Príklad 1

ONDREJ DUBAJ
xdubaj00

- daný automat obsahuje zero - lch
  - hľadáme ľubý časovo konvergentný nekonečný beh, ktorý bude obsahovať nekonečné množstvo diskrétnych krokov a zároveň nebude spĺňať podmienky neexistencie zeno behu (hodiny sú aspoň 1x resetované a zároveň aspoň jeden krok cyklu nevyžaduje beh času)
  - zeno beh automatu A_1:

$$p = (A, x=0, y=0) \xrightarrow{a_1} (B, x=0, y=0) \xrightarrow{a_2} (C, x=0, y=0) \xrightarrow{a_4}$$

$$\xrightarrow{a_4} (A, x=0, y=0)$$

  - beh je konvergentný a nekonečný
  - obsahuje nekonečné množstvo diskrétnych krokov
  - žiadny krok cyklu nevyžaduje beh času

- daný automat obsahuje timelock
  - je to konfigurácia $c = (l, v)$ pre ktorú platí $Paths_{div}(c) = \emptyset$
  - beh vedúci do timelocku:

$$(A, x=0, y=0) \xrightarrow{a_1} (B, x=0, y=0) \xrightarrow{a_2} (C, x=0, y=0) \xrightarrow{a_4}$$

$$\xrightarrow{a_4} (A, x=0, y=0) \xrightarrow{10} (A, x=10, y=10)$$

timelock

  - z daného timelocku je možné robiť iba časové kroky, diskrétne nie a aj to iba dovtedy, pokiaľ bude $y < 15$ → y sa limitne bude blížiť k 15 v nekonečnom množstve časových krokov →
  → konvergentný nekonečný beh

časový

  - množina $Paths_{div}(A, x=10, y=10) = \emptyset$

$$(A, x=10, y=10) \xrightarrow{1} (A, x=11, y=11) \xrightarrow{0,1} (A, x=11,1, y=11.1) \xrightarrow{0,01} \dots$$

Príklad 2

- dôkaz UNION
  - máme 2 jazyky nad konečnými abecedami L_1 a L_2.
  - máme 2 automaty s množinami koncových stavov A_1, A_2, kde:

$$L_1(A_1) = \{w_1 \mid A_1 \text{ prijíma } w_1\}$$

$$L_2(A_2) = \{w_2 \mid A_2 \text{ prijíma } w_2\}$$

kde A_1, A_2 sú definované nasledovne:

$$A_1 = (Loc_1, Acc_1, C_1, \hookrightarrow_1, Loc_{01}, Inv_1, AP_1, L_1, Loc_{acc1})$$

$$A_2 = (Loc_2, Acc_2, C_2, \hookrightarrow_2, Loc_{02}, Inv_2, AP_2, L_2, Loc_{acc2})$$

- keďže L_1, L_2 sú jazyky nad konečnými abecedami, množina slov patriaca do daného jazyka bude tiež konečná a taktiež automat prijímajúci slová daného jazyka bude mať konečný počet stavov a pravidiel
- a automatov A_1, A_2 môžeme zostrojiť A_3, kde:

$$A_3 = (Loc_1 \cup Loc_2, Acc_1 \cup Acc_2, C_1 \cup C_2, \hookrightarrow_1 \cup \hookrightarrow_2, Loc_{01} \cup Loc_{02}, Inv_1 \cup Inv_2, AP_1 \cup AP_2, L_1 \cup L_2, Loc_{acc1} \cup Loc_{acc2})$$

- automat A_3 bude prijímať množinu slov, ktorá vyplynie zjednotením množín slov prijímaných automatmi A_1, A_2 pretože bude mať konečný počet stavov a pravidiel zahrňujúci všetky stavy a pravidlá automatov A_1, A_2
- z toho vyplýva, že automat A_3 bude prijímať množinu slov patriacu do jazyka L_3, kde:

$$L_3(A_3) = \{w_3 \mid A_3 \text{ prijíma } w_3\}$$

množín

kde množina slov jazyka L_3 zahrňuje zjednotenie všetkých slov jazykov L_1, L_2

- z vyššie uvedeného vyplýva, že zjednotením jazykov $L_1 \cup L_2$ vznikne jazyk L_3, ktorý obsahuje všetky slová jazykov L_1, L_2 a teda operácia zjednotenia jazykov prijímaných časovými automatmi je uzavretá, nakoľko vieme zostrojiť časový automat A_3 prijímajúci jazyk L_3 zjednotením jednotlivých prvkov N-tic automatov A_1, A_2.

- dôkaz KONKATENACE
- máme 2 jazyky nad konečnými slovami L_1 a L_2
- máme 2 automaty s množinami koncových stavov A_1, A_2, kde:

$$L_1(A_1) = \{ w_1 \mid A_1 \text{ prijíma } w_1 \}$$
$$L_2(A_2) = \{ w_2 \mid A_2 \text{ prijíma } w_2 \}$$

kde A_1, A_2 sú definované rovnako, ako v dôkaze UNIONU v prvej časti tohto príkladu

- keďže L_1, L_2 sú jazyky nad konečnými slovami, množina slov patriaca do daného jazyka bude tiež konečná a taktiež automat prijímajúci slová daného jazyka bude mať konečný počet stavov a pravidiel
- z automatov A_1, A_2 môžeme skonštruovať automat A_3, kde:

$$A_3 = (Loc_1 \cup Loc_2, Act_1 \cup Act_2 \cup Act_N, C_1 \cup C_2, \hookrightarrow_1 \cup \hookrightarrow_2 \cup \hookrightarrow_N, Loc_{01}, Inv_1 \cup Inv_2, AP_1 \cup AP_2, L_1 \cup L_2, Loc_{acc_2})$$

- ~~automat A_3 bude~~

kde $Act_N = \{ concat\text{-}word \}$

$$\hookrightarrow_N \subseteq Loc_{acc_1} \times CC_N(C) \times Act_N \times \forall g \in C_2 . g = 0 \times Loc_{02}$$

kde $CC_N(C) = \emptyset$

- automat A_3 bude prijímať množinu slov, ktorá vyhovuje konkatenácii dvoch karteziánskeho súčinu slov jazykov L_1, L_2, kde karteziánsky súčin slov je definovaný ako:

$$L_1 \times L_2 \stackrel{def}{=} \{ (w_1, w_2) \mid w_1 \in L_1 \wedge w_2 \in L_2 \}$$

- a taktiež vyplýva, že automat A_3 bude prijímať množinu slov patriacu do jazyka L_3, kde:

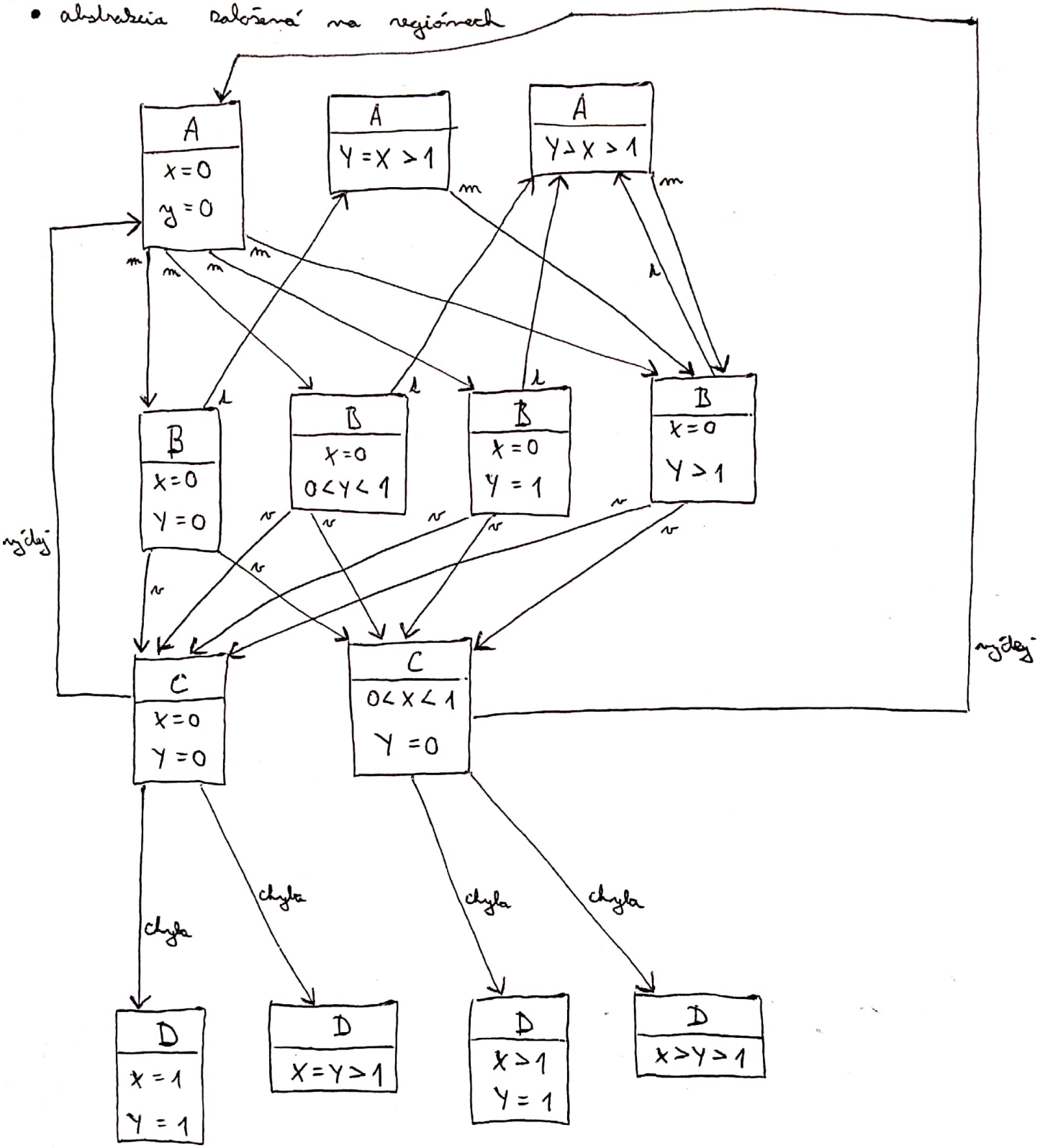
$$L_3(A_3) = \{ w_3 \mid A_3 \text{ prijíma } w_3 \}$$

kde množina slov jazyka L_3 zahŕňa konkatenáciu slov jazykov L_1, L_2

- Z vyššie uvedeného vyplýva, že konkatenáciou jazykov $L_1 \cdot L_2$ vznikne jazyk L_3, ktorý sú prijímané automatom A_3 a keďže operácia konkatenácia jazykov prijímaných časovými automatmi je uzavretá, nakoľko vieme zostrojiť časovaný automat A_3 prijímajúci jazyk L_3

Príklad 3

- abstrakcia založená na regiónoch



vysvetlivky:

v ~ volba - šípa

l ~ šimsoud

m ~ mince

- stav v ktorom platí predikát error je dostupný, pokiaľ existuje cesta do stavu D, kde error $\in L(D)$
  - príklad cesty:

$$(A, x=0, y=0) \xrightarrow{\text{mince}} (B, x=0, y=0) \xrightarrow{\text{volla - kava}} (C, x=0, y=0) \xrightarrow{1.5} \xrightarrow{1.5} (C, x=1.5, y=1.5) \xrightarrow{\text{chyba}} (D, x=1.5, y=1.5)$$

- máme automat A_2 a TCTL formulu $\phi = \exists (\text{run } U^{\leq 2} \text{ error})$
  - nájdeme množinu splňujúcich konfigurácií $Sat(\phi)$ a množinu počiatočných konfigurácií $Init_{A_2}$ automatu A_2
  - v prípade, že bude platiť $Init_{A_2} \subseteq Sat(\phi)$, tak automat A_2 splňuje formulu ϕ

$$Init_{A_2} = \{(A, x=0, y=0)\}$$
$$Sat(\phi) = \{(A, x=0, y=0), (B, x=0, y=0), (C, x=0, y=0), \dots\}$$

  - množina $Sat(\phi)$ je nekonečná, nakoľko existuje nekonečne veľa konfigurácií automatu A_2, kde platí $s \models \exists (\text{run } U^{\leq 2} \text{ error})$, kde $s \in Sat(\phi)$ a $\pi \models (\text{run } U^{\leq 2} \text{ error})$ pre nejakú cestu $\pi \in Paths_{dis}(s)$
  - pre naše účely je podstatné, že platí

$$Init_{A_2} \subseteq Sat(\phi)$$

a čoho vyplýva, že Automat A_2 splňuje formulu ϕ, a teda platí:

$$\boxed{A_2 \models \exists (\text{run } U^{\leq 2} \text{ error})}$$

- máme stav $s = (B, x = y = 0)$ automatu A_2 a TCTL formulu

$\phi = \forall (run\ U^{<2}\ init)$

- chceme dokázať $s \models \phi$
- aby mal $s \models \phi$ byť platné, tak pre každú cestu $\pi \in Path_{div}(s)$ musí platiť $\pi \models (run\ U^{<2}\ init)$
- vyberáme si množinu $Path_{div}(s)$, z ktorej sa pokúsime vyjsť iba pár prvkov, nakoľko nám stačí nájsť iba 1 divergentnú cestu π, pre ktorú nebude platiť $\pi \models (run\ U^{<2}\ init)$ aby sme dokázali, že $s \models \phi$ nie je platné

$Path_{div}(s) = \{$

$(B, x = y = 0) \xrightarrow{3} (B, x = y = 3) \xrightarrow{time} (A, x = y = 3) \xrightarrow{mince} (B, x = 0, y = 3) \ldots$

$(B, x = y = 0) \xrightarrow{1.5} (B, x = y = 1.5) \xrightarrow{time} (A, x = y = 1.5) \xrightarrow{mince} (B, x = 0, y = 1.5) \ldots$

$\ldots$

$\}$

- vidíme, že cesta $\pi_1 = (B, x = y = 0) \xrightarrow{3} (B, x = y = 3) \xrightarrow{time} (A, x = y = 3) \xrightarrow{mince}$

$\xrightarrow{mince} (B, x = 0, y = 3) \xrightarrow{?} \ldots$ patrí do $Path_{div}(s)$, avšak

~~nie~~ neplatí $\pi_1 \models (run\ U^{<2}\ init)$, z čoho vyplýva, že vyššie uvedená podmienka neplatí pre všetky cesty π patriace do $Path_{div}(s)$

- z vyššie uvedeného vyplýva, že

$$(B, x = y = 0) \models \forall (run\ U^{<2}\ init)$$

nie je platná