládá z míst, přechodů a hran.

- **Značení** = rozmístění tokenů.
- **Dosažitelnost =** značení M je dosažitelné právě když existuje posloupnost odpálení z M0 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 6: Petriho síť, značení míst dle hodinových ručiček: A, B, C, D




Obrázek 7: Petriho síť převedená do konečného automatu

## 7 Časované automaty

### 7.1 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.

### 7.2 Definice

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
- $\mathcal{I} : S \to \mathcal{C}(X)$ jsou invarianty lokací
- $\mathcal{T} \subset S \times \mathcal{C}(X) \times \mathcal{P}(X) \times S$ (přechody, guard, reset)

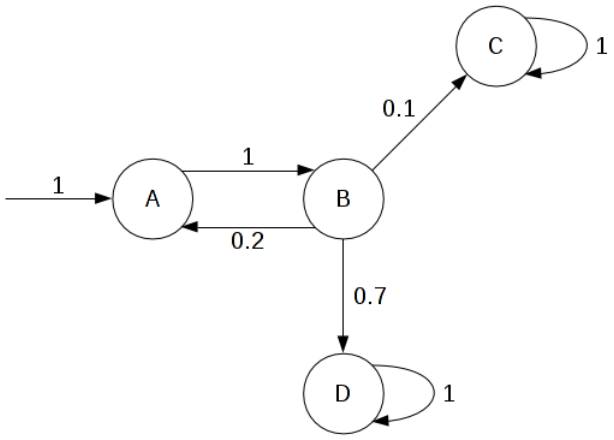
- Pokud nemá stav invariant, můžeme v něm setrvat neomezeně dlouho.
- 2 typy přechodů:
  - časovaný přechod - ze stavu A do stavu A, značíme $(s, v) \xrightarrow{d} (s, v + d)$ ... čas ubíhá v určité lokaci
  - akční přechod - ze stavu A do stavu B, značíme $(s, v) \xrightarrow{\triangle} (s', v')$ ... změna lokace

**Struktura přechodové funkce**

$$\left( \underbrace{A}_{\text{odkud}}, \underbrace{\mathbf{T}}_{\text{guard}}, \underbrace{\{x\}}_{\text{resetuj hodiny x}}, \underbrace{B}_{\text{kam}} \right)$$

# 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 8: Probabilistický model

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

$$
\begin{aligned}
x_A &= 1 * x_B && (1)\\
x_B &= 0,2 * x_A + 0,1 * x_C + 0,7 * x_D && (2)\\
x_C &= 0 && (3)\\
x_D &= 1 && (4)
\end{aligned}
$$

- (1) Ze stavu vede jen jedna cesta.
- (2) Ze stavu 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.