Masarykova univerzita
Fakulta informatiky

Modelování systémů s reálným časem a pravděpodobností

Bakalářská práce

Eva Tesařová

Brno, jaro 2014

Prohlášení

Prohlašuji, že tato bakalářská práce je mým původním autorským dílem, které jsem vypracovala samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používala nebo z nich čerpala, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

Vedoucí práce: doc. RNDr. Jiří Barnat, Ph.D.

Poděkování

Ráda bych poděkovala svému vedoucímu doc. RNDr. Jiřímu Barnatovi, Ph.D., za odborné vedení, jeho rady a čas při zpracování této práce.

Shrnutí

Práce se zaměřuje na formalismy, které umožňují zachytit kombinaci pravděpodobnostních, nedeterministických a časových vlastností systémů. U každého formalismu jsou stručně popsány způsoby specifikace požadavků a techniky používané pro ověřování modelu. Dále je navrženo rozšíření pravděpodobnostních časových automatů, které je porovnáno s nerozšířenou verzí, a je nastíněn způsob, jakým by mohla probíhat verifikace.

Klíčová slova

Markovovy modely, časový automat, pravděpodobnostní časový automat, časové a pravděpodobnostní temporální logiky, ověřování modelu

Obsah

Kapitola 1

Úvod

V posledních letech se stále zvyšuje složitost počítačových systémů, a tedy i snaha ověřit bezpečnost, spolehlivost a efektivitu v oblastech, jakými jsou například medicína, letecký provoz nebo bezdrátové síťování. Jednou z nejrozšířenějších verifikačních technik je testování, které na jednu stranu poskytuje jednoduchý nástroj pro odhalování chyb, na druhou stranu však není schopno zaručit opravdovou bezchybnost systému. Právě ve výše zmiňovaných oblastech proto není vhodné spoléhat se pouze na výsledky testů, neboť přítomnost chyby má často fatální následky. Vyvstává tedy potřeba zavedení technik, které budou schopny zaručit správné chování systému.

Jednou z vhodných metod je *ověřování modelu*, které je založeno na vytvoření formálního modelu verifikovaného systému. Model se z pravidla sestává z lokací, které reprezentují stavy systému, a pravidel popisujících přechod mezi nimi. Pomocí formulí temporální logiky jsou pak vyjádřeny požadavky na systém a je automaticky rozhodnuto, zda systém splňuje specifikaci. Jinými slovy, je dána specifikace ϕ, relace splnitelnosti $\models$, model systému $\mathcal{M}$ a ptáme se, zda-li $\mathcal{M} \models \phi$. Často se také hovoří o metodě ověřování modelu v souvislosti s *analýzou dosažitelnosti*. Typicky zkoumáme, zda je dosažitelný nějaký stav z množiny chybových stavů. Při formální verifikaci je narozdíl od testování jistota, že systém se skutečně chová požadovaným způsobem. To je zaručeno prozkoumáním všech možných chování systému, což s sebou na druhou stranu přináší výpočetní složitost.

Mnoho reálných systémů vykazuje nedeterministické a pravděpodobnostní chování. Často jsou rovněž kladeny požadavky na rychlost odezvy systému. Je tedy vhodné pro jejich modelování volit takové formalismy, které vystihnou tyto vlastnosti.

První kapitola se zabývá formalismy, které kombinují pravděpodobnost a nedeterminismus. Jejich použití je vhodné například pro modelování souběžných systémů, jejichž chování je ovlivněno pravděpodobnostní volbou. Ve druhé kapitole je popsán časový automat, který zachycuje čas i nedeterminismus. Umožňuje pak ověřit pomocí časových temporálních logik vlastnosti jako „Zpráva bude doručena do 5 sekund od odeslání.“. Spojením pravděpodobnostních, časových i nedeterministických vlastností se zabývá třetí kapitola, kde jsou představeny varianty pravděpodobnostních časových automatů.

Cílem práce bylo předložit přehled formalismů s výše uvedenými vlastnostmi, předvést logiky používané k specifikaci vlastností, demonstrovat rozdíly v jejich chování na vhodných příkladech a zmínit způsob, jakým probíhá verifikace. Dále pak diskutovat možnost rozšíření pravděpodobnostních časových automatů, které by umožnilo zachytit setrvání v určitém stavu po dobu ovlivněnou pravděpodobností funkcí. Formálně je rozšíření zadefinováno v poslední kapitole spolu s porovnáním s nerozšířenou verzí.

Kapitola 2

Pravděpodobnostní modely

Pravděpodobnost hraje důležitou roli při analýze širokého spektra komplexních systémů, které přechází přes komunikační, multimediální a bezpečnostní protokoly, randomizované distribuované algoritmy až po biologické systémy [18]. V této kapitole budou představeny Markovovy řetězce a Markovovy rozhodovací procesy, které tvoří jedny z nejpoužívanějších formalismů umožňujících pravděpodobnost zachytit.

Pravděpodobnostní měření

Nejprve uvedeme základní matematické pojmy z oblasti pravděpodobnosti.

Definice 1. σ-algebra nad libovolnou množinou Ω je neprázdný systém podmnožin $\Sigma \subseteq 2^{\Omega}$ splňující následující:

- $\Omega \in \Sigma$
- $A \in \Sigma \Rightarrow \overline{A} \in \Sigma$
- Σ je uzavřená vzhledem k nejvýše spočetnému počtu sjednocení

Množinu Ω nazýváme *základním prostorem*. Představuje všechny možné výsledky. Systém podmnožin Σ označujeme jako *jevové pole*. Nad jednotlivými prvky jevového pole zavedeme pravděpodobnostní měření.

Definice 2. Pravděpodobnostní prostor je trojice (Ω, Σ, Pr), kde

- Ω je základní prostor
- Σ je σ-algebra nad Ω
- $Pr : \Sigma \to [0, 1]$ je pravděpodobnostní funkce splňující
  - $Pr(\Omega) = 1$
  - $Pr(\cup_{i \in I} A_i) = \sum_{i \in I} A_i$ pro každý nejvýše spočetný systém po dvou disjunktních jevů

Pro každou ne nutně spočetnou množinu Q definujeme $\mathcal{D}(Q)$ jako množinu všech diskrétních pravděpodobnostních funkcí $\mu : Q \to \mathbb{R}$ nad množinou Q takových, že existuje spočetná podmnožina $Q' \subseteq Q$, aby platilo $\sum_{q' \in Q} \mu(q') = 1$. Pravděpodobnostní funkce nad množinou Q značená jako μ_a, kde $a \in A$, je definována předpisem $\mu(a) = 1$. Pro ostatní hodnoty je nedefinována.

Součin dvou diskrétních pravděpodobnostních funkcí μ_1 a μ_2 nad množinou Q definujeme jako $\mu_1 \otimes \mu_2 = \mu_1(q_1) \cdot \mu_2(q_2)$ pro všechna $q_1, q_2 \in Q$.

2.1 Markovovy řetězce

Markovův řetězec je model založen na reprezentaci systému pomocí jeho stavů a přechodů mezi nimi. Přechody mezi stavy jsou čistě pravděpodobnostní a výběr dalšího stavu není závislý na historii systému, tj. závisí pouze na aktuálním stavu, nikoliv na přechodech, které mu předcházely.

Atomické propozice. Atomické propozice jsou tvrzení, která vyjadřují určité vlastnosti stavů. Je nutné, aby byla tato tvrzení algoritmicky ověřitelná. Atomickými propozicemi jsou typicky výrazy nad proměnnými. Množinu atomických propozic budeme značit jako AP.

Definice 3. [19]Markovův řetěz (zkratka DTMC z anglického discrete-time Markov chain) je čtveřice $\mathcal{D} = (S, s_0, P, L)$, kde

- S je množina stavů,
- $s_0 \in S$ je počáteční stav,
- $P : S \times S \to [0, 1]$ je matice pravděpodobnosti přechodu splňující $\sum_{s' \in S} P(s, s') = 1$ pro všechna $s \in S$,
- $L : S \to 2^{AP}$ je značkovací funkce přiřazující každému stavu množinu atomických propozic z množiny AP.

Cesta v DTMC je neprázdná posloupnost $\omega = s_0 s_1 \dots$ taková, že pro každé $i \in \mathbb{N}$ platí $P(s_i, s_{i+1}) > 0$.

Používáme následující značení:

- $Path_s$ je množina všech nekonečných cest začínajících ve stavu s. Alternativně budeme označovat nekonečné cesty jako běhy.
- $Path_s^{fin}$ označuje množinu všech konečných cest začínajících ve stavu s.

Na obrázku 2.1 je znázorněn jednoduchý Markovovův řetězec modelující házení mincí. Z iniciálního stavu q_0 je učiněna pravděpodobnostní volba, zda padne hlava nebo orel. Obě možnosti mají stejnou pravděpodobnost rovnu $\frac{1}{2}$.

2.1.1 Pravděpodobnostní měření a DTMC

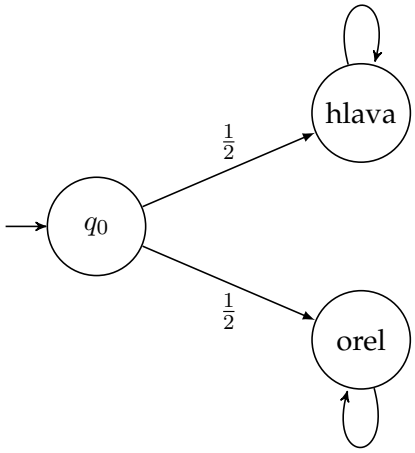
[19] Cylindrická množina $cyl(\pi)$ pro konečnou cestu π a množinu nekonečných cest Ω je množina všech nekonečných cest z Ω, jejichž prefixem je π. Tedy

$$cyl(\pi) = \{\omega \in \Omega \mid \pi \text{ je prefixem } \omega\}.$$

Pravděpodobnost $\mathcal{P}(\pi)$ konečné cesty $\pi = s_0 s_1 \dots s_n$ je definována jako

- $\mathcal{P}(\pi) = 1$, je-li cesta je tvořena jediným stavem s,
- $\mathcal{P}(\pi) = P(s_0, s_1) \cdot P(s_1, s_2) \cdot \dots \cdot P(s_{n-1}, s_n)$ jinak.

Obrázek 2.1: Příklad Markovova řetězce



Pravděpodobnostní prostor
Pravděpodobnostní prostor pro DTMC $\mathcal{D}$ je trojice (Ω, Σ, Pr):

- Základní prostor Ω představuje množina všech běhů.
- Jevové pole Σ definujeme jako nejmenší σ-algebru obsahující všechny cylindrické množiny $cyl(\pi)$, kde π je libovolná konečná cesta.
- Pro cylindrické množiny definujeme $Pr(cyl(\pi)) = \mathcal{P}(\pi)$, pravděpodobnost ostatních jevů je dána vlastnostmi pravděpodobnostní funkce.

2.2 Markovovy rozhodovací procesy

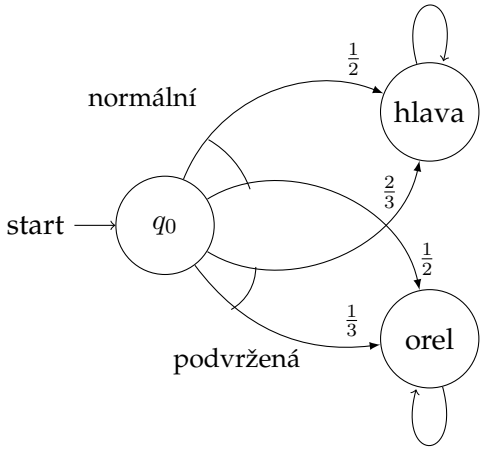
Markovovy rozhodovací procesy umožňují zachytit jak pravděpodobnostní, tak nedeterministické vlastnosti systémů. Nedeterministické chování často vykazují systémy interagující nějakým způsobem s okolím – tedy například většina dnešních aplikací, které své chování přizpůsobují volbě uživatele, či paralelní a distribuované systémy. Zjednodušeně řečeno, Markovovův rozhodovací proces je Markovův řetězec, který má navíc možnost nedeterministické volby mezi pravděpodobnostními funkcemi určujícími přechody.

Definice 4. Markovův rozhodovací proces (MDP z anglického *Markov decision process*) je pětice $\mathcal{M} = (S, s_0, Act, Steps, L)$, kde

- S je množina stavů,
- $s_0 \in S$ je počáteční stav,
- Act je množina akcí,
- $Steps : S \times Act \to \mathcal{D}(S)$ je pravděpodobnostní přechodová funkce,
- $L : S \to 2^{AP}$ je značkovací funkce přiřazující každému stavu množinu atomických propozic z množiny AP.

Obrázek 2.2 zachycuje Markovův rozhodovací proces znázorňující systém se dvěma tlačítky *normální* a *podvržená*. Stisknutí tlačítka provádí vnější uživatel a učiní tak volbu mezi použitím normální mince (hlava i orel padají se stejnou pravděpodobností) nebo podvržené (hlava padá s pravděpodobností $\frac{2}{3}$). Po stisknutí tlačítka systém učiní pravděpodobnostní rozhodnutí o tom, jestli padla hlava nebo orel. Uživatel stojí vně systému a o jeho rozhodnutí nemůžeme z hlediska pravděpodobnosti výběru jednotlivých tlačítek nic předpokládat, proto je nezbytné použít nedeterminismus.

Obrázek 2.2: Ukázka MDP [23]



Funkce $A : S \rightarrow 2^{Act}$ přiřazuje každému stavu množinu akcí, které v něm lze provést. Tedy $A(s) = \{a \in Act \mid Steps(s, a)$ je definováno$\}$.

MDP pracuje tak, že v každém stavu s je nejprve nedeterministicky vybrána akce $a \in A(s)$ a poté s ohledem na pravděpodobnostní funkci $Steps(s, a) = \mu$ učiněn přechod do stavu s', pro který $\mu(s') > 0$.

Cesta v MDP je (konečná či nekonečná) posloupnost $\pi = s_0(a_1, \mu_1)s_1(a_2, \mu_2)\ldots$, kde pro každé $i \in \mathbb{N}$ platí $a_{i+1} \in A(s_i), s_i \in S, \mu_i = Steps(s_i, a_{i+1}), \mu_i(s_{i+1}) > 0$. Jako $\pi(i)$ budeme označovat i-tý navštívený stav, přičemž počáteční stav považujeme za nultý navštívený. Sufix cesty π začínající i-tým stavem značíme jako π^i.

Bez újmy na obecnosti můžeme předpokládat, že MDP je neblokující (tj. v každém stavu může udělat přechod do nového stavu). Pro každý stav s, ze kterého nevede žádný pravděpodobnostní přechod, přidáme (s, a, μ_s) do množiny $Steps$.

Podobně jako v případě DTMC používáme následující značení:

- $Path_s$ je množina všech nekonečných cest začínajících ve stavu s. Alternativně budeme označovat nekonečné cesty jako běhy.
- $Path_s^{fin}$ označuje množinu všech konečných cest začínajících ve stavu s.

Strategie

Abychom mohli formálně zavést pravděpodobnostní měření nad cestami, je třeba rozhodnout nedeterminismus vzniklý volbou akce (potažmo tedy i volbou pravděpodobnostní distribuce). Každé možné řešení nedeterministické volby je reprezentováno *strategií*, která udává, jakou akci máme zvolit jako další v závislosti na historii cesty.

Definice 5. [19]Strategie (též označována jako plánovač) MDP $\mathcal{M} = (S, s_0, Act, Steps, L)$ je funkce $\sigma : Path^{fin} \to Act$ mapující každou konečnou cestu $\pi = s_0(a_1, \mu_1) \dots s_n$ na dvojici (a_{n+1}, μ_{n+1}), kde $a_{n+1} \in A(s_n), \mu_{n+1} = Steps(s_n, a_{n+1})$.

Množinu všech možných strategií nad MDP $\mathcal{M}$ budeme značit jako $Adv_{\mathcal{M}}$.
S každou strategií σ spojíme množinu $Path_s^\sigma$ všech cest $\pi = s_0(a_1, \mu_1) \dots s_n \in Path_{s_0=s}$, pro které platí na všech pozicích $i \geq 0 : \sigma(s_0(a_1, \mu_1) \dots s_i) = (a_{i+1}, \mu_{i+1})$.

2.2.1 Pravděpodobnostní měření a MDP

K definování pravděpodobnostního prostoru MDP využijeme strategií. Neformálně řečeno, každá strategie pro MDP nám udává jistý DTMC, který zachovává pravděpodobnostní volby.

Indukovaný DTMC

Definice 6. [19]Pro MDP $\mathcal{M} = (S, s_0, Act, Steps, L)$ a strategii $\sigma \in Adv_{\mathcal{M}}$ definujeme indukovaný DTMC $\mathcal{D}_{\mathcal{M}}^\sigma = (S', s_0, P, L)$, kde

- množina stavů S je množina cest $Path_{s_0}^{fin}$,
- iniciální stav je s_0, tj. cesta tvořená jediným stavem s_0,
- $P(\pi s\, , \pi') = \begin{cases} \mu(s') & \text{jestliže } \pi' = \pi(a, \mu)s', \sigma(\pi) = (a, \mu), \\ 0 & \text{jinak.} \end{cases}$

Mezi nekonečnými cestami MDP se strategií σ a DTMC $\mathcal{D}_{\mathcal{M}}^\sigma$ existuje bijekce.

Definice 7. Buď DTMC $\mathcal{D}_{\mathcal{M}}^\sigma$ s pravděpodobnostním prostorem (Ω, Σ, Pr) indukovaný DTMC pro MDP $\mathcal{M}$ a strategií σ. Pak MDP $\mathcal{M}$ se strategií σ má pravděpodobnostní prostor $(Path_{\mathcal{M}}^\sigma, \Sigma^\sigma = \Sigma, Pr^\sigma = Pr)$.

2.3 Pravděpodobnostní logiky

Při formální verifikaci systému vzniká potřeba se přesně matematicky vyjádřit o vlastnostech, které chceme v systému verifikovat. Typicky chceme umět vyjádřit, že v systému nastane někdy určitý jev nebo že od určité doby bude platit nějaké tvrzení. Pro popis těchto vlastností je vhodné využít některé z temporálních logik, které narozdíl od klasické výrokové nebo predikátové logiky dobře postihují chování systému v čase.

Existuje více temporálních logik, které jsou vhodné k popisu vlastností MDP a DTMC. V textu jsme se zaměřili na pravděpodobnostní rozšíření *lineární temporální logiky* (LTL), která nahlíží na budoucnost systému jako na uspořádání stavů do lineární množiny, a *logiky výpočetního stromu* (CTL z anglického *computation tree logic*), která zachycuje budoucnost systému jako stromovou hierarchii. První z logik se vyjadřuje o platnosti atomických propozic v jednotlivých bězích systému, druhá ověřuje platnost formule oproti stromu možných běhů z určitého stavu.

2.3.1 Pravděpodobnostní logika výpočetního stromu (PCTL)

Nejprve představíme syntaxi pravděpodobnostní logiky výpočetního stromu (PTCTL z anglického *probabilistic computation tree logic*), která vznikla z CTL výměnou existenčního a univerzálního kvantifikátoru za pravděpodobnostní operátor $P_{\bowtie k}$. Formule $P_{\bowtie k}(\phi)$ je pak pravdivá, jestliže pravděpodobnost splnění formule cesty ϕ vyhovuje omezení $\bowtie k$ ($\bowtie k$ je porovnání s konstantou $k \in [0,1]$). Formule cesty ($\mathrm{X}\,\phi$, $\phi\,\mathrm{U}\,\phi$) se mohou objevit pouze v rámci pravděpodobnostního operátoru obdobně, jako je tomu v případě CTL.

Definice 8. Nechť AP je množina atomických propozic, $k \in [0,1]$, $\bowtie \in \{<, >, =, \leq, \geq\}$, $p \in AP$, pak syntaxe PCTL formule ϕ je dána následujícím předpisem:

$$\begin{aligned} \phi &:= \quad p \mid \neg\phi \mid \phi \vee \phi \mid P_{\bowtie k}(\psi) \\ \psi &:= \quad \mathrm{X}\,\phi \mid \phi\,\mathrm{U}\,\phi \end{aligned}$$

Syntaktické zkratky F a G definujeme běžným způsobem:

- $\mathrm{F}\,\phi = true\,\mathrm{U}\,\phi$ – někde v běhu platí ϕ
- $\mathrm{G}\,\phi = \neg F \neg\phi$ – v běhu vždy platí ϕ

Modelem PCTL formule je stav. V práci definujeme sémantiku pro stavy Markovova řetězce a Markovova rozhodovacího procesu.

Definice 9. Nechť $\mathcal{D} = (S, s_0, P, L)$ je DTMC, ϕ, ψ jsou PCTL formule, stav $s \in S$, ω cesta v DTMC. Pak definujeme sémantiku následovně:

- $s \vDash p \overset{def}{\iff} p \in L(s)$
- $s \vDash \neg\phi \overset{def}{\iff} s \nvDash \phi$
- $s \vDash \phi \vee \psi \overset{def}{\iff} s \vDash \phi \vee \omega \vDash \psi$
- $s \vDash P_{\bowtie k}(\phi) \overset{def}{\iff} Pr(\{\omega \in Path_s \mid \omega \vDash \phi\}) \bowtie k$
- $\omega \vDash_\psi \mathrm{X}\,\phi \overset{def}{\iff} \omega(1) \vDash \phi$
- $\omega \vDash_\psi \phi\,\mathrm{U}\,\phi \overset{def}{\iff} \exists k. \forall 0 \leq i < k : \omega(i) \vDash \phi \wedge \omega(k) \vDash \psi$

V definici relace splnitelnosti je v případě MDP kvůli nedeterminismu třeba uvažovat všechny možné strategie. Jinak se definice sémantiky neliší od DTMC.

Definice 10. Nechť $\mathcal{M} = (S, s_0, Act, Steps, L)$ je MDP, ϕ, ψ jsou PCTL formule, stav $s \in S$, ω cesta v MDP. Pak definujeme sémantiku následovně:

- $s \vDash p \overset{def}{\iff} p \in L(s)$
- $s \vDash \neg\phi \overset{def}{\iff} s \nvDash \phi$
- $s \vDash \phi \vee \psi \overset{def}{\iff} s \vDash \phi \vee \omega \vDash \psi$
- $s \vDash P_{\bowtie k}(\phi) \overset{def}{\iff} Pr(\{\omega \in Path_s^\sigma \mid \omega \vDash \phi\}) \bowtie k$ pro všechny možné strategie $\sigma \in Adv_{\mathcal{M}}$
- $\omega \vDash_\psi \mathrm{X}\,\phi \overset{def}{\iff} \omega(1) \vDash \phi$
- $\omega \vDash_\psi \phi\,\mathrm{U}\,\phi \overset{def}{\iff} \exists k. \forall 0 \leq i < k : \omega(i) \vDash \phi \wedge \omega(k) \vDash \psi$

2.3.2 Pravděpodobnostní lineární temporální logika

Nejprve představíme nerozšířenou LTL.

Definice 11. Nechť AP je množina atomických propozic, pak syntaxe formule ϕ lineární temporální logiky je dána následujícím předpisem:

$$\phi := p \mid \neg\phi \mid \phi \vee \phi \mid \mathrm{X}\,\phi \mid \phi\,\mathrm{U}\,\phi,$$

kde $p \in AP$.

Stejně jako pro CTL uvažujeme odvozené operátory F a G.
Modelem LTL formule je běh. Definice sémantiky pro DTMC se neliší od MDP.

Definice 12. Buďte ϕ, ψ LTL formule, ω běh (MDP či DTMC), L značkovací funkce, $p \in AP$, kde AP je množina atomických propozic. Pak sémantika je definována následovně:

- $\omega \vDash p \overset{def}{\iff} p \in L(\omega(0))$
- $\omega \vDash \neg\phi \overset{def}{\iff} \omega \nvDash \phi$
- $\omega \vDash \phi \vee \psi \overset{def}{\iff} \omega \vDash \phi \vee \omega \vDash \psi$
- $\omega \vDash \mathrm{X}\,\phi \overset{def}{\iff} \omega^1 \vDash \phi$
- $\omega \vDash \phi\,\mathrm{U}\,\psi \overset{def}{\iff} \exists k.\forall 0 \leq i < k.\omega^i \vDash \phi \wedge \omega^k \vDash \psi$

Každá LTL formule vypovídá pouze o vlastnostech běhů systému. U pravděpodobnostních modelů nás ale zpravidla bude zajímat, jestli cesty splňující LTL formuli mají nenulovou pravděpodobnost, případně jestli je pravděpodobnost cesty splňující formuli vyšší než určitá mez. Podobně jako u CTL použijeme k specifikaci pravděpodobnosti operátor $P_{\bowtie k}$. Při verifikaci LTL vlastností se typicky ptáme, jestli všechny běhy z počátečního stavu mají jisou vlastnost. Z tohoto důvodu tento operátor nepřipojíme pouze k samotné logické formuli jako tomu bylo u CTL, ale spojíme jej i s konkrétním stavem. Formule $P_{\bowtie k}(s, \phi)$ bude splněna, jestliže pravděpodobnost všech cest možných ze stavu s splňující ϕ vyhovuje $\bowtie k$.

Definice 13. Nechť ϕ je LTL formule, s stav DTMC. Pak formule $P_{\bowtie k}(s, \phi)$ je splněna, jestliže $Pr(\{\omega \mid \omega \in Path_s, \omega \vDash \phi\})$ vyhovuje omezení $\bowtie k$.

Definice 14. Nechť ϕ je LTL formule, s stav MDP. Pak formule $P_{\bowtie k}(s, \phi)$ je splněna, jestliže $Pr(\{\omega \mid \omega \in Path_s^\sigma, \omega \vDash \phi\})$ vyhovuje omezení $\bowtie k$ pro všechny strategie $\sigma \in Adv$.

2.4 Pravděpodobnostní ověřování modelu

Metody pro pravděpodobnostní ověřování modelu často hledají kompromis mezi přesností a škálovatelností. V současné době existují dva hlavní přístupy – *numerická analýza* založená na přesném řešení rovnic a *statistická analýza* založená na statistickém testování hypotéz na vybraných vzorcích. Numerická analýza je oproti statistickým metodám přesnější a je lepší ji použít v případě, kdy požadujeme přesné výsledky. Její nevýhoda je ale veliká paměťová náročnost a mnoho reálných modelů je prakticky neupočítatelných. Naproti tomu statistická analýza je méně paměťově náročná, a je tedy jediným východiskem pro složitější systémy. Statistická analýza není už z principu tak přesná jako numerická analýza, protože je založena na statistických informacích, které z daného vzorku získáme. Více o statistické a numerické analýze lze nalézt v [12, 25].

2.4.1 Statistická analýza

Základem statistické analýzy je generování vzorků cest tak dlouho, dokud se neobdrží dostatek informací pro statisticky podložené tvrzení, že systém (ne)splňuje danou formuli. Rozhodnutí o ukončení algoritmu bývá realizováno pomocí míry chyby α, β typu I. a II. Algoritmus je pak ukončen, je-li pravděpodobnost „*false negative*“ $\leq \alpha$ a pravděpodobnost „*false positive*“ $\leq \beta$. Vybírání a uchovávání informací o vzorcích není paměťově náročné a velmi dobře škáluje pro velké systémy. Složitost statistických metod závisí na přesnosti, s jakou chceme měření provádět.

Jedním z problémů statistické analýzy pro MDP je rozřešení nedetermismu během vzorkování, neboť požadujeme, aby formule platila při všech možných strategiích. V [12] je podán přesný popis algoritmu pro ověřování modelu pro MDP. Funguje na principu *Monte Carlo* simulace a hledání strategie, kterou lze použít jako protipříklad k vyvrácení formule. V každé iteraci algoritmu dochází k vylepšení předchozí strategie a následně k spuštění statistických algoritmů pro DTMC.

2.4.2 Numerická analýza

PCTL ověřování modelu pro DTMC. Numerické ověření splnitelnosti PCTL formule je postaveno na podobném principu jako CTL ověřování modelu. Pro formule, které nejsou pravděpodobnostně ohraničeny, jsou použity obdoby standardních algoritmů využívaných pro CTL.

Je-li formule tvaru $P_{\bowtie k}(\mathrm{X}\,\phi)$, pak jsou napočítány stavy splňující ϕ – označme je jako S^{ano}. Pravděpodobnost $\mathscr{P}(s, \mathrm{X}\,\phi)$, že ve stavu s je splněna formule $\mathrm{X}\,\phi$, je dána jako $\sum_{s' \in S^{\mathrm{ano}}} P(s, s')$. Pro formuli tvaru $P_{\bowtie k}(\phi\,\mathrm{U}\,\psi)$ jsou nejprve standardně předpočítány stavy, v nichž formule $\phi\,\mathrm{U}\,\psi$ určitě platí nebo určitě neplatí. Označíme tyto množiny stavů jako S^{ano} a S^{ne}. Následně je pomocí množin S^{ano} a S^{ne} počítána pravděpodobnost, s jakou bude formule platit v ostatních stavech. Níže uvedená rovnice zachycuje induktivní vztah pro pravděpodobnost splnitelnosti formule $\phi\,\mathrm{U}\,\psi$ ve stavu s:

$$\mathscr{P}(s, \phi\,\mathrm{U}\,\psi) = \begin{cases} 0, & \text{jestliže } s \in S^{\mathrm{ne}}, \\ 1, & \text{jestliže } s \in S^{\mathrm{ano}}, \\ \sum\limits_{s' \in S} P(s, s') \cdot \mathscr{P}(s', \phi\,\mathrm{U}\,\psi), & \text{jestliže } s \in S \setminus (S^{\mathrm{ne}} \cup S^{\mathrm{ano}}). \end{cases}$$

Pro ověřovaný stav s je pak vypočtená pravděpodobnost $\mathscr{P}(s, \phi\,\mathrm{U}\,\psi)$ porovnána s $\bowtie k$. Složitost výpočtu je lineární vůči velikosti ověřované formule a polynomiální vzhledem k počtu stavů [11].

PCTL ověřování modelu pro MDP. Podobně jako při statistickém ověřování modelu, je i v numerickém přístupu třeba vypořádat se s nedeterminismem. Protože pravděpodobnost splnitelnosti závisí na výběru konkrétní strategie, budeme rozlišovat *minimální* a *maximální* pravděpodobnost splnitelnosti (která počítá s „nejhorší“ a „nejlepší“ strategií). Kterou z těchto pravděpodobnostní budeme počítat, závisí na tom, leží-li $\bowtie$ v $\{<, \leq\}$ nebo v $\{>, \geq\}$. V prvním případě nás bude zajímat maximální pravděpodobnost, v druhém pak minimální. Samotnému algoritmu opět předchází výpočet množin S^{ano} a S^{ne}.

Uvedeme rovnice pro výpočet minimální pravděpodobnosti. Obdobně by byly sestaveny rovnice pro maximální případ[1].

$$\mathscr{P}(s, \mathrm{X}\,\phi) = \min_{a \in A(s)} \left\{ \sum_{s' \in S^{\mathrm{ano}}} Steps(s,a)(s') \mid Steps(s,a) = \mu \right\}$$

Pro $\phi = \phi_1 \,\mathrm{U}\, \phi_2$ má rovnice následující tvar:

$$\mathscr{P}(s,\phi) = \begin{cases} 0, & \text{jestliže } s \in S^{\mathrm{ne}}, \\ 1, & \text{jestliže } s \in S^{\mathrm{ano}}, \\ \min\limits_{a \in A(s)} \left\{ \sum\limits_{s' \in S} Steps(s,a)(s') \cdot \mathscr{P}(s',\phi) \right\}, & \text{jestliže } s \in S \setminus (S^{\mathrm{ne}} \cup S^{\mathrm{ano}}). \end{cases}$$

Podrobnější popis numerické analýzy pro DTMC a MDP lze nalézt v [19].

Ověřování modelu pro pravděpodobnostní LTL. Algoritmy pro ověřování modelu PLTL formule $P_{\bowtie k}(s,\phi)$ fungují na stejném principu jako standardní algoritmy pro nepravděpodobnostní LTL. Pro formuli $\neg\phi$ je nejprve vytvořen ekvivalentní automat nad nekonečnými slovy (většinou se používá Rabinův nebo Büchiho automat) a je vytvořen produkt ověřovaného modelu a automatu. Dále jsou identifikovány *koncové komponenty produktu*, což jsou silně souvislé komponenty, ze kterých nevede cesta do žádného stavu mimo ně. Pravděpodobnost dosažení koncových komponent obsahujících akceptující stav je pak porovnána s $\bowtie k$. Složitost je dvojitě exponenciální vzhledem k velikosti formule a polynomiální vzhledem k počtu stavů [5].

Příklad 15. Uvažujme MDP 2.2 uvedený výše a specifikaci pomocí PLTL formule $\phi = P_{>\frac{1}{2}}(\textit{padne hlava})$.

Budeme-li chtít zjistit, zda systém je modelem PLTL formule ϕ, budeme muset zkoumat, zda všechny strategie generují množinu cest vyhovující této formuli. Při volbě normální mince ale padne hlava s pravděpodobností $\frac{1}{2}$, a tedy systém nesplňuje specifikovanou vlastnost.

Mohlo by se zdát, že nedeterminismu se můžeme vyvarovat, budeme-li předpokládat, že uživatel vybere podvrženou i normální minci se stejnou pravděpodobností ($\frac{1}{2}$). Pak můžeme přesně určit pravděpodobnost, s jakou padne hlava (stane se tak s pravděpodobností $\frac{7}{12}$). Systém tedy splňuje specifikovanou vlastnost, a liší se tedy zásadně ve vlastnostech oproti uvedenému MDP.

1. Složitost numerické analýzy spočívá právě v řešení těchto lineárnách rovnic. V praxi používáme aproximativní metody k určení řešení.

Kapitola 3

Časové modely

Vedle nedeterminismu a pravděpodobnostních voleb je často na systém kladen požadavek změny stavu s určitým časovým omezením. U aplikací, které využívají například beztrátové síťování nebo bezpečnostní protokoly, je nezbytné zaměřit se při verifikaci i na časové vlastnosti. Abychom se mohli vyjádřit o vlastnostech systému v závislosti na čase, zavedeme formalismus časových automatů [2]. V literatuře lze objevit rozličné množství definic časového automatu, které byly motivovány snahou zlepšit schopnost analýzy jistých vlastností systémů s reálným časem (jako je například paměťová spotřeba systémů, plánování a rozvrhování) nebo potřebou zvýšit vyjadřovací sílu. Přehled variant časových automatů lze nalézt v [24]. Příkladem rozšíření s vyšší vyjadřovací sílou jsou například pravděpodobnostní časové automaty popsané v další kapitole.

3.1 Časový automat

Časový automat je konečně stavový, nedeterministický automat rozšířený o množinu hodin, které nabývají hodnot daných časovou doménou. Čas plyne stejně rychle na všech hodinách. K usměrňování chodu automatu je s každou hranou spjata množina stráží, které kladou omezení na hodnoty hodin.

Jako časovou doménu $\mathbb{T}$ budeme v případě časových automatů uvažovat buď nezáporná reálná čísla $\mathbb{R}_0^+$ nebo přirozená čísla $\mathbb{N}$ (uvažujeme včetně nuly). Hovoříme pak o *spojitém* či *diskrétním* čase.

Definice 16. Označený přechodový systém je čtveřice $(S, s_0, Act, \rightarrow)$, kde

- S je množina stavů,
- s_0 je počáteční stav,
- Act je množina akcí,
- $\rightarrow \subseteq S \times Act \times S$ je přechodová relace.

Pokud $(s, a, s') \in \rightarrow$, pak píšeme $s \xrightarrow{a} s'$.

Poznámka 17. Poznamenejme, že neklademe žádné omezení na kardinalitu množiny stavů. Narozdíl od konečných automatů nemusí být množina stavů ani konečná, ani spočetná.

Nechť $\mathbb{T}$ je časová doména, $\mathcal{X}$ konečná množina proměnných (*hodin*). Funkce $v : \mathcal{X} \rightarrow \mathbb{T}$ představuje valuaci hodin. Definujeme $v + t$ pro všechna $t \in \mathbb{T}$ a $x \in \mathcal{X}$ jako $(v + t)(x) = v(x) + t$. Obdobně uvažujeme $v - t$. Po resetování všech hodin z množiny $X \subseteq \mathcal{X}$ při valuaci v obdržíme valuaci $v[X := 0]$ definovanou následovně:

$$v[X := 0](x) = \begin{cases} 0, & \text{jestliže } x \in X, \\ v(x), & \text{jinak.} \end{cases}$$

Pro libovolnou množinu proměnných Q nad $\mathbb{T}$ značíme $\mathbb{T}^Q$ jako množinu všech valuací všech prvků z Q.

3.1.1 Hodinová omezení

Na množině $\mathcal{X}$ zavedeme *hodinová omezení*, která budou jistým způsobem vymezovat možné valuace hodin.

Definice 18. [20]Nechť $x, y \in \mathcal{X}, c \in \mathbb{N}, \sim \in \{<, \leq, >, \geq, =\}$, pak hodinová omezení ξ jsou dána následující syntaxí

$$\xi := x \sim c \mid x + c \sim y \mid \neg\xi \mid \xi \wedge \xi.$$

Množinu všech hodinových omezení nad hodinami $\mathcal{X}$ značíme jako $\mathcal{B}(\mathcal{X})$. Největší konstantu objevující se v hodinovém omezení ξ značíme $c_{max}(\xi)$. Valuace hodin v splňuje omezení ξ ($v \vDash \xi$), jestliže substitucí hodnotou $v(x)$ za každou hodinovou proměnnou x vyskytující se ve formuli ξ, obdržíme pravdivý výrok.
Syntaktické zkratky, které reprezentují hodinová omezení, které splňuje každá valuace ($True$), či nesplňuje žádná valuace ($False$), jsme schopni vyrobit běžným způsobem.

Definice 19. Časový automat (TA z anglického *timed automata*) s časovou doménou $\mathbb{T}$ je pětice $\mathcal{A} = (L, l_0, \mathcal{X}, Act, E, Inv)$, kde

- L je konečná množina lokací,
- l_0 je počáteční lokace,
- $\mathcal{X}$ je konečná množina hodin nad $\mathbb{T}$,
- Act je konečná množina akcí,
- $E \subseteq L \times \mathcal{B}(\mathcal{X}) \times Act \times 2^{\mathcal{X}} \times L$ je konečná množina hran,
- $inv : L \to \mathcal{B}(\mathcal{X})$ je funkce, která každé lokaci přiřazuje hodinové omezení, $inv(l)$ nazýváme invariantem lokace l.

Stavy automaty chápeme jako dvojice (l, v), kde $l \in L$, v je valuace hodin taková, že $v \vDash inv(l)$. Pokud se dále v textu budeme odkazovat na stav časového automatu s_i, kde i je index z $\mathbb{N}$, budeme automaticky předpokládat, že $s_i = (l_i, v_i)$.

Při spuštění automatu jsou všechny hodiny nastaveny na 0 (tj. $\forall x \in \mathcal{X} : v(x) = 0$), počáteční valuaci budeme označovat jako v_0. Během běhu automatu může být při každém přechodu resetována množina $X \in 2^{\mathcal{X}}$.

Stav (l, v) může buď, pokud to dovoluje invariant lokace, provést časový přechod, tj. setrvat ve stávající lokaci a nechat plynout čas, nebo učinit diskrétní přechod. Při diskrétním přechodu provede automat nějakou akci a, přejde do stavu (l', v') a vyresetuje množinu hodin $X \subseteq \mathcal{X}$. Takto může učinit, jestliže $(l, g, a, X, l') \in E$ a $v \vDash g$. Zónu g nazýváme stráží (anglicky *guard*) lokace l.

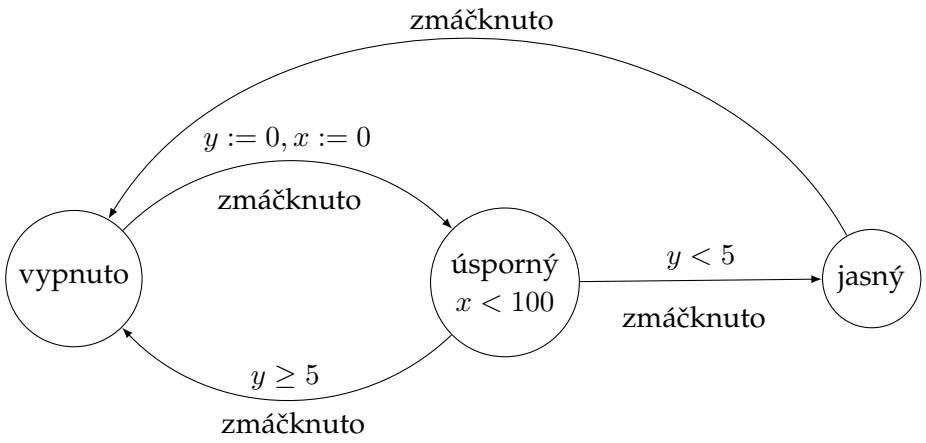
Definice 20. Sémantika časového automatu $\mathcal{A} = (L, l_0, \mathcal{X}, Act, E, \mathcal{L})$ s časovou doménou $\mathbb{T}$ je označený přechodový systém $[\![\mathcal{A}]\!]_{\mathbb{T}} = (S, s_0, Act \cup \mathbb{T}, \rightarrow)$, kde

- $S \subseteq L \times \mathbb{T}^{\mathcal{X}}$ je množina stavů systému,
- $s_0 = (l_0, v_0)$ je počáteční stav tvořený počáteční lokací a inicální valuací hodin v_0,
- $\rightarrow \subseteq S \times (\mathbb{T} \cup Act) \times S$ je přechodová relace splňující následující
  - **časový přechod:** $t \in \mathbb{T}, (l, v) \xrightarrow{t} (l, v + t)$, jestliže $\forall t' \in [0, t] : v + t' \vDash inv(l)$,
  - **diskrétní přechod:** $a \in Act, (l, v) \xrightarrow{a} (l', v')$, jestliže existuje hrana $(l, g, a, X, l') \in E$ taková, že $v \vDash g, v' \vDash inv(l')$ a $v' = v[X := 0]$.

Stráž lokace. Stráže lokací *usměrňují* běh systému tím, že povolují v jednotlivých časových okamžicích jen některé přechody.

Invariant lokace. K zajištění *živosti* systému je v časovém automatu zaveden invariant lokace, který (mimo jiné) zabezpečuje, aby hodnoty hodin nepřesáhly určitou mez. Použití invariantů může vyústit k uváznutí v momentě, kdy valuace hodin dosáhne horní hranice invariantu, ale již není přítomna žádná odchozí hrana, a tedy systém nemůže učinit ani diskrétní, ani časový přechod. Většina definic uvažuje pouze časové automaty, kde nemůže dojít k uváznutí, nebo invarianty zakazuje a k zaručení živosti používá jiné metody – například využívá Büchiho nebo Mullerovy akceptační podmínky [1].

Obrázek 3.1: Ukázka časového automatu [6].



Příklad 21. Na obrázku 3.1 je jednoduchý příklad časového automatu modelujícího inteligentní lampu. Lampa může být ve třech režimech: *zhasnuto*, *úsporný* a *jasný*. Když je provedeno stisknutí talčítka (akce *zmáčknuto*), automat přejde do nové lokace, která závisí na stavu, ve kterém se lampa nacházela v době stisknutí vypínače. Pomocí stráží jsme schopni vyjádřit závislost přechodu s vazbou na rychlost, se kterou uživatel stiskne vypínač. Ze stavu *vypnuto* přejde lampa pod dvěma rychlými stisky do stavu *jasný*, pod dvěma

1. Původně byla v definici [2] časového automatu zaručena živost právě takto.

pomalými stisky přejde opět do stavu *vypnuto*. V režimu *úsporný* navíc lampa nesmí zůstat déle než 100 časových jednotek. To zaručuje invariant lokace.

Výrazy na hranách představují stráže lokací a hodiny, které se mají po provedení přechodu resetovat ($x := 0$ značí reset hodin x).

Cesta časového automatu. Cesta časového automatu $\mathcal{A}$ je ne nutně konečná posloupnost

$$\pi = (l_0, v_0)\alpha_1(l_1, v_1)\alpha_2...(l_i, v_i)\alpha_{i+1}....,$$

kde α_j je buď akce automatu ($\alpha_j \in Act$), nebo časová prodleva ($\alpha_j \in \mathbb{T}$) a $\forall i \geq 1$ existuje v $[\![\mathcal{A}]\!]$ přechod $(l_{i-1}, v_{i-1}) \xrightarrow{\alpha_i} (l_i, v_i)$. Alternativně opět mluvíme o nekonečných cestách jako o bězích.
Zavedeme značení:

- $\pi(i) = (l_i, v_i)$
- pro $t \in \mathbb{T}$ a $\pi(i) = (l_i, v_i)$ definujeme $\pi(i) + t = (l_i, v_i + t)$
- π^i je sufix cesty začínající stavem $\pi(i)$
- $Path_s$ značí množinu všech běhů ze stavu s, $Path_s^{fin}$ pak množinu všech konečných cest
- $\mathscr{D}_\pi(i) \in \mathbb{T}$ udává čas, ve kterém se automat dostal do stavu $\pi(i)$. Tj.

$$\mathscr{D}_\pi(i) = \sum\{\alpha_k \mid 1 \leq k \ \leq i, \alpha_k \in \mathbb{T}\}.$$

Zenónovy běhy. Jako Zenónovy běhy (používá se též značení časově konvergentní běhy) označujeme běhy, ve kterých systém v konečném čase navštíví nekonečně mnoho stavů. Běh ω je Zenónovým během, jestliže existuje $t \in \mathbb{R}_0^+$ takové, že pro každé $j \in \mathbb{N}$ platí $\mathscr{D}_\omega(j) \leq t$. Zenónovy běhy neodpovídají chování reálných systémů, a proto je nebudeme při verifikaci uvažovat.

Diskrétní versus spojitý čas. Volba časové domény silně ovlivňuje způsob, jakým probíhá ověřování modelu. V případě volby diskrétního času (hodiny nabývají hodnot přirozených čísel) probíhá verifikace běžným způsobem. Čas v systému lze modelovat explicitně pomocí stavů, což vede k exponenciálnímu nárůstu systému[2]. U spojitých časových automatů můžou hodiny nabývat hodnot všech nezáporných reálných čísel, což zapříčiňuje nespočetně velký stavový systém. Čas je pak modelován implicitně a je třeba složitějších verifikačních technik. Složitost algoritmů je však u spojitých i diskrétních modelů stejná.

Nadále v práci budeme předpokládat spojitou časovou doménu $\mathbb{R}_0^+$ a diskrétními systémy se nebudeme zabývat.

3.1.2 Automaty s jiným tvarem hodinových omezení.

V literatuře se často liší způsob, jakým jsou definována hodinová omezení, která můžeme použít pro invariant nebo stráž lokace. V práci jsme se drželi definice, kterou používá nástroj pro ověřování modelu PRISM [18].

2. Verifikaci usnadňuje, že se stavy není třeba manipulovat explicitně.

Příkladem jsou například časové automaty, které jako invariant či stráž lokace povolují pouze omezení daná konjunkcí výrazů tvaru $x \sim c$, $x + c \sim y$ (nemáme k dispozici disjunkci a negaci).

Při verifikaci je někdy třeba uvažovat pouze automaty, které neobsahují hodinová omezení tvaru $x + c \sim y$ (tedy neporovnávají mezi sebou dvoje hodiny) – označujeme je jako *automaty bez diagonálních omezení*.

Poznamenejme, že oba výše uvedené příklady dávají stejně expresivní nástroj pro definici časových automatů jako hodinová omezení uvažovaná v této práci. Obecný automat lze převést na automat bez diagonálních omezení. Při převodu však dochází k exponenciálnímu nárůstu velikosti modelu (vzhledem k počtu hodin). Konstrukci lze nalézt v [2]. Podobně disjunkce v hodinových omezeních se dá „vyrobit“ znásobením stavů.

3.2 UPPAAL a časové automaty

UPPAAL [6] je automatizovaný nástroj podporující ověřování modelu časových automatů. Syntaxi a sémantiku časového automatu definuje UPPAAL podobně jako výše až na hodinová omezení, která dovolujeme pouze jako konjunkci výrazů tvaru $x \sim c$, $x + c \sim y$. Navíc je množina akcí rozdělena na *vstupní*, *výstupní* a *vnitřní* τ akce. Typ akce nemá vliv na sémantiku časového automatu, až při kompozici více automatů budou použity vstupní a komplementární výstupní akce k synchronizaci. Více v další kapitole.

3.2.1 Síť časových automatů

UPPAAL umožňuje paralelní kompozicí časových automatů pomocí CCS paralelního operátoru zachytit spolupráci více procesů – paralelní kompozice více automatů je označována jako *síť časových automatů*. Synchronizace procesů probíhá pomocí komplementárních vstupních a výstupních akcí. Předpokládáme, že množina akcí automatu je tvořena *vstupními* akcemi, které značíme jako $a?$, a *výstupními* akcemi $a!$ (pro libovolné a). Vnitřní akce systému označujeme jako τ akce. Síť navíc může být rozšířena o množinu proměnných, které jsou používány stejně jako v programovacích jazycích. Stav automatu je pak tvořen lokací, valuací hodin a hodnotou těchto proměnných.

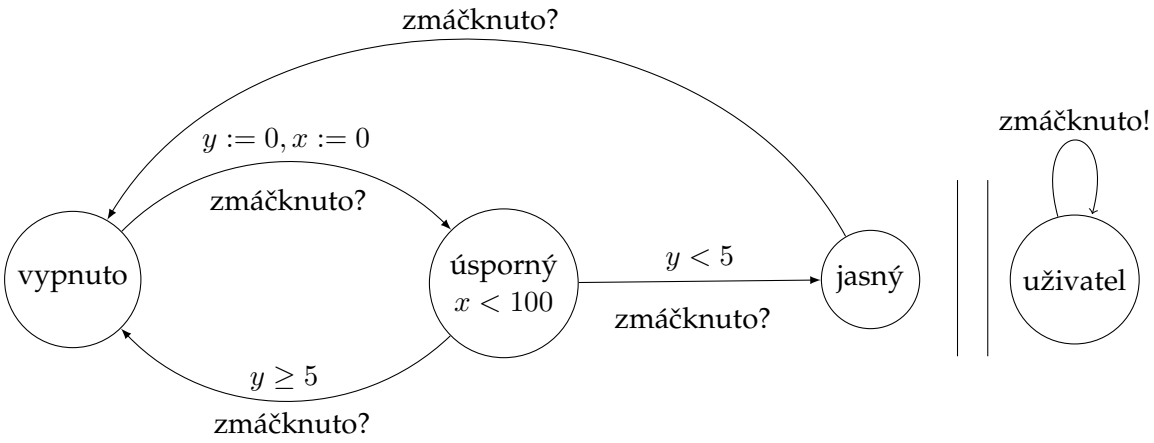
[6] Stavy sítě složené z n komponent tvaru $\mathcal{A}_i = (L_i, l_i^0, \mathcal{X}, Act, E_i, Inv_i)$ budeme chápat jako vektory, jehož složky tvoří jednotlivé lokace komponent. Lokace sítě je tedy vektor $\bar{l} = (l_1, \dots l_n)$. S každou lokací sítě je spjat invariant lokace, který definujeme pro lokaci $\bar{l} = (l_1, \dots l_n)$ jako $inv(\bar{l}) = \bigwedge_{1 \leq i \leq n}(inv_i(l_i))$. Nahrazení i-té složky l_i z $\bar{l}$ za l_i' zapisujeme jako $\bar{l}[l_i' \backslash l_i]$.

V síti časových automatů existují tři typy přechodů. Stejně jako je tomu u jednotlivých komponent, může i v síti pouze *plynout čas* bez změny lokace. Může být provedena *vnitřní akce* τ některé z komponent, která neovlivní chování ostatních komponent. Nebo může dojít k *synchronizaci* automatů pomocí komplementárních akcí. Poznamenejme, že vždy, když jedna z komponent provede vnější (nebo vnitřní) akci, musí další komponenta v systému provést odpovídající vnitřní (vnější) akci.

Definice 22. Pro síť tvořenou n časovými automaty tvaru $\mathcal{A}_i = (L_i, l_i^0, \mathcal{X}, Act, E_i, Inv_i)$, , kde množiny hodin a akcí jednotlivých komponent jsou po dvou disjunktní, je sémantika definována jako přechodový systém $(S, s_0, Act \cup \mathbb{R}_0^+ \rightarrow)$, kde

- $S = (L_1 \times \cdots \times L_n) \times {\mathbb{R}_0^+}^{\mathcal{X}}$ je množina stavů,
- $s_0 = (\bar{l}_0, v_0)$ je počáteční stav,
- přechodová relace $\rightarrow \subseteq S \times (Act \cup \mathbb{R}_0^+) \times S$ splňuje následující:
  - **časový přechod:** $t \in \mathbb{R}_0^+, (\bar{l}, v) \xrightarrow{t} (\bar{l}, v + t)$ jestliže $\forall t' \in [0, t] \colon v + t' \vDash inv(\bar{l})$,
  - **vnitřní akce:** $a \in Act, a = \tau, (\bar{l}, v) \xrightarrow{\tau} (\bar{l}[l_i'\backslash l_i], v')$, jestliže existuje hrana $(l_i, g, \tau, X, l_i') \in E_i$ taková že, $v \vDash g$, $v' \vDash inv(\bar{l}[l_i'\backslash l_i])$ a $v' = v[X := 0]$,
  - **synchronizace:** $(\bar{l}, v) \xrightarrow{a} (\bar{l}[l_i'\backslash l_i, l_j'\backslash l_j], v')$, jestliže existují hrany $(l_i, g_i, a!, X_i, l_i') \in E_i, (l_j, g_j, a?, X_j, l_j') \in E_j$ takové, že $v \vDash g_i \wedge g_j$, $v' \vDash inv(\bar{l}[l_i'\backslash l_i, l_j'\backslash l_j])$ a $v' = v[X_i \cup X_j := 0]$.

Obrázek 3.2: Ukázka sítě časových automatů [6].



Příklad 23. Na obrázku 3.2 je ukázka sítě časových automatů. První z komponent je již představená inteligentní lampa, druhou pak uživatel. Pokaždé, když uživatel provede rozsvícení lampy, musí lampa zareagovat komplementární akcí.

3.3 Časové logiky

U vestavěných nebo kritických systémů je nezbytné verifikovat chování s ohledem na reálný čas. Pro formální specifikaci vlastností je tedy vhodné volit takové logiky, které se dokáží vyjádřit kromě základních temporálních vlastností i o časových omezeních. Typicky chceme umět formálně vyjádřit vlastnosti jako *„Paket bude vždy doručen do 5 s“* nebo *„Po odeslání zprávy přijde odpověď do 10 s“*. Existuje více logik, které umožňují specifikovat časové vlastnosti. V práci se zaměříme pouze na časové rozšíření logiky výpočetního stromu (TCTL z anglického *timed computation tree logic*). Definici jsme převzali z [7]. Příkladem dalších časových logik může být například metrická temporální logika (MTL), časová výroková temporální logika (TPTL z anglického *timed propositional temporal logic*) nebo MITL, více v [16], kde je uveden i přehled rozhodnutelnosti problémů pro tyto logiky nad časovými automaty. K specifikaci se často používá i LTL bez časového rozšíření (případně CTL).

3.3.1 Časová logika výpočetního stromu

Časová logika výpočetního stromu je stavově orientovaná, tedy platnost formulí ověřujeme nad stavy. Pro tento účel připojíme k časovému automatu s množinou lokací L značkovací funkci $\mathcal{L} : L \to 2^{AP}$, která přiřadí každé lokaci množinu atomických propozic z množiny AP. Pro stavy časového automatu pak uvažujeme značkovací funkci $\mathcal{L}'$ definovanou pro každý stav (l, v) jako $\mathcal{L}'(l, v) = \mathcal{L}(l)$.

Definice 24. Nechť AP je množina atomických propozic a $\mathcal{X}$ množina hodin. Pak syntaxe TCTL formule ϕ je dána následovně:

$$\begin{aligned} \phi &:= \quad p \mid \zeta \mid z.\phi \mid \neg\phi \mid \phi \vee \phi \mid E[\psi] \mid A[\psi], \\ \psi &:= \quad \phi \,\mathrm{U}\, \phi, \end{aligned}$$

kde $p \in AP, z \ \in \mathcal{Z}, \zeta \in \mathcal{B}(\mathcal{X} \cup \mathcal{Z})$. Množina hodin $\mathcal{Z}$ je množina *hodin formule* ϕ.

TCTL vznikla z CTL přidáním hodinového omezení ζ mezi atomy formule. Dále byl přidán operátor $z.\phi$. Pomocí tohoto operátoru můžeme zavést nové hodiny formule, jež budou počítat čas od nuly. Pro lepší představu můžeme na $z.\phi$ nahlížet jako na formuli ϕ „stráženou“ hodinami z, které umožní vyhodnocení formule ϕ v okamžiku svého resetování.

Povšimněme si, že TCTL formule neobsahují narozdíl od CTL operátor X. Důvodem je, že v případě spojitého času není jasné, co má být další stav systému – je nekonečně možných následníků stavu.

Pro představu o způsobu tvoření časových formulí uvedeme příklad. Vlastnost, že hodiny modelu x nepřekročí hodnotu 3 dříve než uplyne 8 časových jednotek, můžeme vyjádřit jako TCTL formuli $z.A[(x \leq 3)\,\mathrm{U}(z \ = 8)]$.

Definice 25. Nechť $\mathcal{A}$ je časový automat, ϕ, ψ jsou TCTL formule, ω běh TA. Modelem TCTL formule je dvojice $(s, \mathcal{E})$, kde s je stav $[\![\mathcal{A}]\!]$, $\mathcal{E} \in \mathbb{R}_0^{+\mathcal{Z}}$ je valuace hodin formule.

Sémantika TCTL je dána následovně:

- $(s, \mathcal{E}) \vDash p \overset{def}{\iff} p \in \mathcal{L}(s)$
- $(s, \mathcal{E}) \vDash \zeta \overset{def}{\iff} v \ \cup \mathcal{E} \vDash \zeta$
- $(s, \mathcal{E}) \vDash z.\phi \overset{def}{\iff} (s, \mathcal{E}[z := 0]) \vDash \phi$
- $(s, \mathcal{E}) \vDash \neg\phi \overset{def}{\iff} (s, \mathcal{E}) \nvDash \phi$
- $(s, \mathcal{E}) \vDash \phi \vee \psi \overset{def}{\iff} (s, \mathcal{E}) \vDash \phi \vee (s, \mathcal{E}) \vDash \psi$
- $(s, \mathcal{E}) \vDash E[\phi \,\mathrm{U}\, \psi] \overset{def}{\iff} \exists \omega \in Path_s.\, (\omega, \mathcal{E}) \vDash_\psi \phi \,\mathrm{U}\, \psi$
- $(s, \mathcal{E}) \vDash A[\phi \,\mathrm{U}\, \psi] \overset{def}{\iff} \forall \omega \in Path_s\, (\omega, \mathcal{E}) \vDash_\psi \phi \,\mathrm{U}\, \psi$
- $(\omega, \mathcal{E}) \vDash_\psi \phi \,\mathrm{U}\, \psi \overset{def}{\iff} \exists k \ \in \mathbb{N}, t \in [0, \mathscr{D}_\omega(k+1) - \mathscr{D}_\omega(k)]$:
  - (i) $(\omega(k) + t, \mathcal{E} + \mathscr{D}_\omega(k) + t) \vDash \psi$
  - (ii) $\forall t' \leq t.(\omega(k) + t', \mathcal{E} + \mathscr{D}_\omega(k) + t') \vDash \phi \vee \psi$

(iii) $\forall i < k. \forall t' \in [0, \mathscr{D}_\omega(j+1) - \mathscr{D}_\omega(j)]. (\omega(j) + t', \mathcal{E} + \mathscr{D}_\omega(j) + t') \vDash \phi \vee \psi$

Povšimněme si, jakým způsobem je definována sémantika formule cesty $\phi \,\mathrm{U}\, \psi$. Požadujeme, aby v nějakém stavu cesty platilo ψ a do té doby platilo $\phi \vee \psi$ (narozdíl od klasické CTL, kde požadujeme, aby dokud nebude platit ψ, platilo ϕ). Důvodem je opět spojitý čas v sytému v kombinaci s *otevřenými intervaly*. Uvažujme formuli $A[x \leq 5 \,\mathrm{U}\, x > 5]$. Pokud bychom převzali původní definici sémantiky klasické CTL, musel by existovat stav, kde platí $x > 5$ a do té doby platí $x \leq 5$. Pokud ale v nějakém stavu (l, v) platí $x > 5$, pak díky spojitému času existuje t' takové, že $(v - t')(x) > 5$, tudíž by byla celá formule $A[x \leq 5 \,\mathrm{U}\, x > 5]$ automaticky vyhodnocena jako nepravdivá (neboť $(l', v - t') \nvDash x \leq 5$ zvolíme-li dostatečně malé t').

Definice 26. Nechť $\mathcal{A}$ je TA, s_0 jeho počáteční stav, ϕ je TCTL formule. Pak řekneme, že $\mathcal{A}$ splňuje specifikaci ϕ ($\mathcal{A} \vDash \phi$) právě tehdy, když $(s_0, \mathcal{E}_0) \vDash \phi$[3].

3.4 Ověřování modelu pro spojité časové automaty

Většina metod používaných k ověřování modelu časových automatů je založena na přeložení automatu na konečně stavový systém, který zachovává jisté vlastnosti. V případě, že ověřujeme splnitelnost formule časové temporální logiky, je i formule převedena na odpovídající formuli bez časového rozšíření. Na vzniklý systém jsou pak aplikovány běžné algoritmy používané pro ověřování modelu.

V dalších odstavcích stručně předvedeme některé techniky abstrakce časových automatů. Bude diskutována možnost jejich použití při verifikaci LTL a TCTL vlasností. Více v přehledu [26].

Regionová abstrakce. První metoda je založena na budování tzv. *grafu regionů*, což je konečně stavový přechodový systém, který je silně bisimulačně ekvivalentní s původním časovým automatem. Bisimulaci uvažujeme jak vzhledem k diskrétním přechodům, tak vzhledem k časovým přechodům. Pro časový automat $\mathcal{A}$ budeme odpovídající graf regionů značit jako $RG(\mathcal{A})$[4].

Označme c jako největší konstantu objevující se v nějakém hodinovém omezení automatu $\mathcal{A}$. Dvě valuace v, v' jsou *hodinově ekvivalentní*, jestliže pro všechny hodiny x, y

1. souhlasí celá část $v(x)$ s celou částí $v'(x)$, nebo obě valuace hodin přesáhly hodnotu c,
2. je celá část rozdílu jejich dvou různých valuací ($v(x) - v'(y)$ a $v'(x) - v'(y)$) nepřevyšující hodnotu c stejná.

Třídy rozkladu označujeme jako *regiony*. Lze vypozorovat, že region sdružuje ty valuace, které v dané lokaci buď všechny splňují stráž, nebo ji nesplňují žádná.

Jednotlivé stavy $RG(\mathcal{A})$ jsou třídy rozkladu stavů $[\![\mathcal{A}]\!]$ podle ekvivalence hodin ($(l, v) \cong (l', v') \Leftrightarrow l = l', v$ je hodinově ekvivalentní s v') a přechody mezi stavy $RG(\mathcal{A})$ odpovídají přechodům původního $\mathcal{A}$. Příklad grafu regionů je na obrázku 3.3.

3. Připomeňme, že $\mathcal{E}_0$ značí valuaci hodin fomule, kde jsou všechny hodiny nastaveny na 0.
4. Zkratka RG pochází z anglického *region graph*.

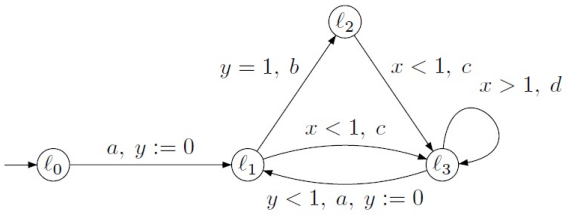
TCTL ověřování modelu. Verifikace probíhá tak, že TCTL formule ϕ je přeložena na ekvivalentní CTL formuli ϕ' a na $RG(\mathcal{A})$ jsou spuštěny standardní algoritmy pro ověřování modelu. Platí

$$RG(\mathcal{A}) \vDash \phi' \Leftrightarrow \mathcal{A} \vDash \phi.$$

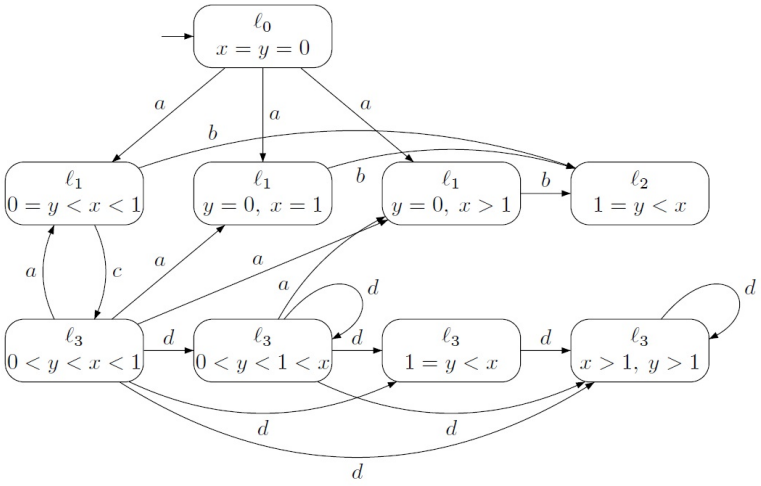
LTL ověřování modelu. Nejprve je vytvořen Büchi automat, který akceptuje právě ty běhy, které nesplňují LTL formuli. Následně je zkonstruován produkt grafu regionů s Büchi automatem a je rozhodnuto, zda akceptovaný jazyk je prázdný [14].

Verifikace pomocí této metody je obtížná, neboť počet regionů v $RG(\mathcal{A})$ spojený s každou lokací je exponenicální vzhledem ke konstantě c a počtu hodin v $\mathcal{A}$. V praxi se nepoužívá a slouží pouze ke stanovení rozhodnutelnosti a složitosti problémů verifikace. Další metoda, která bude představena je založena na principu regionové abstrakce, využívá však implicitní manipulaci se stavy, což vede k lepším výsledkům.

Obrázek 3.3: Ukázka grafu regionů [8]



(a) Časový automat



(b) Graf regionů

Digitalizace hodin. S každým grafovým regionem je spjata množina reprezentantů, kteří nabývají hodnot z předem dané konečné množiny – hodiny jsou diskretizovány a jejich hodnota se zvyšuje s pevně určeným časovým kvantem. I zde zřejmě dochází k exponenciálnímu nárůstu stavů (stavů je minimálně tolik, co regionů v regionové abstrakci) ale s tou výhodou, že lze aplikovat dobře známé datové struktury a algoritmy pro diskrétní systémy a

není třeba manipulovat se stavy vzniklého systému explicitně. Tato metoda je aplikovatelná pouze na automaty bez diagonálních omezení [26].

Další dva uvedené přístupy jsou založeny na nalezení hrubší abstrakce, než kterou udává regionová. Způsob abstrakce není znám předem a není závislý na syntaxi, proto tato metoda mnohem lépe škáluje než výše uvedené. Při hledání abstrakce bývá využíváno automatických prostředků.

Analýza využívající zónovou abstrakci. Systém je abstrahován pomocí tzv. *zón*, které můžeme chápat jako konvexní množinu valuací, jež je dána sjednocením některých regionů. Abstraktní stav je pak tvořen podobně, jako tomu bylo v případě grafu regionů, lokací a zónou.

Definice 27. Zóna α je množina valuací popsatelná hodinovým omezením ζ tvaru

$$\zeta = x \sim c \mid x + c \sim y \mid \zeta \wedge \zeta,$$

kde x, y jsou hodiny, c konstanta, $\sim \in \{<, \leq, =, \geq, >\}$. Valuace v splňuje omezení ζ ($v \models \zeta$), jestliže substitucí hodnotou $v(x)$ za každou hodinovou proměnnou x obdržíme pravdivý výrok.
Formálně je zóna množina

$$\alpha = \{v \in {\mathbb{R}_0^+}^{\mathcal{X}} \mid v \models \zeta\}.$$

Množinu všech zón nad hodinami $\mathcal{X}$ značíme jako $Zones(\mathcal{X})$. Dá se ukázat, že množinové operace (například průnik, rozdíl) jsou korektně definované operace nad zónami. Pro množinu hodin $X \subseteq \mathcal{X}$ a zónu $\zeta \in Zones(\mathcal{X})$ definujeme operaci

$$\zeta[X := 0] = \{v[X := 0] \mid v \models \zeta\}$$

a operaci diagonální projekce, která inutitivně řečeno rozšíří zónu v diagonálním směru do nekonečna

$$\nearrow \zeta = \{v \mid \exists t > 0.\ (v - t) \models \zeta\}.$$

Obdobně by se dala definovat operace zpětné diagonální projekce. Tyto operace jsou rovněž korektní operací nad zónami. Další používané operace spolu s důkazy lze nalézt v [13].

Zónovou abstrakci můžeme využít například při určování dosažitelnosti. Abstraktní stavy jsou pak napočítávány podél cest, které kladou na hodiny omezení formou invariantů a hodin. Rozlišujeme *dopřednou dosažitelnost*, kdy postupně napočítáváme dosažitelné stavy od iniciálního, a *zpětnou dosažitelnost*, kdy jdeme odzadu a zjišťujeme, jestli je daný stav dosažitelný z iniciálního.

Stejně jako v případě regionové abstrakce můžeme ověřovat splnitelnost formulí temporálních logik. Postupujeme obdobně jako v případě analýzy využívající regionovou abstrakci. Při verifikaci LTL formulí je třeba uvažovat pouze automaty bez diagonálních omezení.

Analýza využívající bisimulační ekvivalence. Poslední ze zmíněných metod bude využívat bisimulační ekvivalenci. Cílem této metody je nalézt hrubší ekvivalenci, která povede k menšímu počtu stavů než ta využívaná pro graf regionů – ideálně bisimulační ekvivalenci (uvažující časové i diskrétní přechody), která generuje nejmenší stavový prostor vzhledem ke korektním výsledkům.

Kapitola 4

Pravděpodobnostní časové modely

V této kapitole představíme modely kombinující časové i pravděpodobnostní vlastnosti. Formalismy, které rozšiřují časové automaty o pravděpodobnostní volby bývají souhrnně označovány jako *pravděpodobnostní časové automaty* (PTA z anglického *probabilistic timed automata*), ale v jednotlivých zdrojích se definice výrazně liší. Nejprve představíme pravděpodobnostní časové automaty, které používá nástroj pro pravděpodobnostní ověřování modelu PRISM [18]. Následně pak uvedeme stručný přehled dalších pravděpodonostních časových automatů popsaných v [3], [9], spolu se stručným porovnáním vzhledem k možnostem verifikace. Všechny představené modely budou pracovat se spojitým časem, tedy hodiny nabývají hodnot z $\mathbb{R}_0^+$.

Poznámka 28. Vedle pravděpodobnostních časových automatů existuje mnoho dalších formalismů, které se dají použít k modelování systémů s ohledem na čas a pravděpodobnost. V práci se jimi však nebudeme zabývat. Jako příklad můžeme uvést Markovovy řetězce ve spojitém čase [4], což jsou Markovovy řetězce obohaceny o čas, přičemž délka setrvání v jednotlivých stavech je dána exponenciálním rozdělením. Dále pak zobecněné semi-Markovské procesy [10] nebo zobecněné stochastické Petriho sítě [17]. Přehled některých modelů lze nalézt v [15].

4.1 PRISM a pravděpodobnostní časové automaty

Definice PTA tak, jak ji zavádí nástroj PRISM, se liší od výše uvedené definice časového automatu pouze přechodovou funkcí, která vybírá následující stav a množinu hodin k resetování s ohledem na pravděpodobnostní funkci. PRISM PTA je kombinací Markovových rozhodovacích procesů a klasických časových automatů představených výše. Časový automat z předchozí kapitoly je pouze speciální případ PTA, kdy je výběr následujícího stavu řízen pravděpodobnostní funkcí, která určí s pravděpodobností 1 jediný stav. Budeme používat některé pojmy, které jsme zavedli pro TA, aniž bychom je znovu definovali pro PTA, neboť jejich význam bude zřejmý. Rovněž budeme využívat některých pojmů uvedených v kapitole o Markovských rozhodovacích procesech.

Definice 29. [20] Pravděpodobnostní časový automat $\mathcal{A}$ je definován jako osmice $(L, l_0, \mathcal{X}, Act, inv, enab, prob, \mathcal{L})$, kde

- L je konečná množina lokací,
- $l_0 \in L$ je počáteční lokace,
- $\mathcal{X}$ je konečná množina hodin,
- Act je konečná množina akcí,

- $inv : L \to \mathcal{B}(\mathcal{X})$ je funkce, která každé proměnné přiřazuje omezení, $inv(l)$ nazýváme invariantem lokace l,
- $enab : L \times Act \to \mathcal{B}(\mathcal{X})$ je podmínka umožňující přechod, nazývaná stráž lokace,
- $prob : L \times Act \to (\mathcal{D}(2^{\mathcal{X}} \times L))$ je parciální pravděpodobnostní přechodová funkce,
- $\mathcal{L} : L \to 2^{AP}$ je značkovací funkce přiřazující každé lokaci množinu atomických propozic z množiny AP.

Formálně definujeme sémantiku PTA pomocí MDP. Automat se v každé lokaci rozhodne, jestli nechá plynout čas (obdobně jako je tomu u časového automatu), nebo jestli přejde do nové lokace. Přechod do nové lokace sestává z nedeterministické volby akce a následně výběru nové lokace a hodin k resetování s ohledem na pravděpodobnostní funkci. Poznamenejme, že stráž lokace je vždy spjatá s konkrétní pravděpodobnostní funkcí (tj. omezení určené stráží lokace je nezávislé na pravděpodobnostní volbě následujícího stavu).

Definice 30. Nechť $\mathcal{A} = (L, l_0, \mathcal{X}, Act, inv, enab, prob, \mathcal{L})$ je pravděpodobnostní časový automat. Pak sémantikou $\mathcal{A}$ je Markovův rozhodovací proces $[\![\mathcal{A}]\!] = (S, s_0, Act \cup \mathbb{R}_0^+, Steps)$, kde

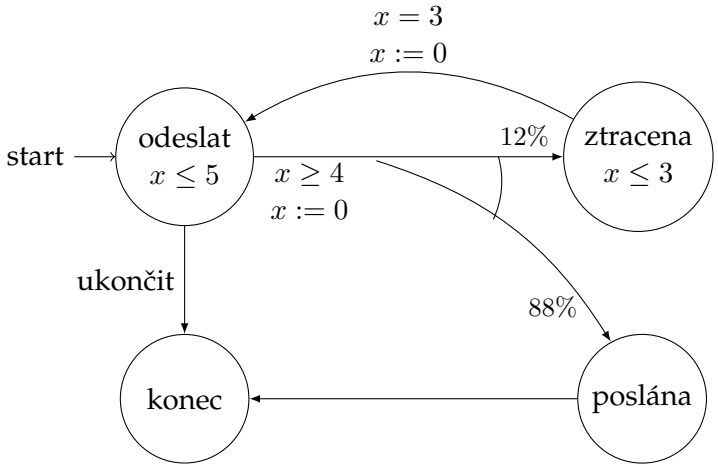
- $S \subseteq L \times {\mathbb{R}_0^+}^{\mathcal{X}}$, kde $(l, v) \in S \Leftrightarrow v \vDash inv(l)$,
- $s_0 = (l_0, v_0)$,
- $Steps((l, v), a) = \mu$ právě tehdy, když je splněna jedna z následujících podmínek:
  - **časový přechod:** $a = t \in \mathbb{R}_0^+$, $\mu = \mu_{(l, v\,+t)}$ tak, že $\forall t' \in [0, t] \colon v\ + t' \vDash inv(l)$,
  - **diskrétní přechod:** $a \in Act$, $prob(l, a) = p$ a pro každé $(l', v') \in S$ platí $v' \vDash enab(l, a) \Rightarrow$
$$\mu(l', v') = \sum_{X \subseteq \mathcal{X} \wedge v' = v[X:=0]} p(X, l').$$
  Jinak je $Steps((l, v), a)$ nedefinováno.

Příklad 31. Na obrázku 4.1 je znázorněn proces zasílání zprávy. Odeslání každé zprávy je učiněno během 5. časové jednotky, kterou systém čeká ve stavu *odeslat*. Zpráva je posílána po nespolehlivém kanálu a s 12% dojde k její ztrátě. V takovém případě počká systém přesně 3 jednotky času a přejde znovu do stavu *odeslat*, v tomto stavu je rovněž možno celý proces ukončit.

Cesta PTA $\mathcal{A}$ je cesta (konečná či nekonečná) $\pi = s_0(a_0, \mu_0)s_1(a_1, \mu_1)\ldots$ odpovídajícího $[\![\mathcal{A}]\!]$. Stejně uvažujeme množiny $Path_s^{fin}$ a $Path_s$. Budeme-li mluvit o strategii PTA, pak máme na mysli strategii pro $[\![\mathcal{A}]\!]$.

Zenónovy běhy. V kapitole o časových automatech jsme definovali Zenónovy běhy, které představují nežádoucí chování systémů (nežádoucí ve smyslu neodpovídající realitě) a při verifikaci jsme je ignorovali. Obdobně budeme postupovat v případě PTA (tedy uvažujeme pouze divergentní běhy).

Obrázek 4.1: Ukázka PTA modelujícího odesílání zpráv.



4.1.1 Paralelní kompozice

Často je užitečné definovat komplexní systémy pomocí paralelní kompozice jednotlivých komponent, které spolu interagují.

Definice 32. Paralelní kompozice $PTA_1 = (L_1, l_1^0, \mathcal{X}_1, Act_1, inv_1, enab_1, prob_1, \mathcal{L}_1)$ a $PTA_2 = (L_2, l_2^0, \mathcal{X}_2, Act_2, inv_2, enab_2, prob_2, \mathcal{L}_2)$ je pravděpodobnostní časový automat $PTA_1 \parallel PTA_2 = (L_1 \times L_2, (l_1^0, l_2^0), \mathcal{X}_1 \cup \mathcal{X}_2, Act_1 \cup Act_2, inv, enab, prob, \mathcal{L})$, kde

- $inv(l_1, l_2) = inv_1(l_1) \wedge inv_2(l_2)$ pro všechna $(l_1, l_2) \in L_1 \times L_2$,
- $$enab((l_1, l_2), a) = \begin{cases} enab_1(l_1, a) \wedge enab_2(l_2, a), & \text{jestliže } a \in Act_1 \cap Act_2, \\ enab_1(l_1, a), & \text{jestliže } a \in Act_1 \setminus Act_2, \\ enab_2(l_2, a), & \text{jestliže } a \in Act_2 \setminus Act_1, \end{cases}$$
- $prob((l_1, l_2), a) = p$ právě tehdy, když platí jedna z následujících podmínek:
  - **společná akce:** $a \in Act_1 \cap Act_2$, $prob(l_1, a) = p_1$, $prob(l_2, a) = p_2$ a pravděpodobnosntí funkce p je definována jako $p = p_1 \otimes p_2$,
  - **akce a** $\in \mathbf{Act_i} \setminus \mathbf{Act_j}$ pro $i = 1, j = 2$ nebo $i = 2, j = 1$, pak pro i takové, že $a \in Act_i$, je $prob_i(l_i, a) = p_i$ a $p = p_i \otimes \mu_{(\emptyset, l_j)}$,
- $\mathcal{L}$ je definována jako $\mathcal{L}(l_1, l_2) = \mathcal{L}_1(l_1) \cup \mathcal{L}_2(l_2)$.

4.2 Další varianty pravděpodobnostních časových automatů

První z variant pravděpodobnostních časových automatů, kterou představíme, je uvedena v článku [3]. Jedná se o mírně modifikovaný časový automat, ke kterému je navíc definována pravděpodobnostní sémantika. Formalismus představíme pouze na intuitivní úrovni.

S každým stavem s TA je spjata spojitá pravděpodobnostní funkce μ^s, na kterou jsou kladena jistá omezení. Příkladem funkcí, která pak tato omezení splňují, je například pravděpodobnostní funkce exponenciálního rozdělení pro otevřené intervaly, případně pravděpodobnostní funkce rovnoměrného rozdělení pro uzavřené.

Rovněž přiřadíme každému stavu pravděpodobnostní funkci nad hranami p^s, kde pro každou hranu e platí $p^s(e) > 0$ právě tehdy, když je povolena stráží lokace.

Je zaveden pojem *symbolické cesty*, což je intuitivně řečeno množina cest, během nichž automat prošel stejnou posloupnost hran. Symbolickou cestu, kdy automat vyšel ze stavu s a procházel hrany $e_1 \dots e_n$ značíme jako $\pi(s, e_1 \dots e_n)$.

Induktivně je definována pravděpodobnost symbolických cest jako

$$\mathbb{P}(\pi(s, e_1 \dots e_n)) = \int_{t \in I(s,e_1)} p_{s+t}(e_1)\, \mathbb{P}(\pi(s_t, e_2 \dots e_n)\, d\mu_s(t),$$

kde $I(s, e) = \{t \in \mathbb{R}_+ \mid$ ve stavu $s + t$ může být učiněn přechod pod hranou $e\}$,
$s_t = s'$ takový, že $(s + t)$ může přejít pod hranou e do s'.

Přirozeně pak můžeme podobně, jako tomu bylo v případě DTMC, zavést pro každou symbolickou cestu π cylindrickou množinu $cyl(\pi)$ s pravděpodobností $\mathbb{P}(cyl(\pi)) = \mathbb{P}(\pi)$

LTL ověřování modelu. V článku je popsána sémantika LTL nad konečnými cestami a je řešen kvalitativní problém, zda automat téměř jistě splňuje LTL formuli ϕ, tedy zda platí $\mathbb{P}(s_0 \vDash \phi) = 1$. Pomocí regionové konstrukce podobné té pro časové automaty[1] je ukázáno, že problém je rozhodnutelný pro automaty s jedněmi hodinami.

Také UPPAAL definuje pro časové automaty pravděpodobnostní sémantiku [9]. Navíc je přidána ke každé lokaci její *cena*. Hodiny narozdíl od předchozí definice 3.2 odpočítávají čas s rozdílnou rychlostí danou cenou lokace. Vyjadřovací síla rozšíření je shodná s lineárními hybridními automaty [1]. Podobně jako v [3] je s každým stavem spjata spojitá pravděpodobnostní funkce nad prodlevami a pravděpodobnostní funkce určující následující stav. V uvedeném článku se autoři omezují opět na rovnoměrné nebo exponenciální rozdělení, tvrdí však, že možné je libovolné (nejsou udány žádné omezující podmínky jako tomu bylo v předchozím případě).

Navíc je představen způsob, jakým lze jednotlivé časové automaty skládat pomocí paralelní kompozice. Síť časových automatů byla představena již v kapitole 3.2. Je k ní pouze přidána pravděpodobnostní sémantika. Chování systému je založeno na „závodu“ mezi komponentami. Komponenta, která „zvítězí“, vybere akci, pod kterou se má přejít do nové lokace a ostatní komponenty musí respektovat výběr této akce. Každá z komponent pak dokončí závod v nějakém čase daném její pravděpodobnostní funkcí. V nové lokaci se opět odehraje další závod. UPPAAL rovněž představuje pojem symbolické cesty a její pravděpodobnost je opět definována induktivně, jako integrál přes všechny možné „výherní“ časy.

Ověřování modelu pro síť časových automatů. UPPAAL provádí statistické ověřování modelu, jehož principy byly představeny pro pravděpodobnostní modely (MDP a DTMC). Jako logiku pro popis vlastností používá pravděpodobnostní (PWCTL z anglického *probabilistic weighted computation tree logic*).

4.3 Pravděpodobnostní časová logika výpočetního stromu

Pravděpodobnostní časová logika výpočetního stromu (PTCTL z anglického *probabilistic timed computation tree logic*) vznikla kombinací pravděpodobnostní a časové CTL. V různých

1. Narozdíl od TA, kdy byl konstruován konečně stavový přechodový systém, je nyní konstruován DTMC.

zdrojích lze nalézt různé definice. V práci jsme se drželi definice z [22].

Definice 33. Nechť AP je množina atomických propozic, $\mathcal{X}$ množina hodin modelu, $k \in [0,1]$, $\bowtie \in \{<, >, =, \leq, \geq\}$. Pak syntaxe PTCTL formule ϕ je dána následovně:

$$\begin{aligned} \phi &:= \quad p \mid \zeta \mid z.\phi \mid \neg\phi \mid \phi \vee \phi \mid P_{\bowtie k}[\psi], \\ \psi &:= \quad \phi \,\mathrm{U}\, \phi, \end{aligned}$$

kde $p \in AP, z \in \mathcal{Z}, \zeta \in \mathcal{B}(\mathcal{X} \cup \mathcal{Z})$. Množina hodin $\mathcal{Z}$ je množina *hodin formule* ϕ.

Definice 34. Nechť $\mathcal{M}$ je PTA, ϕ, ψ jsou PTCTL formule, ω běh PTA. Modelem PTCTL formule je dvojice $(s, \mathcal{E})$, kde $s = (l, v)$ je stav $[\![\mathcal{A}]\!]$, $\mathcal{E} \in \mathbb{R}_0^{+\mathcal{Z}}$ je valuace hodin formule.

Sémantika PTCTL je dána následovně:

- $(s, \mathcal{E}) \vDash p \overset{def}{\iff} p \in \mathcal{L}(s)$
- $(s, \mathcal{E}) \vDash \zeta \overset{def}{\iff} v \cup \mathcal{E} \vDash \zeta$
- $(s, \mathcal{E}) \vDash z.\phi \overset{def}{\iff} (s, \mathcal{E}[z := 0]) \vDash \phi$
- $(s, \mathcal{E}) \vDash \neg\phi \overset{def}{\iff} (s, \mathcal{E}) \nvDash \phi$
- $(s, \mathcal{E}) \vDash \phi \vee \psi \overset{def}{\iff} (s, \mathcal{E}) \vDash \phi \vee (s, \mathcal{E}) \vDash \psi$
- $(s, \mathcal{E}) \vDash P_{\bowtie k}[\phi \,\mathrm{U}\, \psi] \overset{def}{\iff} Pr^{\sigma}(\{\omega \in Path_s^{\sigma} \mid (\omega, \mathcal{E}) \vDash_{\psi} \phi \,\mathrm{U}\, \psi\}) \bowtie k$ pro všechny strategie σ
- $(\omega, \mathcal{E}) \vDash_{\psi} \phi \,\mathrm{U}\, \psi \Leftrightarrow \exists k \in \mathbb{N}, t \in [0, \mathscr{D}_{\pi}(k+1) - \mathscr{D}_{\pi}(k)]$:
  - (i) $(\pi(k) + t, \mathcal{E} + \mathscr{D}_{\pi}(k) + t) \vDash \psi$,
  - (ii) $\forall t' \leq t.(\pi(k) + t', \mathcal{E} + \mathscr{D}_{\pi}(k) + t') \vDash \phi \vee \psi$
  - (iii) $\forall i < k. \forall t' \in [0, \mathscr{D}_{\pi}(j+1) - \mathscr{D}_{\pi}(j)].(\pi(j) + t', \mathcal{E} + \mathscr{D}_{\pi}(j) + t') \vDash \phi \vee \psi$

Definice 35. Nechť $\mathcal{M}$ je PTA, s_0 jeho počáteční stav, ϕ je PTCTL formule. Pak řekneme, že $\mathcal{M}$ splňuje specifikaci ϕ ($\mathcal{M} \vDash \phi$) právě tehdy, když $(s_0, \mathcal{E}_0) \vDash \phi$[2].

4.4 Ověřování modelu PRISM PTA.

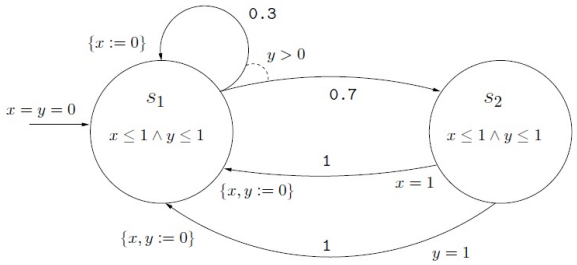
Metody, které používá PRISM pro ověřování modelu PTA vycházejí z metod, které jsme představili v kapitole o časových automatech, neboť obdobně dochází k nespočetně velkému nárůstu stavů.

Regionová abstrakce. Jedním z možných přístupů je regionová abstrakce, která zavádí na valuacích hodin stejnou ekvivalenci, jako je tomu v případě TA. Rozdílně je pak konstruován $RG(\mathcal{A})$, kdy zohledňujeme pravděpodobnostní volbu – nejedná se o přechodový systém, ale o Markovův rozhodovací proces. Popis konstrukce s důkazem lze nalézt v [21].

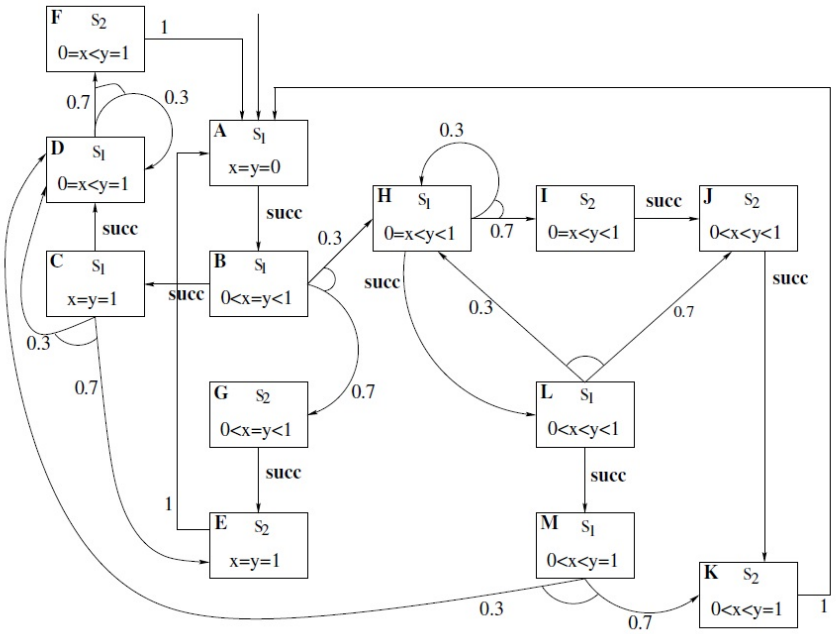
Příklad regionové abstrakce je na obrázku 4.2. Obdobně probíhá i samotné ověření modelu, kdy je PTCTL formule přeložena na ekvivalentní PCTL a je ověřena její splnitelnost oproti zkonstruovanému MDP.

2. Připomeňme, že $\mathcal{E}_0$ značí valuaci hodin fomule, kde jsou všechny hodiny nastaveny na 0.

Obrázek 4.2: Ukázka regionové abstrakce pro PTA [21]



(a) Pravděpodobnostní časový automat



(b) Odpovídající graf regionů

Analýza pomocí reprezentantů. Podobný princip jako v případě TA. Není však možno tuto techniku aplikovat na obecné PTCTL formule. Nejsou povoleny vnořené PTCTL formule a hodinová omezení vyskytující se v PTA musí být bez neostrých nerovností.

Zónová abstrakce. Také je možné použít analýzu pomocí zónové abstrakce, která je podobná té představené pro TA. Více například v [21].

Kapitola 5

Rozšíření pravděpodobnostních časových automatů

V práci jsme rozšířili PRISM PTA o pravděpodobnost změny lokace v jistých časových okamžicích. Naší motivací byla možnost vyjádřit, že v dané lokaci je při určité valuaci vyšší šance diskrétního přechodu než při jiné. Narozdíl od výše popsaných formalismů umožňujících pravděpodobností volbu prodlevy, neuvažujeme spojité pravděpodobnostní rozdělení, které by pro každou možnou prodlevu přesně určovalo její pravděpodobnost, což sice na jednu stranu přináší jistá omezení, na druhou stranu ale získáme větší volnost při definování pravděpodobnostní funkce. V novém formalismu využíváme diskrétní pravděpodobnostní funkce, proto jsme jej nazvali *diskrétní pravděpodobnostní časový automat* (zkráceně DPTA), která udává pravděpodobnost přechodu systému pro skupiny valuací. Snažili jsme se co nejvíce zachovat chování PRISM PTA, aby bylo možné využít stávajících technik pro verifikaci DPTA. V principu se chování DPTA od PTA liší pouze v rozhodnutí, jestli bude učiněn časový nebo diskrétní přechod. U PTA bylo nedeterministicky rozhodnuto mezi časovým a diskrétním přechodem a v prvním případě byla nedeterministicky vybrána prodleva určité délky. Volba mezi časovým a diskrétním přechodem je u DPTA řízena pravděpodobnostně. Zde je další rozdíl oproti stávajícím modelům využívajících pravděpodobnosti při určování prodlev, DPTA umožňuje určit pravděpodobnost opuštění lokace při jisté valuaci, nikoli explicitně určit pravděpodobnost délky prodlevy.

Uvažme studenta a časový automat, který zjednodušeně modeluje jeho denní režim. Automat obsahuje jediné hodiny x, které se každou noc o půlnoci vynulují a začínají znovu odpočítávat čas. Student může být pouze v jednom ze dvou stavů – může *bdít* nebo *snít*. U běžného časového automatu není způsob jak vyjádřit, že v určitou denní dobu je větší šance, že student odejde ze stavu *snít* nebo *bdít*. Pouze můžeme studentovi zakázat spánek či bdění v určité hodiny. Možné řešení je přidat k časovému automatu pravděpodobností funkci, která pro různé denní doby specifikuje pravděpodobnost, že student opustí daný stav. Tedy například ve stavu *bdít* je od 10:00 do 1:00 je 90% šance přechodu do druhého stavu, v jiné hodiny je tato šance 15%. Obdobně můžeme funkci definovat pro stav *snít*.

V příkladu se spícím studentem byla jediná časová míra denní doba, a proto jsme nahlíželi na čas jako na jedinou proměnnou. Pravděpodobnost přechodu pak byla specifikována po intervalech. Co ale když na scénu vstoupí další faktor ovlivňující pravděpodobnost změny stavu studenta s ohledem na čas? Přidejme ke stavům ještě stav *jídlo*, k hodinám systému přidejme hodiny y, které se vynulují vždy, když student navštíví stav *jídlo*. Čím déle nenavštíví student stav *jídlo*, tím vyšší je šance odchodu ze stavu *snění*. Nyní již pravděpodobnost opuštění stavu závisí na dvou proměnných x a y. Obecně pak můžeme mít hodin v systému až n.

Pravděpodobnost přechodu systému v závislosti na čase budeme obecně uvažovat pro jisté konvexní podmnožiny n-rozměrného časového prostoru (narozdíl od předchozího příkladu, kde jsme pravděpodobnost specifikovali pro jednorozměrné intervaly). Jako vhodná skupina podmnožin se nabízí zóny.

Rozklad časového prostoru. Pro konečnou množinu hodin $\mathcal{X}$ nad časovou doménou $\mathbb{R}_0^+$ je časový prostor $\mathcal{T}$ definován jako $\mathcal{T} \subseteq {\mathbb{R}_0^+}^{\mathcal{X}}$.

Definice 36. Nechť $\mathcal{T}$ je časový prostor, pak rozkladem $\mathcal{T}$ rozumíme systém $\langle\mathcal{T}\rangle$ konečně mnoha neprázdných, disjunktních množin valuací $\alpha_1, \alpha_2, \ldots, \alpha_n$ tak, že platí

$$\mathcal{T} = \bigcup_{\alpha \in [\mathcal{T}]} \alpha,$$

a každá množina $\alpha \in \langle\mathcal{T}\rangle$ je zónou (definice 27). Množiny α budeme označovat jako třídy.

Při rozkladu $\langle\mathcal{T}\rangle$ označíme $[v]$ pro libovolnou valuaci $v \in \mathcal{T}$ jako třídu z $\langle\mathcal{T}\rangle$, která obsahuje valuaci v.

Poznámka 37. Poznamenejme, že ne zdaleka pro každý časový prostor musí takový rozklad existovat. Pokud ale budeme uvažovat pouze časové prostory nad hodinami $\mathcal{X}$, které jsou popsatelné omezeními z $\mathcal{B}(\mathcal{X})$ (tedy prostory, pro které existuje ξ takové, že $\mathcal{T} = \{v \in {\mathbb{R}_0^+}^{\mathcal{X}} \mid v \vDash \xi\}$), pak takový rozklad vždy existuje – jako rozklad můžeme uvážit třídy tvořené regiony.

Třída následníka. Třída následníka pro valuaci v je intuitivně řečeno první třída lišící se od $[v]$, do které se můžeme dostat, přičteme-li k valuaci v jistý časový přírůstek.

Definice 38. Nechť $\mathcal{X}$ je množina hodin, $\mathcal{T} \subseteq {\mathbb{R}_0^+}^{\mathcal{X}}$ je časový prostor nad $\mathcal{X}$ a $\langle\mathcal{T}\rangle$ jeho rozklad. Relace $\rightsquigarrow$ mezi množinou $\mathcal{T}$ a $\langle\mathcal{T}\rangle$ je relací následníka, jestliže $\forall v \in \mathcal{T}, \alpha \in \langle\mathcal{T}\rangle$ takové, že $[v] \neq \alpha$ platí:

$$v \rightsquigarrow \alpha \Leftrightarrow \exists t \in \mathbb{R}_0^+ \colon v + t \in \alpha \wedge \forall t' < t \colon v + t' \in \alpha \cup [v].$$

Je-li $v \rightsquigarrow \alpha$, pak píšeme $\alpha = succ(v)$ a říkáme, že α je třída následníka pro v.

Povšimněme si, že pro některé valuace neexistuje třída následníka ($succ(v) = \bot$).

Nyní rozšíříme relaci následníka na relaci mezi třídou a množinou tříd. Pak $succ(\alpha)$ definujeme jako

$$succ(\alpha) = \{\beta \mid \exists v \in \alpha \colon \beta = succ(v)\}.$$

Jako $\langle\mathcal{T}\rangle^*$ budeme značit rozklad na časovém prostoru $\mathcal{T}$, kde pro všecha $\alpha \in \langle\mathcal{T}\rangle^*$ platí jedna z následujících podmínek:

- $|succ(\alpha)| = 1$ a zároveň pro všechny valuace $v \in \alpha$ je $succ(v)$ definováno,
- $|succ(\alpha)| = 0$.

Jestliže $\alpha \in \langle\mathcal{T}\rangle^*$ a $succ(\alpha) = \{\beta\}$, pak nutně β je následník všech valuací z α (zřejmě).

Pokud dále v textu budeme mluvit o rozkladu na časovém prostoru $\mathcal{T}$, budeme mít automaticky na mysli rozklad $\langle\mathcal{T}\rangle^*$ s výše uvedenými vlastnostmi.

Pravděpodobnostní funkce diskrétního přechodu. Na rozkladu časového prostoru definujeme *pravděpodobnostní funkci diskrétního přechodu*, která bude určovat pravděpodobnost setrvání v jednotlivých lokacích. Pro každou třídu z rozkladu vymezuje pravděpodobnostní funkce diskrétního přechodu pravděpodobnost, že automat učiní diskrétní přechod uvnitř stávající časové třídy. Aby byla funkce smysluplně definovaná, požadujeme, aby při kladné

pravděpodobnosti učinění přechodu bylo opravdu možné přechod učinit, a naopak, při kladné pravděpodobnosti časového přechodu bylo možné učinit časový přechod. V následující definici můžeme chápat množinu X právě jako množinu valuací, ve kterých je možný diskrétní přechod.

Definice 39. Nechť $\mathcal{X}$ je množina hodin, X je podmnožina valuací hodin $\mathbb{R}_0^{+\mathcal{X}}$, $\mathcal{T}$ časový prostor a $\langle\mathcal{T}\rangle$ jeho rozklad. Funkce $f_{\mathcal{T}}^{X} : \langle\mathcal{T}\rangle \to [0,1]$ je pravděpodobnostní funkce diskrétního přechodu, jestliže platí:

- Pokud $trans(l)(\alpha) > 0$, pak všechny hodnoty z α leží v X.
- Je-li $trans(l)(\alpha) \neq 1$, pak pro každou valuaci v zóně α existuje časový následník.

Syntaxe DPTA. Nyní rozšíříme klasický pravděpodobnostní časový automat o možnost učinit pravděpodobnostní volbu mezi diskrétním a časovým přechodem.

Definice 40. Diskrétní pravděpodobnostní časový automat $\mathcal{A}$ je

$$\mathcal{A} = (L, l_0, \mathcal{X}, Act, inv, enab, prob, part, trans, \mathcal{L}),$$

kde pro jednotlivé složky platí:

- L je konečná množina lokací,
- $l_0 \in L$ je počáteční lokace,
- $\mathcal{X}$ je konečná množina hodin s doménou $\mathbb{R}_0^+$,
- Act je konečná množina akcí,
- $inv : L \to \mathcal{B}(\mathcal{X})$ je funkce, která každé lokaci přiřazuje hodinové omezení,
- $enab : L \times Act \to \mathcal{B}(\mathcal{X})$ je umožňující podmínka přechodu zvaná stráž lokace,
- $prob : L \times Act \to (\mathcal{D}(2^{\mathcal{X}} \times L))$ je parciální přechodová funkce,
- $part$ je funkce přiřazující každé lokaci l rozklad $\langle\mathcal{T}\rangle^*$ na časovém prostoru $\mathcal{T}$, kde $\mathcal{T} = inv(l)$,
- $trans$ je funkce vracející pro každou lokaci l pravděpodobnostní funkci diskrétního přechodu
$$trans(l) = f_{inv(l)}^{\mathscr{I}(l)} : part(l) \to [0,1]$$
kde $\mathscr{I}(l) = \{v \in \mathbb{R}_0^{+\mathcal{X}} \mid \exists a \in Act\colon prob(l,a)$ je definováno a $v \vDash enab(l,a)\}$,
- $\mathcal{L} : L \to 2^{AP}$ je funkce přirazující každé lokaci množinu atomických propozic.

Pro každou lokaci $l \in L$ a valuaci hodin v značíme $[v]_l$ jako třídu z $part(l)$ obsahující valuaci v.

Sémantika DPTA. Sémantika DPTA bude definována pomocí Markovského procesu, který bude mít tři druhy stavů. První skupina stavů bude realizovat pravděpodobnostní volbu mezi časovým a diskrétním přechodem s ohledem na pravděpodobnostní funkci danou funkcí $trans$. Druhé skupině bude umožněno učinit časový přechod tak, aby výsledná valuace neopustila časovou třídu, a v případě, že časový přechod již nelze učinit, bude vynucen diskrétní přechod. Třetí skupina stavů budou stavy zajišťující časový přechod mezi třídami valuací.

Definice 41. Nechť $\mathcal{A} = (L, l_0, \mathcal{X}, Act, inv, enab, prob, part, trans, \mathcal{L})$ je DPTA. Pak sémantikou $\mathcal{A}$ je Markovův rozhodovací proces $\mathcal{M} = (S, s_0, Act \cup \mathbb{R}_0^+ \cup \{\varepsilon\}, Steps, \mathcal{L})$, kde

- $S \subseteq L \times {\mathbb{R}_0^+}^{\mathcal{X}} \times \{0, 1, 2_{Zones(\mathcal{X})}\}\}$, kde

$$(l, v, i) \in S \Leftrightarrow v \vDash inv(l)$$

  a pokud $i = 2_\alpha$, pak $\alpha \in part(l)$ a platí $\alpha = succ(v) \vee \alpha = [v]_l$,

- $s_0 = (l_0, v_0, 0)$.
- Přechodová funkce $Steps$ je definována podle druhu jednotlivých stavů:
  - **přechody stavů typu 0:**

    $Steps((l, v, 0), a) = \mu$ právě tehdy, když nastává jeden z následujících případů

    - *časový přechod:*

$$a = t \in \mathbb{R}_0^+, \mu = \mu_{(l, v+t, 0)} \wedge \forall t' \leq t \colon v + t' \in [v]_l$$ [1]

    - *pravděpodobnostní volba mezi časovým a diskrétním přechodem:*

$$a = \varepsilon, \mu(l, v, i) = \begin{cases} trans(l)([v]_l), & \text{jestliže } i = 1, \\ 1 - trans(l)([v]_l), & \text{jestliže } i = 2_{succ(v)}. \end{cases}$$

  - **přechody stavů typu 1:**

    $Steps((l, v, 1), a) = \mu$ právě tehdy, když jeden z následujících případů:

    - *časový přechod:*

$$a = t \in \mathbb{R}_0^+, \mu = \mu_{(l, v+t, 1)} \wedge \forall t' \leq t \colon v + t' \in [v]_l$$

    - *diskrétní přechod:*

$$a \in Act, prob(l, a) = p \text{ a pro každé } (l', v', 0) \in S \text{ platí}$$

$$v' \vDash inv(l'), v \vDash enab(l, a) \Rightarrow \mu(l', v', 0) = \sum_{X \subseteq \mathcal{X} \wedge v' = v[X:=0]} p(X, l')$$

1. Díky kovexnosti $[v]$ by postačovala podmínka $v + t \in [v]_l$.

– **přechody stavů typu 2:**

$Steps((l, v, 2_\alpha), a) = \mu$ právě tehdy, když jeden z následujících případů:

∗ *časový přechod:*

$$a = t \in \mathbb{R}_0^+, \mu = \mu_{(l,v+t,2_\alpha)}, v\ + t \in ([v]_l \cup \alpha),$$

∗ *diskrétní přechod:*

$$a = \varepsilon, \mu = \mu_{(l,v,0)}, v\ \in \alpha.$$

- Značkovací funkce $\mathcal{L}$ je definována jako

$$\mathcal{L}((l, v, i)) = \mathcal{L}(l).$$

Dále v textu budeme používat značení s_j^i, kde $i \in \mathbb{N}_0, j \in \{0, 1, 2_\alpha\}$, které značí stav

$$s_i^j = (l_i, v_i, j).$$

Běhy a strategie DPTA. Běhy DPTA jsou definovány stejně jako v případě skrze Markovské procesy. Podobně pak strategie.

PTCTL a DPTA. Splnitelnost PTCTL formulí pro DPTA se neliší od té definované pro PTA (modely PTCTL formule jsou stavy odpovídajícího MDP). Typ stavu (zda se jedná o stav ze skupiny 0, 1 nebo 2) nemá vliv na splnitelnost PTCTL formule.

Definice 42. Nechť $\mathcal{M}$ je DPTA, s_0 jeho počáteční stav, ϕ je PTCTL formule. Pak řekneme, že $\mathcal{M}$ splňuje specifikaci ϕ ($\mathcal{M} \vDash \phi$) právě tehdy, když $(s_0, \mathcal{E}_0) \vDash \phi$[2].

5.1 Porovnání DPTA a PTA vhledem k PTCTL ověřování modelu

Ukážeme, že problém PTCTL verifikace DPTA se dá převést na PTCTL verifikaci PTA a obráceně. Složitost převodu a případné zvětšení modelu po převodu bude diskutováno v další kapitole spolu s návrhem řešení problému.

Definice 43. Nechť $\mathcal{M} = (S, s_0, Act, Steps, \mathcal{L})$ a $\mathcal{M}' = (S', s_0', Act', Steps', \mathcal{L}')$ jsou časové Markovské procesy. Pak MDP jsou ekvivalentní vzhledem k PTCTL ($\mathcal{M} \cong_{PTCTL} \mathcal{M}'$) jestliže existuje úplná relace $\mathcal{R} : S \ \rightarrow S'$ taková, že $s_0 \mathcal{R}\, s_0'$ a pro libovolnou PTCTL formuli ϕ, valuaci hodin formule $\mathcal{E}$ a libovolné $s \in S, s' \in S'$ t.ž. $s \mathcal{R}\, s'$ platí

$$(s, \mathcal{E}) \vDash \phi \Leftrightarrow (s', \mathcal{E}) \vDash \phi.$$

Pokud $\mathcal{M} \cong_{PTCTL} \mathcal{M}'$, pak k ověření stavu $s \in S$ oproti PTCTL formuli ϕ stačí nalézt s' takové, že $s \mathcal{R}\, s'$ a ověřit s' oproti ϕ v $\mathcal{M}'$. Podobně v obráceném směru.

Definice 44. Bud'te $\mathcal{A}$ PTA, $\mathcal{A}'$ DPTA. Pak automaty jsou ekvivalentní vzhledem k PTCTL ($\mathcal{A} \cong_{PTCTL} \mathcal{A}'$), jestliže platí

$$\mathcal{A} \vDash \phi \Leftrightarrow \mathcal{A}' \vDash \phi.$$

2. Připomeňme, že $\mathcal{E}_0$ značí valuaci hodin fomule, kde jsou všechny hodiny nastaveny na 0.

5.1.1 Převod PTA na ekvivalentní DPTA

Pro ověření splnitelnosti PTCTL formule pro PTA stačí nalézt DPTA, jehož sémantika je se sémantikou daného PTA v relaci $\cong_{PTCTL}$. Pokud všechny stavy v jisté relaci budou zachovávat splnitelnost PTCTL formule, pak jistě i počáteční stavy, a tedy automaty budou ekvivalentní podle definice 44.

Převod libovolného PTA na ekvivalentní DPTA je zřejmý. Pro každou lokaci l zadefinujeme funkci $part(l)$ jako funkci, která rozdělí invariant lokace na jedinou zónu, a to invariant lokace samotný. Funkce $trans(l)$ bude mít pak definiční obor tvořený jediným prvkem $inv(l)$ a platí $trans(l)(inv(l)) = 1$.

Lemma 45. *Mějme libovolný PTA* $\mathcal{A} = (L, l_0, \mathcal{X}, Act, inv, enab, prob, \mathcal{L})$ *a DPTA* $\mathcal{A}'$ *definovaný jako* $\mathcal{A}' = (L, l_0, \mathcal{X}, Act, inv, enab, prob, part, trans, \mathcal{L})$, *kde*

$$\forall l \in L\colon part(l) = \{inv(l)\},$$

$$trans(l)(inv(l)) = 1.$$

Pak platí $[\![\mathcal{A}]\!] \cong_{PTCTL} [\![\mathcal{A}']\!]$.

Důkaz. Ukážeme, že existuje $\mathcal{R}$ taková, že platí $[\![\mathcal{A}]\!] \cong_{PTCTL} [\![\mathcal{A}']\!]$. Důkaz povedeme indukcí vzhledem k hloubce zanoření formule ϕ. Hloubku zanoření formule chápeme jako hloubku derivačního stromu odpovídajícímu jejímu odvození. Dokážeme, že $[\![\mathcal{A}]\!] \cong_{PTCTL} [\![\mathcal{A}']\!]$ uvažujeme-li pouze atomické PTCTL formule (tedy formule, které jsou atomickou propozicí nebo hodinovým omezením). To bude tvořit bázi našeho důkazu. Induktivně pak bude dokázána platnost celého lemmatu.

Definice 46. Buďte $\mathcal{A}$ PTA a $\mathcal{A}'$ DPTA se shodnou množinou lokací a hodin. Označme $[\![\mathcal{A}]\!] = (S, s_0, Act, Steps, \mathcal{L})$ a $[\![\mathcal{A}']\!] = (S', s'_0, Act', Steps', \mathcal{L}')$. Pak relace $\mathcal{R} : S \to S'$ splňuje pro všechna $(l, v) \in S, (l', v', i) \in S'$:

$$(l, v)\,\mathcal{R}(l', v', i) \overset{def}{\iff} l = l', v\ = v'.$$

Snadno se nahlédne, že tímto způsobem definovaná relace splňuje $s_0\,\mathcal{R}\,s'_0$ a že je úplná.

Ať je $\mathcal{E}$ valuace hodin formule, ϕ libovolná PTCTL formule, $[\![\mathcal{A}]\!] = (S, s_0, Act, Steps, \mathcal{L})$, $s = (l, v) \in S$ libovolné, $[\![\mathcal{A}']\!] = (S', s'_0, Act', Steps', \mathcal{L}')$ a $s' \in S'$ stav takový, že $s\,\mathcal{R}\,s'$ (tedy $s' = (l, v, i)$). Nejprve dokážeme bázi lemmatu (tj. dokážeme jeho platnost pro atomické formule).

- Formule $\phi = p$, kde $p \in AP$,
  $(s, \mathcal{E}) \vDash \phi \Leftrightarrow p \in \mathcal{L}(s)$, z definice relace $\mathcal{R}$ plyne, že $p \in \mathcal{L}'(s') \Leftrightarrow p \in \mathcal{L}(s)$ (neboť stavy mají shodné lokace a značkovací funkce je závislá pouze na lokaci), a tedy

  $$(s', \mathcal{E}) \vDash p \Leftrightarrow p \in \mathcal{L}(s) \Leftrightarrow (s, \mathcal{E}) \vDash p.$$

- Formule $\phi = \zeta, \zeta \in \mathcal{B}(\mathcal{X})$,
  pak tvrzení zřejmě platí, neboť podle definice relace $\mathcal{R}$ jsou valuace ve stavech s a s' stejné, a tedy

  $$((l, v), \mathcal{E}) \vDash \zeta \Leftrightarrow v\ \cup \mathcal{E} \vDash \phi \Leftrightarrow ((l, v, i), \mathcal{E}) \vDash \zeta.$$

Dokázali platnost tvrzení lemmatu 45 pro atomické formule, které tvoří základ každé PTCTL formule – tedy pro formule s hloubkou zanoření 1. Předpokládejme nyní, že lemma platí pro všechny formule ϕ s hloubkou zanoření nejvýše k a dokážeme, že pak platí i pro hloubku zanoření $k+1$.

- Důkaz platnosti pro formule tvaru $\neg\phi, \phi \vee \psi$ je zřejmý, neboť můžeme využít indukčního předpokladu pro formule ϕ, ψ.
- Formule $\phi = z.\psi$, kde ψ je PTCTL formule s hloubkou zanoření o jedna menší než ϕ, platí $((l,v),\mathcal{E}) \vDash z.\psi \Leftrightarrow ((l,v),\mathcal{E}[z:=0]) \vDash \psi \Leftrightarrow ((l,v,i),\mathcal{E}[z:=0]) \vDash \psi$. Využijeme indukčního předpokladu a dostáváme $(s,\mathcal{E}) \vDash \phi \Leftrightarrow (s',\mathcal{E}) \vDash \phi$.
- Důkaz tvrzení pro formuli ϕ tvaru $\phi = P_{\bowtie k}[\phi_1 \,\mathrm{U}\, \phi_2]$ provedeme tak, že ukážeme, že pro libovolnou cestu $\omega \in Path_s$ můžeme sestrojit cestu $\bar{\omega} \in Path_{s'}$, aby platilo $\omega \vDash \phi_1 \,\mathrm{U}\, \phi_2 \Leftrightarrow \bar{\omega} \vDash \phi_1 \,\mathrm{U}\, \phi_2$. Totéž učiníme v opačném směru, pro každou cestu $\omega' \in Path'_s$ sestrojíme cestu $\bar{\omega}' \in Path_s$. Navíc se ukáže, že odpovídající cesty nejsme schopni z hlediska pravděpodobnosti rozlišit žádným konečným prefixem. Pak tedy nutně bude pro všechny možné strategie σ, σ' platit $Pr(\{\omega \in Path_s^\sigma \mid \omega \vDash \phi \,\mathrm{U}\, \psi\}) \leq Pr(\{\omega' \mid \omega' \in Path_{s'}^\sigma, \omega' \vDash \phi \,\mathrm{U}\, \psi\})$ a zároveň $Pr(\{\omega \in Path_s^\sigma \mid \omega \vDash \phi \,\mathrm{U}\, \psi\}) \geq Pr(\{\omega' \mid \omega' \in Path_{s'}^{\sigma'}, \omega' \vDash \phi \,\mathrm{U}\, \psi\})$, tedy nutně $Pr(\{\omega \in Path_s^\sigma \mid \omega \vDash \phi \,\mathrm{U}\, \psi\}) = Pr(\{\omega' \mid \omega' \in Path_{s'}^{\sigma'}, \omega' \vDash \phi \,\mathrm{U}\, \psi\})$.
  Proto platí i $(s,\mathcal{E}) \vDash P_{\bowtie k}[\phi_1 \,\mathrm{U}\, \phi_2] \Leftrightarrow (s',\mathcal{E}) \vDash P_{\bowtie k}[\phi_1 \,\mathrm{U}\, \phi_2]$.

  Nechť $\omega = s_0(a_1,p_1)s_1(a_2p_2)\cdots \in Path_s$. Index m_i bude značit index stavu, do kterého byl učiněn i-tý diskrétní přechod (tj. je-li první diskrétní přechod učiněn ze stavu s_0 do stavu s_1, pak $m_1 = 1$). Odpovídající cestu $\bar{\omega}$ sestavíme z úseků

$$[\bar{\omega}]_i = (l_{m_{i-1}}, v_{m_{i-1}}, 0)(a_{m_{i-1}+1}+1, p_{m_{i-1}+1}) \ldots (l_{m_i-1}, v_{m_i-1}, 0) \\ (\varepsilon, trans(l_{m_i-1})([v_{m_i-1},1]))(l_{m_i-1}, v_{m_i-1}, 1)(a_{m_i}, p_{m_i}).$$

  Snadno se nahlédne, že $l_{m_{i-1}} = \cdots = l_{m_i-1} = l_i$ a $[v_{m_{i-1}}] = \cdots = [v_{m_i-1}] = inv(l_i)$, a tedy konstrukce $[\bar{\omega}]_i$ je korektní vzhledem k sémantice $[\![\mathcal{A}']\!]$.
  Jestliže $(l_{m_i-1}, v_{m_i-1}) \xrightarrow{a_{m_i}, p_{m_i}} (l_{m_i}, v_{m_i})$, pak i $(l_{m_i-1}, v_{m_i-1}, 1) \xrightarrow{a_{m_i}, p_{m_i}} (l_{m_i}, v_{m_i}, 0)$ (viz konstrukce $\mathcal{A}'$ a definice sémantiky DPTA), a tedy spojení úseků je rovněž korektní vzhledem k sémantice $[\![\mathcal{A}']\!]$.

  Ukážeme, že pro libovolné $\omega \in Path_s$ platí $\omega \vDash \phi_1 \,\mathrm{U}\, \phi_2 \Leftrightarrow \bar{\omega} \vDash \phi_1 \,\mathrm{U}\, \phi_2$ a zároveň nejsme cesty schopni rozlišit z hlediska pravděpodobnosti žádným konečným přefixem. Lépe řečeno, ke každé cylindrické množině obsahující ω, lze nalézt cylindrickou množinu obsahující $\bar{\omega}$ spjatou se stejnou pravděpodobností.
  Pak nutně platí $Pr(\{\omega \in Path_s^\sigma \mid \omega \vDash \phi \,\mathrm{U}\, \psi\}) \leq Pr(\{\omega' \mid \omega' \in Path_{s'}^{\sigma'}, \omega' \vDash \phi \,\mathrm{U}\, \psi\})$.

  Vezměme libovolnou cestu $\omega \in Path_s$ a valuaci hodin formule $\mathcal{E}$, pak

$$(\omega,\mathcal{E}) \vDash_\psi \phi \,\mathrm{U}\, \psi \Leftrightarrow \exists k \in \mathbb{N}, t \in [0, \mathscr{D}_\omega(k+1) - \mathscr{D}_\omega(k)] :$$

  (i) $(\omega(k)+t, \mathcal{E} + \mathscr{D}_\omega(k) + t) \vDash \psi$,
  (ii) $\forall t' \leq t.(\omega(k)+t', \mathcal{E} + \mathscr{D}_\omega(k) + t') \vDash \phi \vee \psi$,
  (iii) $\forall i < k. \forall t' \in [0, \mathscr{D}_\omega(j+1) - \mathscr{D}_\omega(j)].(\omega(j)+t', \mathcal{E} + \mathscr{D}_\omega(j) + t') \vDash \phi \vee \psi$.

využijeme indukčního předpokladu a dostáváme

$$(\omega, \mathcal{E}) \vDash_{\psi} \phi \,\mathrm{U}\, \psi \Leftrightarrow \exists k \ \in \mathbb{N}, t \in [0, \mathscr{D}_\omega(k+1) - \mathscr{D}_\omega(k)] :$$

(i) $(s_k^i + t, \mathcal{E} + \mathscr{D}_\omega(k) + t) \vDash \psi$,

(ii) $\forall t' \leq t.(s_k^i + t', \mathcal{E} + \mathscr{D}_\omega(k) + t') \vDash \phi \vee \psi$,

(iii) $\forall i < k. \forall t' \in [0, \mathscr{D}_\omega(j+1) - \mathscr{D}_\omega(j)].(s_j^i + t', \mathcal{E} + \mathscr{D}_\omega(j) + t') \vDash \phi \vee \psi$,

kde s_n značí $\omega(n)$. Díky konstrukci $\bar{\omega}$ dostáváme

$$(\omega, \mathcal{E}) \vDash_p \phi \,\mathrm{U}\, \psi \Leftrightarrow (\bar{\omega}, \mathcal{E}) \vDash_p \phi \,\mathrm{U}\, \psi.$$

Uvažme libovolný prefix $\pi_\omega \in Path_s^{fin} = s_0(a_1, p_1)s_1 \ldots (a_n, p_n)s_n$ cesty ω. Označme $m_{\max}$ jako nejvyšší index stavu, do kterého byl učiněn diskrétní přechod (tj. v cestě bylo učiněno $\max$ diskrétních přechodů). Pak cesta

$$\pi_{\bar{\omega}} = [\bar{\omega}]_1[\bar{\omega}]_2 \ldots\ldots [\bar{\omega}]_{max}(s^0_{max})(a_{max+1}, p_{max+1}) \ldots (a_n, p_n)(s_n^0)$$

je prefixem cesty $\bar{\omega} \in Path_{s'}$.

Nyní určíme pravděpodobnost obou cest. V cestě π_ω stačí uvažovat pouze pravděpodobnost diskrétních přechodů, časové přechody mají pravděpodobnost 1, tedy

$$\mathcal{P}(\pi_\omega) = \prod_{i \leq max} p_{m_i}(l_{m_i}).$$

Při výpočtu pravděpodobnosti cesty $\pi_{\bar{\omega}}$ stačí uvažovat pouze diskrétní přechody, které spojují jednotlivé úseky $[\bar{\omega}]_i$, neboť ostatní časové a diskrétní přechody mají pravděpodobnost 1. Tedy

$$\mathcal{P}(\pi_{\bar{\omega}}) = \prod_{i \leq max} p_{m_i}(l_{m_i}) = \mathcal{P}(\pi_\omega)$$

a dostáváme požadované.

Platí tedy $Pr(\{\omega \in Path_s^\sigma \mid \omega(1) \vDash \phi \,\mathrm{U}\, \psi\}) \leq Pr(\{\omega' \mid \omega' \in Path_{s'}^\sigma, \omega'(1) \vDash \phi \,\mathrm{U}\, \psi\})$.

Totéž nyní provedeme v opačném směru. Nechť $\omega' = s_0^{j_0}(a_1, p_1)s_1^{j_1}(a_2, p_2) \cdots \in Path_{s'}$. Obsahuje-li cesta stav typu 2, tak je její pravděpodobnost nulová (díky definici funkce $trans$ a $part$), a je tedy zbytečné takové cesty uvažovat, neboť neovlivní splnitelnost formule ϕ. Předpokládejme tedy dále, že cesta ω neosahuje stavy typu 2. Intuitivně, odpovídající cesta $\bar{\omega}'$ vznikne z ω' tak, že „vypustíme“ informaci o typu stavu a každý diskrétní přechod do stavu typu 1 ze stavu typu 0 bude nahrazen časovým přechodem o nulové délce trvání.

$$\bar{\omega'} = s_0(a'_1, p'_1)s_1(a'_2, p'_2) \ldots$$

kde $\forall i \geq 1$ platí jedno z následujících

$$j_{i-1} = 0 \wedge j_i = 1 \Rightarrow a'_i = 0 \in \mathbb{R}_0^+, p'_i = \mu_{s_i}$$

$$j_{i-1} = j_i \Rightarrow a'_i = a_i, p'_i = {p_i}^3.$$

Snadno nahlédne, že konstrukce cesty je korektní vzhledem k sémantice.

Ke každému konečnému prefixu $\pi_{\omega'} = s_0(a_1, p_1)s_1 \cdots \in Path^{fin}_{s'}$ existuje konečný prefix $\pi_{\bar{\omega'}} \in Path^{fin}_{s}$ se stejnou pravděpodobností. Důkaz je zřejmý.

Důkaz tvrzení $(\omega, \mathcal{E}) \vDash_\psi \phi \,\mathrm{U}\, \psi \Leftrightarrow (\bar{\omega}, \mathcal{E}) \vDash_\psi \phi \,\mathrm{U}\, \psi$ by se provedl obdobně jako v předchozím případě.

Platí tedy $Pr(\{\omega \in Path^{\sigma}_{s} \mid \omega(1) \vDash \phi \,\mathrm{U}\, \psi\}) \geq Pr(\{\omega' \mid \omega' \in Path^{\sigma}_{s'}, \omega'(1) \vDash \phi \,\mathrm{U}\, \psi\})$, což spolu s předchozími výsledky dává

$$(s, \mathcal{E}) \vDash P_{\bowtie k}[\phi_1 \,\mathrm{U}\, \phi_2] \Leftrightarrow (s', \mathcal{E}) \vDash P_{\bowtie k}[\phi_1 \,\mathrm{U}\, \phi_2]. \qquad \square$$

Věta 46.1. *Nechť $\mathcal{A}$ je libovolný PTA. Pak existuje DPTA $\mathcal{A}'$ takový, že $\mathcal{A} \cong_{PTCTL} \mathcal{A}'$.*

Důkaz. Platnost tvrzení vyplývá z lemmatu 45. Dokázali jsme, že pro každý PTA $\mathcal{A}$ existuje DPTA $\mathcal{A}'$ takový, že jejich sémantiky jsou v relaci $\cong_{PTCTL}$, tedy nejsme schopni rozlišit jejich stavy, které jsou v jisté relaci, žádnou PTCTL formulí. Z definice 43 rovněž vyplývá, že počáteční stavy automatů jsou spolu také v oné relaci, tedy nerozlišíme ani ty a bude platit, že $\mathcal{A}$ a $\mathcal{A}'$ jsou ekvivalentní vzhledem k PTCTL. $\square$

5.1.2 Převod DPTA na ekvivalentní PTA

Podobně jako v předchozí sekci zadáme převod libovolného DPTA na PTA. Dokážeme, že sémantiky jsou v relaci $\cong_{PTCTL}$ tak, že nalezneme totální bijekci $\mathcal{R}$, která pouze přejmenovává stavy, ale zachovává přechody a jiné důležité vlastnosti sytému.

Lemma 47. *Buď $\mathcal{A} = (L, l_0, \mathcal{X}, Act, inv, enab, prob, part, trans, \mathcal{L})$ DPTA. Definujeme PTA $\mathcal{A}' = (L', l'_0, \mathcal{X}, Act \cup \{\varepsilon\}, inv', enab', prob', \mathcal{L}')$, kde*

- *$L' \subseteq L \times \{0, 1, 2_{\mathcal{B}(\mathcal{X})}\} \times \mathcal{B}(\mathcal{X})$ tak, že platí*

$$(l, i, \alpha) \in L' \Leftrightarrow \alpha \in part(l)$$

a v případě, že $i = 2_\beta$, pak $\beta \in succ(\alpha)$ (tedy $succ(\alpha) = \{\beta\}$)

- $l'_0 = (l_0, 0, [v_0])$
- $inv(l, i, \alpha) = \begin{cases} \alpha, & \text{jestliže } i \in \{0, 1\}, \\ \alpha \cup ((\nearrow \alpha) \cap \beta), & \text{jestliže } i = 2_\beta. \end{cases}$
- *Funkci $prob'$ a stráž lokací $enab$ definujeme podle druhu jednotlivých stavů.*
  - $\mathbf{i = 0}$, $prob'((l, 0, \alpha), a) = p, enab(l, a) = True \Leftrightarrow$ $a = \varepsilon, \beta \in succ(\alpha)$ *a platí:*

$$p(\emptyset, (l, i', \alpha)) = \begin{cases} trans(l)(\alpha), & \text{jestliže } i' = 1 \\ 1 - trans(l)(\alpha), & \text{jestliže } i' = 2_\beta \end{cases}$$

  - $\mathbf{i = 1}$, $prob'((l, 1, \alpha), a) = p \Leftrightarrow$ $prob(l, a) = p'$, *kde* $p(X, (l', 0, \beta)) = p'(X, l')$ *pro libovolné* β *z množiny* $part(l')$, $enab((l, 1, \alpha), a) = enab(l, a)$

– $\mathbf{i} = \mathbf{2}_\beta$, *pak* $prob'((l, 2_\beta, \alpha), \varepsilon) = p$,
kde $p(\emptyset, (l, 0, \beta)) = \mu_{(\emptyset,(l,0,\beta))}$, $enab((l, 2_\beta, \alpha), \varepsilon) = True$

Jinak je funkce $prob'$ *nedefinována a stráž* $enab$ *neumožňuje přechod.*

- $\mathcal{L}(l, i, \alpha) = \mathcal{L}(l)$.

Pak platí $\mathcal{A} \cong_{PTCTL} \mathcal{A}'$.

Na obrázku 5.1 je znázorněna intuice převodu. Pro jednoduchost předpokládáme, že automat má jediné hodiny x. Pro lokaci l_1 je množina $part(l_1) = \{x \leq 1, x > 1 \wedge x \leq 2, x > 2 \wedge x \leq 3\}$, pro lokace l_2, l_3 jsou $part(l_2) = part(l_3) = \mathbb{R}_0^{+\mathcal{X}}$. Pro zónu $x \leq 1$ je pravděpodobnost přechodu nutně 0%, pro zónu $x > 1 \wedge x \leq 2$ je pravděpodobnost diskrétního přechodu rovna 20%, pro zónu $x > 2 \wedge x \leq 3$ je pravděpodobnost 100%.

Důkaz. Pro důkaz nalezneme relaci $\mathcal{R}$ a ukážeme, že tato relace je ve skutečnosti přejmenování stavů, a tedy platí $\mathcal{A} \cong_{PTCTL} \mathcal{A}'$, protože splnitelnost PTCTL formule nezávisí na pojmenování stavů.

Lemma 48. *Dvojice* $((l, i, \alpha), v)$ *je stavem* $[\![\mathcal{A}']\!]$ *právě tehdy, když platí jedna z následujících podmínek*

- $i \in \{0, 1\}$ *a* $\alpha = [v]_l$
- $i = 2_\beta$ *a buď je* $\alpha = [v]_l$ *a* $\beta = succ(v)$, *nebo* $\alpha \neq [v]_l$ *a* $\beta = [v]_l$.

Důkaz. Jestliže $((l, i, \alpha), v) \in S'$, kde $i \in \{0, 1\}$, pak $v \vDash inv((l, i, \alpha)) \Rightarrow v \vDash \alpha \Rightarrow [v]_l = \alpha$.
Je-li $i = 2_\beta$ a $((l, i, \alpha), v) \in S'$ stav, pak $v \in (\alpha \cup (\nearrow \alpha \cap \beta))$. Buď tedy platí $v \in \alpha$, a tedy $[v]_l = \alpha$ a zároveň $succ(v) = \beta$[4], nebo platí $v \in \beta$, tedy $[v]_l = \beta$. Podobně by se ukázal druhý směr. □

Definice 49. Buď $\mathcal{A}$ DPTA a $\mathcal{A}'$ PTA s vlastnostmi definovanými výše,
$[\![\mathcal{A}]\!] = (S, s_0, Act \cup \mathbb{R}_0^+ \cup \{\varepsilon\}, Steps, \mathcal{L})$ a $[\![\mathcal{A}']\!] = (S', s_0', Act \cup \mathbb{R}_0^+ \cup \{\varepsilon\}, Steps', \mathcal{L}')$.
Pak mezi množinou stavů S a S' zavedeme relaci $\mathcal{R} : S \to S'$ takovou,
že $\forall (l', v', i') \in [\![A]\!], ((l'', i'', \alpha), v'') \in [\![A']\!]$:

$$(l', v', i')\, \mathcal{R}((l'', i'', \alpha), v'') \overset{def}{\iff} l' = l'' = l, v' = v'' = v, i' = i'' = i.$$

Lemma 50. *Relace* $\mathcal{R}$ *je totální zobrazení.*

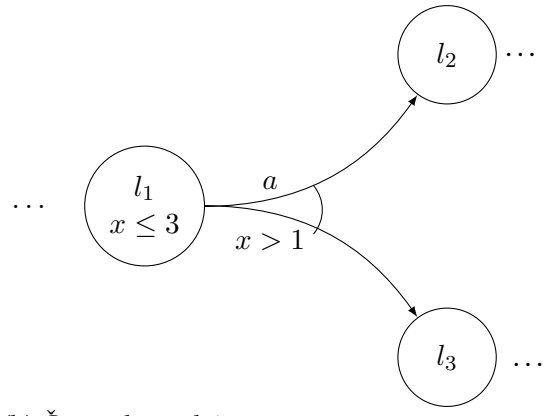
Důkaz. Ukážeme, že ke každému $s = (l, v, i) \in S$ existuje právě jedno $s' \in S'$ takové, že $s\, \mathcal{R}\, s'$. Stačí ukázat, že $s' = ((l, i, \alpha), v) \in S'$ a α je právě jedno.

- Jestliže $i \in \{0, 1\}$, pak z konstrukce PTA $\mathcal{A}'$ plyne, že $\forall \beta \in part(l) \colon (l, i, \beta) \in L$. Podle lemmatu 5.1.2 pro každý stav $((l, i, \alpha), v) \in S'$ platí $\alpha = [v]_l \in part(l)$. Tedy existuje právě jedno takové α, a tedy i právě jeden stav $s' \in S$, který je v relaci $\mathcal{R}$ se stavem $s \in S$.
- Pro $i = 2_\beta$ může zdánlivě nastat více případů. Ukážeme sporem, že je pro každé s právě jedno s'. Předpokládejme tedy, že existuje $s' = ((l, i, \alpha), v), s'' = ((l, i, \gamma), v) \in S'$, kde $\alpha \neq \gamma$, a platí $s\, \mathcal{R}\, s'$ a zároveň $s\, \mathcal{R}\, s''$. Podle lemmatu 5.1.2 platí buď

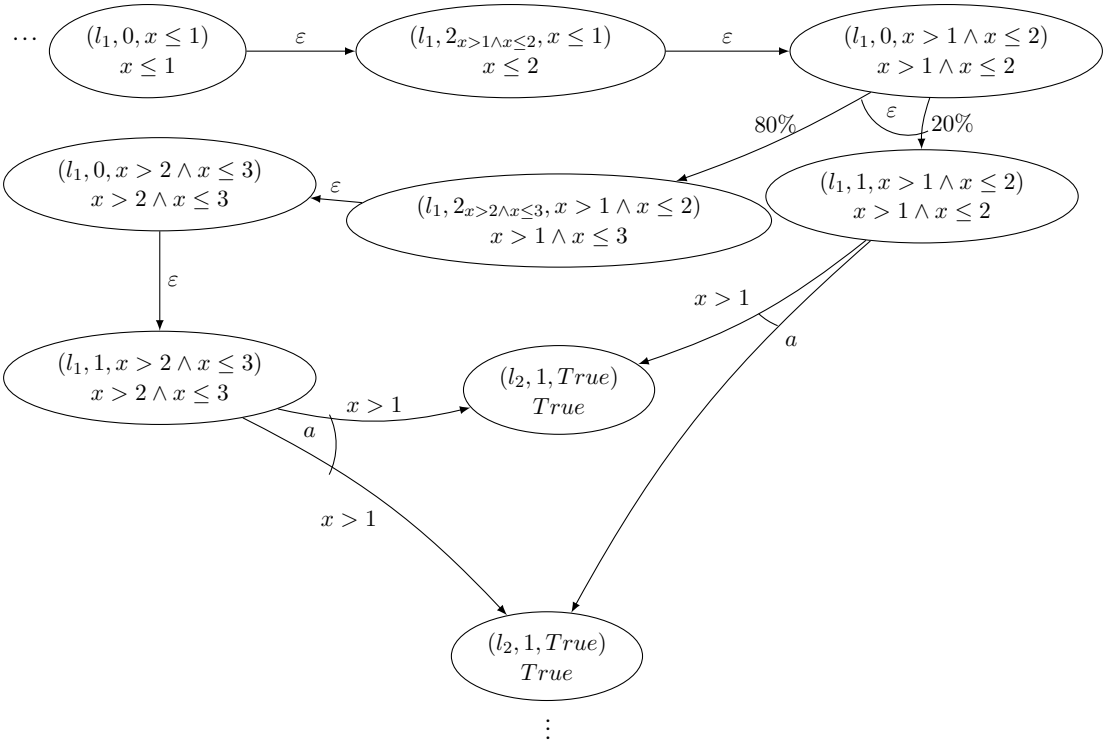
4. Neboť z konstrukce $\mathcal{A}'$ musí $\beta \in succ(\alpha)$, tedy $|succ(\alpha)| = 1$, a proto každá valuace z α musí mít definovaného následníka. Oním následník může být jedině β

Obrázek 5.1: Ukázka převodu DPTA na PTA

(a) Část DPTA



(b) Část odpovídajícího PTA



- $\alpha = [v]_l = \gamma$, pak se ale stavy neliší,
- nebo $\beta = [v]_l \neq \alpha \neq \gamma$. Oba stavy musí splňovat invariant lokace, tedy platí

$$v \in \alpha \cup (\nearrow \alpha \cap \beta) \bigwedge v \in \gamma \cup (\nearrow \gamma \cap \beta) \Rightarrow$$

$$v \in (\nearrow \alpha \cap \beta) \bigwedge v \in (\nearrow \gamma \cap \beta) \Rightarrow$$

$$\exists t, t' \in \mathbb{R}_0^+ : ((v - t) \in \alpha) \wedge ((v - t') \in \gamma).$$

BÚNO předpokládejme, že $t' > t$. Pak díky tomu, že $\alpha \neq \gamma$ existuje t'' takové, že $t' > t'' > t$ a pro všechny hodnoty $d \in [0, t' - t'']$ platí $(v - t') + d \in \gamma \cup \delta$ pro nějaké $\delta \neq \gamma \in part(l)$. Tedy $\delta \in succ(\gamma)$. Jistě je díky konvexnosti $\beta \neq \delta$. To je ale spor s jediným následníkem třídy γ, neboť z konstrukce $\mathcal{A}'$ platí $\beta \in succ(\gamma)$. Dostáváme tedy spor s předpoklady a $\alpha = \gamma$, relace $\mathcal{R}$ je totální zobrazení (totálnost vyplývá z lemmatu 5.1.2).

□

Lemma 51. *Relace $\mathcal{R}$ je injektivní.*

Důkaz. Důkaz přímo plyne z definice $\mathcal{R}$. □

Lemma 52. *Relace $\mathcal{R}$ je totální bijekcí.*

Důkaz. Stačí ukázat, že relace $\mathcal{R}$ je surjektivní. Vše ostatní již bylo dokázáno. Vzor k libovolnému stavu $((l, i, \alpha), v) \in S'$ je stav $(l, v, i) \in S$. Pokud $((l, i, \alpha), v) \in S'$, pak jistě v splňuje omezení dané invariantem $inv((l, i, \alpha)) \subseteq inv(l)$ (viz definice $\mathcal{A}'$), a tím spíš platí $v \in inv(l)$, a tedy $(l, v, i) \in S$. □

Dále v textu budeme občas na $\mathcal{R}$ nahlížet jako na funkci a používat značení

$$\mathcal{R}(s) = s'. s \ \mathcal{R} \ s'.$$

Nyní ukážeme, že relace $\mathcal{R}$ pouze „přejmenovává“ stavy – nahradíme každý výskyt stavu s v popisu $[\![A]\!]$ za stav s', který je s ním v relaci $\mathcal{R}$, a ukážeme, že výsledek je shodný s $[\![A']\!]$. Díky tomu, že $\mathcal{R}$ je bijekce, stačí, že tak učiníme pouze v tomto jednom směru.

Nechť $s \in S$ je libovolné, $s' = \mathcal{R}(s)$.

- Protože $\mathcal{R}$ je totální bijekce, pak nahrazením každého stavu $s \in S$ za stav $s' \in S'$ dostáváme právě množinu S'.
- Iniciální stav MDP $[\![A]\!]$ je stav $s_0 = (l_0, v_0, 0)$, z konstrukce $\mathcal{A}'$ a definice sémantiky PTA dostáváme, že iniciální stav $[\![A']\!]$ je $s'_0 = ((l_0, 0, [v_0]), v_0)$. Tedy platí $s_0 \, \mathcal{R} \, s'_0$.
- Pro každé dva stavy $s_1, s_2 \in S$ a každé $a \in Act$ ukážeme, že $Steps(s_1, a)(s_2) = Steps'(\mathcal{R}(s_1), a)(\mathcal{R}(s_2))$[5].
  - **Stav s_1 je typu 0.**
    Pak i $\mathcal{R}(s_1)$ je typu 0 a dle konstrukce $[\![A']\!]$ a definice sémantiky PTA a DPTA jsou z obou stavů možné odchozí akce pouze časové a ε akce.
    Je-li $s_1 = (l, v, 0)$, pak $\mathcal{R}(s_1) = ((l, 0, [v]_l), v)$ (dle definice $\mathcal{R}$ a lemmatu 5.1.2).
    1. *časový přechod*:
       $Steps(s_1, t)$ je definováno pouze pro $t \in \mathbb{R}_0^+$, pro které $\forall t' \leq t \colon v + t \in [v]_l$ a $Steps(s_1, t) = \mu_{s_1 + t}$. Jinak je nedefinováno.
       Ukážeme, že právě pro tato t je definováno i $Steps'(\mathcal{R}(s_1), t)$.
       $Steps'(\mathcal{R}(s_1), t)$ je definováno pouze pro $t \in \mathbb{R}_0^+$, pro které

$$\forall t' \leq t \colon v + t \vDash inv((l, 0, [v]_l))$$

5. Uvědomme si, že není třeba ukazovat opačný směr díky tomu, že $\mathcal{R}$ je bijekce.

a $Steps(\mathcal{R}(s_1), t) = \mu_{\mathcal{R}(s_1)+t}$.
Z definice invariantu (viz konstrukce $\mathcal{A}'$) je to právě pro ta t, pro která platí

$$\forall t' \leq t \colon v + t \in [v]_l.$$

Zároveň platí $\mathcal{R}(s_1 + t) = \mathcal{R}(s_1) + t$. Dostáváme tedy požadované.

2. *diskrétní přechod*:
$Steps(s_1, \varepsilon)$ je definováno $\forall (l, v, i) \in S$ jako

$$Steps(s_1, \varepsilon)(l, v, i) = \begin{cases} trans(l)([v]_l), & \text{jestliže } i = 1, \\ 1 - trans(l)([v]_l), & \text{jestliže } i = 2_{succ(v)}. \end{cases}$$

Pravděpodobnostní funkce $prob'$ je z konstrukce $\mathcal{A}'$ definována jako

$$prob'((l, 0, [v]_l), \varepsilon)(\emptyset, (l, i, [v]_l)) = \begin{cases} trans(l)([v]_l), & \text{jestliže } i = 1, \\ 1 - trans(l)([v]_l), & \text{jestliže } i = 2_\beta, \end{cases}$$

kde $\beta \in succ([v]_l)$, z toho nutně vyplývá $\beta = succ(v)$. Stráž $enab((l, 0, [v]_l), \varepsilon) = True$, pak z definice sémantiky PTA dostáváme, že $\forall ((l, i, [v]_l), v) \in S'$ je

$$Steps(\mathcal{R}(s_1), \varepsilon)((l, i, [v]_l), v) = \begin{cases} trans(l)([v]_l), & \text{jestliže } i = 1, \\ 1 - trans(l)([v]_l), & \text{jestliže } i = 2_{succ(v)}. \end{cases}$$

Neboť zřejmě platí $((l, i, [v]_l), v) = \mathcal{R}((l, v, i))$, dostáváme požadované.

– **Stav s_1 je typu 1.**
Pak i $\mathcal{R}(s_1)$ je typu 1 a dle konstrukce A' a definice sémantiky PTA a DPTA jsou z obou stavů možné odchozí akce buď časové anebo diskrétní.
Je-li $s_1 = (l, v, 1)$, pak $\mathcal{R}(s_1) = ((l, 1, [v]_l), v)$ (dle definice $\mathcal{R}$ a lemmatu 5.1.2).

1. *časový přechod*:
Ukázalo by se stejně jako pro stav typu 0.
2. *diskrétní přechod*:
$Steps(s_1, a)$ je definováno $\Leftrightarrow$ $prob(l, a) = p$ je definováno a $\forall (l', v', 0) \in S$ takové, že $v' \vDash enab(l, a)$, platí

$$Steps(s_1, a)(l', v', 0) = \sum_{X \subseteq \mathcal{X} \wedge v' = v[X:=0]} p(X, l'),$$

jestliže $v' \vDash enab(l', a)$.
Z konstrukce $\mathcal{A}'$ plyne, že $prob'((l, 1, [v]_l), a) = p'$ je definováno právě tehdy, když je definováno $prob(l, a) = p$, kde $p'(X, (l', 0, \beta)) = p(X, l')$ pro libovolné $\beta \in part(l')$. Navíc platí $enab((l', 0, \beta), a) = enab(l', a)$. Z toho díky sémantice PTA dostáváme, že $\forall ((l', 0, \beta), v') \in S'$ je

$$Steps(\mathcal{R}(s_1), ((l', 0, \beta), v')) = \sum_{X \subseteq \mathcal{X} \wedge v' = v[X:=0]} p'(X, (l', 0, \beta)) = p(X, l'),$$

$\Leftrightarrow$ $prob(l, a)$ je definováno a $v' \vDash enab(l', a)$. Takové β je právě jedno dle lemmatu 5.1.2 a platí $((l', 0, \beta), v') = \mathcal{R}((l', v', 0))$. Ukázali jsme požadované.

– **Stav s_1 je typu 2.** Pak i $\mathcal{R}(s_1)$ je typu 2 a dle konstrukce $[\![A']\!]$ a definice sémantiky PTA a DPTA jsou z obou stavů možné odchozí akce pouze časové a ε akce.

Je-li $s_1 = (l, v, 2_\alpha)$, pak buď

$$\begin{aligned} & \mathcal{R}(s_1) && = ((l, 2_{succ(v)}, [v]_l), v) \text{ a } \alpha \neq [v]_l, \\ \text{nebo} \quad & \mathcal{R}(s_1) && = ((l, 2_{[v]_l}, \beta \neq [v_l]), v) \text{ a } \alpha \in succ(\beta), \end{aligned}$$

1. *časový přechod*: $Steps(s_1, t)$ je definováno pouze pro $t \in \mathbb{R}_0^+$ taková, že $\forall t' \leq t \colon v + t' \in ([v]_l \cup \alpha)$ a $Steps(s_1, t) = \mu_{s_1+t}$. Jinak je nedefinováno.
   Ukážeme, že právě pro tato t je definováno i $Steps(\mathcal{R}(s_1), t)$.
   $Steps(\mathcal{R}(s_1), t)$ je definováno $\Leftrightarrow \forall t' \leq t \colon v + t' \vDash inv((l, 2_\alpha, \beta))$.

   Rozebereme případy podle toho, jakého tvaru je $\mathcal{R}(s_1)$.
   * $\mathcal{R}(s_1) = ((l, 2_\alpha, [v]_l), v)$, kde $\alpha \neq [v_l]$, pak z definice invariantu dostáváme, že $Steps(\mathcal{R}(s_1), t)$ je definováno $\Leftrightarrow \forall t' \leq t \colon v + t' \in [v]_l \cup \alpha$.
     $Steps(\mathcal{R}(s_1), t) = \mu_{(\mathcal{R}(s_1)+t)}$ a platí $\mathcal{R}(s_1) + t = \mathcal{R}(s_1 + t) = s_1 + t$, dostáváme tedy požadované.
   * $\mathcal{R}(s_1) = ((l, 2_{\alpha=[v]_l}, \beta \neq [v_l]), v)$, pak z definice invariantu dostáváme, že $Steps(\mathcal{R}(s_1), t)$ je definováno $\Leftrightarrow \forall t' \leq t \colon v + t' \in [v]_l \cup \beta$, což z důvodu, že $[v]_l \in succ(\beta)$ je právě když $\forall t' \leq t \colon v + t' \in [v]_l$. Tedy právě když $Steps(s_1)$ je definováno (neboť $[v]_l = \alpha$). Dostáváme požadované.
2. *diskrétní přechod:* $Steps(s_1, \varepsilon)(s_2)$ je definováno právě tehdy, když $s_2 = (l, v, 0)$ a zároveň $v \in \alpha$.
   $Steps(\mathcal{R}(s_1), \varepsilon)(s_2')$ je definováno $\Leftrightarrow s_2' = ((l, 0, \alpha), v)$ a $v \vDash inv((l, 0, \alpha))$,
   tj. $\Leftrightarrow v \in \alpha$. Opět platí $Steps(\mathcal{R}(s_1), \varepsilon)(s_2') = Steps(\mathcal{R}(s_1), \varepsilon)(s_2')$ a $s_2' = \mathcal{R}(s_2)$.

• Zbývá ukázat, že $\forall s \in S \colon \mathcal{L}(s) = \mathcal{L}'(\mathcal{R}(s))$. Značkovací funkce závisí pouze na lokaci, která je pro oba stavy stejná, čili tvrzení platí.

□

Věta 52.1. *Nechť $\mathcal{A}$ je libovolný DPTA. Pak existuje PTA $\mathcal{A}'$ takový, že $[\![\mathcal{A}]\!] \cong_{PTCTL} [\![\mathcal{A}']\!]$.*

Důkaz. Platnost tvrzení vyplývá z lemmatu 47. □

Poznámka 53. Dokázali jsme sice, že automaty jsou ekvivalentní pouze vzhledem k PTCTL, ale protože se během důkazu ukázalo, že se jedná pouze o přejmenování stavů, pak jistě i pro další logiky, které jsou interpretovány nad stavy sémantik, budou automaty ekvivalentní.

Ukázali jsme tedy, že sémantiky jsou stejné až na pojmenování stavů, a tedy algoritmy využívané pro PTA ověřování modelu se dají použít i pro DPTA. Nárůst systému po převodu je diskutován v další kapitole.

5.2 Ověřování modelu pro DPTA

Ukázalo se, že libovolný DPTA lze převést na ekvivalentní PTA vzhledem k PTCTL formulím. Při převodu sice nedochází k nárůstu stavů (kterých je v obou případech nespočetně

mnoho), nicméně složitost verifikace PTA vychází mimo jiné z počtu lokací a počtu pravděpodobnostních přechodů. Analyzujeme tedy nárůst, který při tomto převodu vzniká. Přirozeně se pak zvyšuje složitost algoritmů používaných při ověřování modelu PTA, cheme-li je aplikovat i na DPTA.

Mějme DPTA s množinou lokací L. Zkonstruujeme-li odpovídající PTA, pak každé lokaci $l \in L$ odpovídá v nejhorším případě $3 \cdot |part(l)|$ lokací PTA.

Poznámka 54. Poznamenejme, že číslo $|part(l)|$ může být ve skutečnosti velmi velké, neboť v případě, že invariant lokace není omezen, jsme schopni dělit časový prostor téměř do nekonečna. Ve skutečnosti má smysl však uvažovat pouze exponenciální počet částí vzhledem k počtu hodin, neboť z důkazu regionové abstrakce vyplývá, že valuace vyšší, než je určitá hodnota, nejsme již schopni rozlišit.

V případě lokací typu 0 a 2 nedochází k citelnému nárůstu počtu přechodů (z každé takové lokace je možný pouze konstantní počet přechodů). V případě lokace typu 1 jsme však přidali přechod do všech lokací typu 0, které se shodují na lokaci původního DPTA. Takových lokací je až $|part(l)|$.

Nárůst počtu lokací i počtu přechodů ekvivalentního PTA je lineární vzhledem k velikosti největší množině $|part(l)|$, kde l je lokace DPTA.

Kapitola 6

Závěr

V práci byly představeny formalismy používané pro modelování systémů s reálným časem a pravděpodobností. Byly popsány logiky, které jsou vhodné ke specifikaci jejich vlastností, a byl nastíněn způsob, jakým probíhá ověřování modelu. Způsob modelování byl demonstrován na jednoduchých příkladech. Taktéž byly uvedeny problémy, které je třeba vzít při verifikaci v úvahu, a způsob jejich řešení.

Povedlo se rozšířit formalismus PTA tak, že je umožněno zachytit setrvání v určitém stavu po dobu ovlivněnou pravděpodobnostní funkcí. Rozšíření bylo porovnáno s nerozšířenou verzí a ukázalo se, že v obou případech jsme schopni popsat stejná chování systému. Navíc je podán návod, jak zkonstruovat z rozšířeného PTA nerozšířený, který je ekvivalentní vzhledem k PTCTL formulím. Diskutován je rovněž nárůst velikosti systému, ke kterému při převodu dochází.

Literatura

[1] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *THEORETICAL COMPUTER SCIENCE*, 138:3–34, 1995.

[2] Rajeev Alur and David L. Dill. A Theory of Timed Automata. *Theoretical Computer Science*, 126:183–235, 1994.

[3] Christel Baier, Nathalie Bertrand, Patricia Bouyer, Thomas Brihaye, and Marcus Größer. Probabilistic and Topological Semantics for Timed Automata. In V. Arvind and Sanjiva Prasad, editors, *FSTTCS 2007: Foundations of Software Technology and Theoretical Computer Science*, volume 4855 of *Lecture Notes in Computer Science*, pages 179–191. Springer Berlin Heidelberg, 2007.

[4] Christel Baier, Boudewijn Haverkort, Holger Hermanns, and Joost-Pieter Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 29(7):2003, 2003.

[5] Christel Baier, Joost-Pieter Katoen, et al. *Principles of model checking*, volume 26202649. MIT press Cambridge, 2008.

[6] Gerd Behrmann, Alexandre David, and Kim G Larsen. A Tutorial on Uppaal 4.0. *Department of computer science, Aalborg university*, 2006.

[7] P. Bouyer and F. Laroussinie. Model checking and timed CTL. `http://www.comp.nus.edu.sg/~cs5270/2006-semesterII/chapt6.pdf`, 2014. [Online].

[8] P. Bouyer and F. Laroussinie. Model Checking Timed Automata. `http://www.iste.co.uk/data/doc_wrkszvritcbv.pdf`, 2014. [Online].

[9] Peter Bulychev, Alexandre David, Kim Gulstrand Larsen, Marius Mikučionis, Danny Bøgsted Poulsen, Axel Legay, and Zheng Wang. UPPAAL-SMC: Statistical model checking for priced timed automata. *arXiv preprint arXiv:1207.1272*, 2012.

[10] P.W. Glynn. A GSMP formalism for discrete event systems. *Proceedings of the IEEE*, 77(1):14–23, leden 1989.

[11] Hans Hansson and Bengt Jonsson. A Logic for Reasoning about Time and Reliability. *Formal Aspects of Computing*, 6:102–111, 1994.

[12] David Henriques, Joao G Martins, Paolo Zuliani, André Platzer, and Edmund M Clarke. Statistical model checking for markov decision processes. In *Quantitative Evaluation of Systems (QEST), 2012 Ninth International Conference on*, pages 84–93. IEEE, 2012.

[13] Thomas A. Henzinger, Xavier Nicollin, Joseph Sifakis, and Sergio Yovine. Symbolic Model Checking for Real-time Systems. *Information and Computation*, 111:394–406, 1992.

[14] Frédéric Herbreteau, B Srivathsan, and Igor Walukiewicz. Efficient emptiness check for timed büchi automata. In *Computer Aided Verification*, pages 148–161. Springer, 2010.

[15] Ed Brinksma Holger Hermanns and Joost-Pieter Katoen. Lectures on Formal Methods and PerformanceAnalysis. 2001.

[16] Lukáš Holík. Rozhodnutelnost v temporálních logikách. Master's thesis, Masarykova univerzita Fakulta informatiky, 2006.

[17] D Kartson, Gianfranco Balbo, S Donatelli, G Franceschinis, and Giuseppe Conte. *Modelling with generalized stochastic Petri nets*. John Wiley & Sons, Inc., 1994.

[18] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of Probabilistic Real-time Systems. In G. Gopalakrishnan and S. Qadeer, editors, *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

[19] M. Kwiatkowska and D. Parker. Advances in Probabilistic Model Checking. In T. Nipkow, O. Grumberg, and B. Hauptmann, editors, *Software Safety and Security - Tools for Analysis and Verification*, volume 33 of *NATO Science for Peace and Security Series - D: Information and Communication Security*, pages 126–151. IOS Press, 2012.

[20] Marta Kwiatkowska, Gethin Norman, David Parker, and Jeremy Sproston. Verification of Real-Time Probabilistic Systems. *Modeling and Verification of Real-Time Systems: Formalisms and Software Tools*, pages 249–288, 2007.

[21] Marta Kwiatkowska, Gethin Norman, Roberto Segala, and Jeremy Sproston. Automatic verification of real-time systems with discrete probability distributions. *Theoretical Computer Science*, 282:2002, 1999.

[22] J. Rutten, M. Kwiatkowska, G. Norman, and D. Parker. *Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems,* P. Panangaden and F. van Breugel (eds.), volume 23 of *CRM Monograph Series*. American Mathematical Society, 2004.

[23] Roberto Segala. *Modeling and Verification of Randomized Distributed Real-time Systems*. PhD thesis, Cambridge, MA, USA, 1995. Not available from Univ. Microfilms Int.

[24] Md Tawhid Bin Waez, Juergen Dingel, and Karen Rudie. Timed Automata for the Development of Real-Time Systems, 2011.

[25] Håkan LS Younes, Marta Kwiatkowska, Gethin Norman, and David Parker. Numerical vs. statistical probabilistic model checking. *International Journal on Software Tools for Technology Transfer*, 8(3):216–228, 2006.

[26] Sergio Yovine. Model checking timed automata. In *In European Educational Forum: School on Embedded Systems*, pages 114–152. Springer-Verlag, 1998.