Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Editor Petriho Sítí

2019

Roman Wehmhöner

Vedoucí práce: Mgr. Petr Osička, Ph.D.

Studijní obor: Aplikovaná informatika, prezenční forma

Bibliografické údaje

Autor:	Roman Wehmhöner
Název práce:	Editor Petriho Sítí
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2019
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Petr Osička, Ph.D.
Počet stran:	25
Přílohy:	1 CD/DVD
Jazyk práce:	český

Bibliograhic info

Author:	Roman Wehmhöner
Title:	Petri Nets Editor
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2019
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Petr Osička, Ph.D.
Page count:	25
Supplements:	1 CD/DVD
Thesis language:	Czech

Anotace

Cílem bakalářské práce bylo vytvořit editor petriho sítí umožňující jednoduché a pohodlné ovládání. Editor také obsahuje základní nástroje pro analýzu petriho sítí.

Synopsis

Klíčová slova: styl textu; závěrečná práce; dokumentace; ukázkový text

Keywords: text style; thesis; documentation; sample text

Děkuji, děkuji, děkuji.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

1 Petriho sítě **8**
- 1.1 Základní popis . . . 8
- 1.2 Vizuální zobrazení sítě . . . 9
- 1.3 Zkrácený zápis ohodnocení sítě . . . 10
- 1.4 Využití Petriho sítí . . . 10
- 1.5 Graf dosažitelnosti . . . 11
  - 1.5.1 Vlastnosti odvoditelné z Grafu dosažitelnosti . . . 11
  - 1.5.2 Příklady grafu dosažitelnosti s vlastnostmi . . . 13
- 1.6 Graf pokrytí . . . 14
  - 1.6.1 Sestrojení grafu . . . 14
  - 1.6.2 Různé výsledky grafu pokrytí . . . 16
  - 1.6.3 Upravená verze vlastností . . . 16
- 1.7 Příklady sítí . . . 17

2 Editor **18**
- 2.1 Systémové požadavky . . . 18
- 2.2 Rozložení editoru . . . 18
  - 2.2.1 Postranní panel . . . 19
  - 2.2.2 Hlavní plocha editoru . . . 20
  - 2.2.3 Panel nástrojů editoru . . . 20
  - 2.2.4 Tabulka ohodnocení . . . 20
  - 2.2.5 Výsledky analýzy . . . 20
  - 2.2.6 Ovládání . . . 20
  - 2.2.7 Tisk sítě . . . 20
  - 2.2.8 Klávesové zkratky . . . 20

3 Použité technologie **20**
- 3.1 nodejs . . . 20
- 3.2 Typescript . . . 20
- 3.3 Electron . . . 20
- 3.4 Javascriptová Knihovna Data driven documents (D3) . . . 20
- 3.5 Scalable Vector Graphics (SVG) . . . 21

4 Stavba programu **21**
- 4.1 Třída 1 . . . 21
- 4.2 Třída 2 . . . 21
- 4.3 Třída 3 . . . 21
- 4.4 Třída 4 . . . 21

5 Obsah přiloženého CD/DVD **21**

Seznam zkratek **23**

Literatura **24**

Rejstřík **25**

Seznam obrázků

1 síť 10
2 Ukázky jednoduchých sítí s grafem dosažitelnosti 13
3 Příklad neohraničené sítě 14
4 Příklad sítě z knihy Understanding petri nets[1](fig. 14.1) 16
5 Příklad neohraničené sítě s chybějící hranou v grafu pokrytí . . . 17
6 Rozložení editoru 18
8 Postranní panel 19

Seznam tabulek

Seznam vět

1 Definice (Petriho síť) 8
2 Definice (Graf dosažitelnosti) 11

Seznam zdrojových kódů

TODO: Smazat todo: command

1 Petriho sítě

Táto kapitola byla inspirovaná a čerpala informace z knihy Understanding petri nets[1]

1.1 Základní popis

Petriho síťe jsou matematickým nástrojem pro modelování a simulaci paralelních procesů a jejich sychronizaci. Jsou tvořené místy, přechody a hranami které jsou vždy propojením jednoho místa s jedním přechodem.

Definice 1 (Petriho síť)

$$N = \langle P, T, A, M_0 \rangle$$

- N je Petriho sítí
- P je konečná množina míst
- T je konečná množina přechodů
- A je konečná množina hran $A \subseteq ((P \times T) \cup (T \times P)) \times \mathbb{N}_0$
  kde číslo symbolizuje násobek kolik značek hrana „přesune“
- $M_0 : P \to \mathbb{N}_0$ je počáteční ohodnocení sítě (zkráceně ohodnocení) míst kde pro každé místo $p \in P$ existuje počet jeho značek $m \in M_0$

Pro odkazovaní na jednotlivé členy prvků z množiny hran budeme používat notaci $P(a)$ pro odkázání na místo v hraně $a \in A$, $T(a)$ pro odkázání na přechod a $AM(a)$ pro odkaz na násobek.

Každý přechod t může mít 'přiřazený' libovolný počet hran, kde každá hrana a je propojením s některým z míst $p \in P$.
Hrany přechodu t můžeme rozlišit na hrany směrující do přechodu

$$^{\rightarrow}t = \{a \in A | a \in (P \times T \times \mathbb{N}_0) \wedge t = T(a)\}$$

a hrany směřující z přechodu (do místa)

$$t^{\rightarrow} = \{a \in A | a \in (T \times P \times \mathbb{N}_0) \wedge t = T(a)\}$$

dohromady pak všechny hrany přechodu t jsou spojením těchto dvou množin

$$ArcesOfTransition(t, A) = (^{\rightarrow}t \cup t^{\rightarrow}) \subset A$$

Aktuální stav petriho sítě neboli ohodnocení M je funkce přiřazující každému místu $p \in P$ petriho sítě počet značek

$$(\forall p \in P) M(p) \in \mathbb{N}_0$$

počáteční ohodnocení petriho sítě se značí M_0.

Pro dané ohodnocení M je přechod $t \in T$ označený jako povolený pokud všechny hrany směřující do přechodu ${}^{\rightarrow}t$ splňují svou podmínku tzn. hrana splňuje svoji podmínku pokud místo ze kterého vychází má vyšší nebo stejné ohodnocení (v daném M) než je násobek hrany AM

$$IsEnabled(P, t, A, M) = (\forall a \in {}^{\rightarrow}t) M(P(a)) \geq AM(a)$$

Pak si můžeme ještě definovat množinu všech povolených přechodů pro zadané ohodnocení

$$EnabledTransitions(P, T, A, M) = \{t \in T | IsEnabled(P, t, A, M)\}$$

Pokud je přechod t v ohodnocení M Petriho sítě **povolen**, znamená to že může dojít k aktivování tohoto přechodu čímž dojde ke změně aktuálního ohodnocení z M do ohodnocení M' tak, že pro každé místo $p \in P$ a každou hranu $A_{pt} \subset ArcesOfTransition(t, A)$ spojující p s t že nové ohonocení v místě $M'(p)$ je sumou násobků hran $\sum_{a \in A_{pt}} AM(a)$ a původního hodnocení $M(p)$

$$FireTransition(P, t, A, M) = function\ M'$$

Výsledné ohodnocení M' je pak pro každé místo p definováno

$$M'(p) = M(p) + \sum_{a \in \{a_{tp} \in ArcesOfTransition(t,A) \,|\, P(a_{tp}) = p\}} AM(a)$$

Tuto změnu můžeme značit $M \rightarrow^t M'$

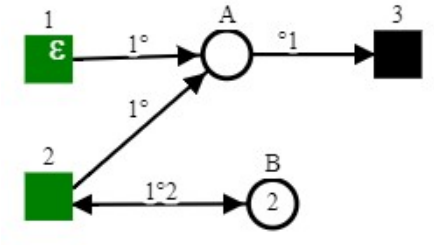
Ohodnocení M' je označené jako **dosažitelné**[{1}] z ohodnocení M pokud existuje sekvence přechodů taková, že jejich postupným aktivováním z ohodnocení M vznikne ohodnocení M'. Ohdonocení M' je dostupné z ohodnocení M pak značíme $M \rightarrow^* M'$.

1.2 Vizuální zobrazení sítě

Pro Petriho síť existuje nejenom matematické zobrazení ale i v praxi více využívané grafické zobrazení. V grafickém zobrazení kolečka symbolizují místa petriho sítě a číslo v kolečku udává počet značek. Přechody jsou symbolizované čtverci. V editoru je čtverec zelený pokud je přechod povolený. Pokud má v sobě čtverec symbol ϵ, znamená to že je přechod určený pro komunikaci sítě s okolním prostředím. Na editor nemá žádný vliv jestli je přechod označený jako ϵ a tak tedy toto označení je jen pro jednoduší orientaci v síti. Kolečka i čtverce mají pak nad sebou značení konkrétního místa/přechodu které symbolizují. Nakonec

samotné hrany jsou symbolizované šipkami které jsou popsané násobkem kolik hrana „přesune“ značek. Popis hran je symbolizován dvěmi čísly oddělenými kolečkem. Číslo před kolečkem značí hodnotu hrany směrující z přechodu do místa, číslo za kolečkem značí hodnotu hrany směrující z místa do přechodu.

Na obrázku 1 můžeme vidět grafické vyobrazení jednoduché sítě s dvěmi místy, třemi přechody a čtyřmi hranami. Již na první pohled si můžeme všimnou



Obrázek 1: Příklad zobrazení jednoduché sítě

že pro jednoduší rozlišení míst a přechodů jsou přechody značené čísli a místa písmeny. Tuto síť vyobrazenou na obrázku 1 bychom mohli matematickým zápisem zapsat jako síť $N = \langle P, T, A, M_0 \rangle$ kde
$P = \{a, b\}$
$T = \{1, 2, 3\}$
$A = \{\langle 1, a, 1\rangle, \langle 2, a, 1\rangle, \langle 2, b, 1\rangle, \langle b, 2, 2\rangle, \langle a, 3, 1\rangle\}$
$M_0(a) = 0; M_0(b) = 2$

V matematickém zápise sítě budeme místa značit malými písmeny (oproti editoru) abychom předešli případným nedorozuměním.

1.3 Zkrácený zápis ohodnocení sítě

Abychom předešli zdlouhavému psaní každého případu ohodnocovací funkce (např. $M(a) = 0; M(c) = 2; \cdots$), zavedeme si kratší zápis. Nejdříve seřadíme všechny místa podle jejich značení jako by šlo o číselnou soustavu (s písmeny místo čísel) neboli $a, b, c, \cdots, z, aa, ab, ac, \cdots$. Pak si z těchto míst uděláme uspořádanou n-tici jejich ohodnocení $\langle M(a), M(b), M(c), \cdots \rangle$. Tuto n-tici pak budeme používat jako kratší zápis ohodnocovací funkce:

$$M = \langle M(a), M(b), M(c), \cdots \rangle$$

Například pro $M'(a) = 3; M'(b) = 0; M'(c) = 5$ by krátký zápis vypadal
$M' = \langle 3, 0, 5 \rangle$

1.4 Využití Petriho sítí

Petriho sítě se používají k analýze a modelování paralelních a distribuovaných systému, databázových systémů atd. a to ať už pro analýzu při vývoji softwaru a nebo pro popis vnitřní struktury již hotového proprietárního softwaru umožnující lepší porozumění uživateli.

1.5 Graf dosažitelnosti

Graf dosažitelnosti je jeden z nejzákladnějších nástrojů pro analýzu Petriho sítí. Obsahuje vždy počáteční ohodnocení a všechny ohodnocení které jsou dostupné z počátečního ohodnocení, takovéto ohodnocení budeme nazývat **dosažitelné ohodnocení**[{2}]. Vrcholy grafu jsou jednotlivá ohodnocení a hrany grafu jsou značené přechody které jsou aktivované aby z počátečního ohodnocení vzniklo cílové.

Definice 2 (Graf dosažitelnosti)

$$RG = \{M, \langle M, T', M' \rangle\}$$

- RG je Graf dosažitelnosti
- M je Vrchol grafu který je zároveň konkrétní ohodnocení petriho sítě
- $\langle M, T', M' \rangle$ je Hrana grafu která je změnou z hodnocení M libovolným přechodem $t \in T'$ ze kterého vzniká M'

1.5.1 Vlastnosti odvoditelné z Grafu dosažitelnosti

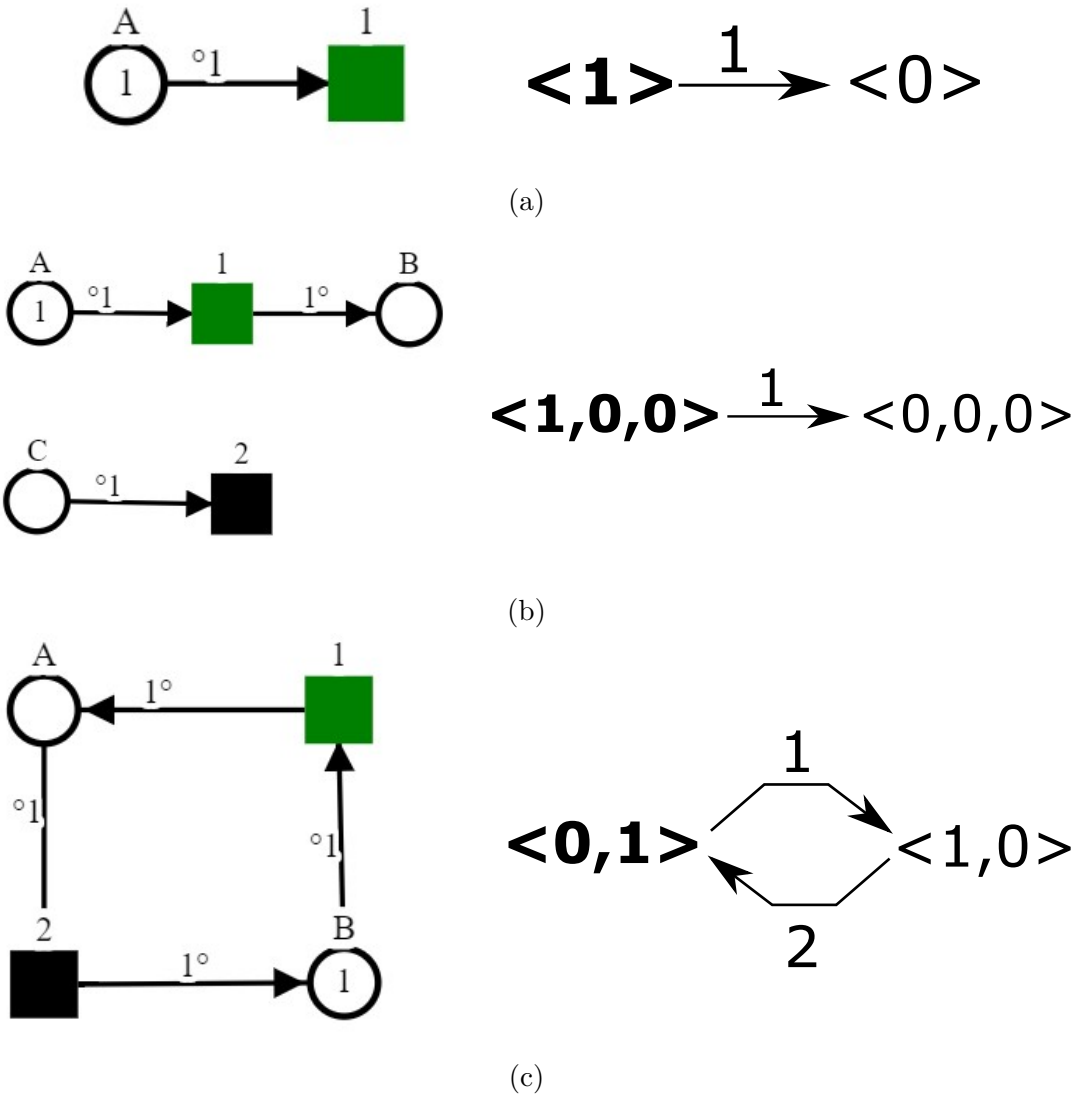
Z grafu dosažitelnosti Petriho sítě jsou odvoditelné tyto vlastnosti:

- Petriho síť je **ohraničená**[{3}] pokud je její graf dosažitelnosti konečný. Pokud existuje takové přirozené číslo n pro které v žádném dosažitelném ohodnocení nepřesahuje žádné místo svým ohodnocením číslo n. Pokud zvolíme n aby splňovalo tuto podmínku a zároveň bylo nejmenší možné, pak můžeme nazývat síť že je **ohraničená** n.
- Petriho síť **skončí**[{4}] za předpokladu že graf je konečný a zároveň neobsahuje žádné cykly. Neboli Petriho sít vždy po nějakém počtu kroků dojde do stavu kdy žádný přechod není povolený.
- Petriho síť je **vratná**[{5}] pokud je její graf silně souvislý. Z každého dosažitelného ohdonocení je dosažitelné počáteční ohodnocení.
- Petriho síť je **bez mrtvého bodu**[{6}] pokud z každého vrcholu grafu vede alespoň jedna hrana. Petriho sít má v každém ohodnocení povolený alespoň jeden přechod.
- Petriho síť je **slabě živá**[{7}] pokud pro každý přechod existuje v grafu hrana označená tímto přechodem. Pro každý přechod Petriho sítě existuje dosažitelné ohodnocení které povoluje daný přechod.
- Petriho síť je **živá**[{8}] pokud pro každý přechod t a každé ohodnocení M existuje v grafu cesta z ohodnocení M do ohodnocení ze kterého vede hrana z označením přechodu t. Pro každý přechod t a každé ohodnocení M existuje dosažitelné ohodnocení M' které přechod t povoluje.

Logickou úvahou pak můžeme určit někré vzájemné závislosti vlastnosti:

- Síť která je **vratná** nebo/a **živá** je zároveň i **bez mrtvého bodu**.
- Síť která je **bez mrtvého bodu** ne**skončí** a zároveň síť která **skončí** není **bez mrtvého bodu**(pozor, neznamená že síť musí mít alespoň jednu z těchto vlastností).
- Síť která není **slabě živá** nemůže být **živá**.

1.5.2 Příklady grafu dosažitelnosti s vlastnostmi



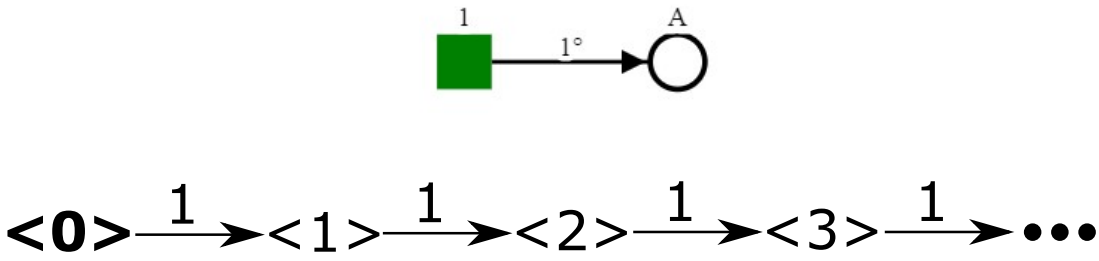
Obrázek 2: Ukázky jednoduchých sítí s grafem dosažitelnosti

Síť na obrázku 2a **skončí** a je **slabě živá**.
Síť na obrázku 2b **skončí** a není **slabě živá**.
A síť na obrázku 2c je **vratná** a **živá**.

Všechny tři sítě jsou ohraničené 1. Také si můžeme všimnout že z výše uvedených vlastností u ukázkových sítí stačí jen vypsané vlastnosti a ostatní se dají odvodit ze závislosti vlastností.

1.6 Graf pokrytí

Hlavní nevýhodou grafu dosažitelnosti je že může být nekonečný a tudíž je nemožné ho zkonstruovat celý. Můžeme velice jednoduše navrhnout a sestrojit triviální Petriho síť (Obrázek 3) u které by konstrukce jejího grafu dosažitelnosti nikdy neskončila.



Obrázek 3: Příklad neohraničené sítě

Proto existuje upravená verze grafu dosažitelnosti nazvaná grafu pokrytí, která může obsahovat tzv. ω ohodnocení které mimo celých čísel přiřadí alespoň jednomu místu i hodnotu ω symbolizující že místo může nabívat nekonečně vysokého počtu značek. Petriho síť se nemůže nacházet v ω ohodnocení, toto ohodnocení je pouze pro vytvoření abstrakce v grafu pokrytí.

Protože hodnotu ω bereme jako nekonečno pak od ní můžeme odečíst nebo přičíst libovolně velké číslo a hodnota se nezmění.

$$\cdots = \omega - 2 = \omega - 1 = \omega = \omega + 1 = \omega + 2 = \cdots$$

Ohodnocení M značíme jako že je ostře menší $<$ než ohodnocení M' pokud pro každé místo p platí $M(p) \leq M'(p)$ a alespoň pro jedno místo p platí $M(p) < M'(p)$.

$$M < M' = ((\forall p \in P) M(p) \leq M'(p)) \wedge (\exists p \in P) M(p) < M'(p)$$

1.6.1 Sestrojení grafu

Sestrojování grafu probíhá postupně přidáváním hran. Nejdříve se přidá počáteční ohodnocení jako kořen grafu. Následně se z grafu vybírají náhodně nevypočítané povolené přechody a pokud vedou do místa které ještě v grafu není tak se přidá a pokud je ostře menší než ohodnocení ze kterého je dosažitelné tak se přidají ω hodnoty na místa ve kterých má více značek. Algoritmus končí výpočet až jsou všechny povolené přechody pro všechny vrcholy v grafu vypočítané.

Sestrojení grafu pokrytí pseudokód MakeCoverabilityGraph1

Pokud sestrojený graf pokrytí neobsahuje žádné ω ohodnocení, pak je graf totožný s grafem dosažitelnosti. Pokud graf pokrytí obsahuje ω ohodnocení, zanamená to že graf dosažitelnosti by byl nekonečný a tudíž by nebylo možné ho zkonstruovat celý a nešli by na něm zjišťovat některé nebo všechy vlastnosti. Proto si vystačíme s algoritmem na vytváření grafu pokrytí.

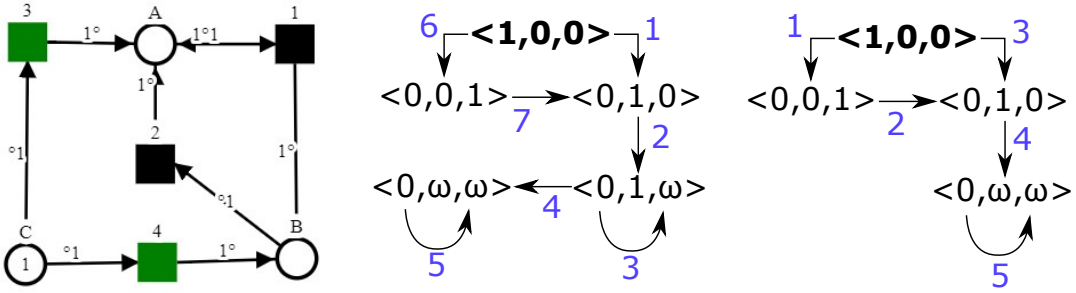
Algorithm 1 MakeCoverabilityGraph

```
1: function MakeCoverabilityGraph(⟨P, T, A, M_0⟩)
2:     ⟨V, E, v_0⟩ := ⟨{M_0}, ∅, M_0⟩
3:     WorkSet := ∅
4:     for all t ∈ EnabledTransitions(P, T, A, M_0) do
5:         WorkSet := WorkSet ∪ {⟨M_0, t⟩}
6:     end for
7:     while WorkSet ≠ ∅ do
8:         ⟨M, t⟩ := RandomElement(WorkSet)
9:         WorkSet := WorkSet \ {⟨M, t⟩}
10:        M' := FireTransition(P, t, A, M)
11:        for all {M'' | M'' ∈ V ∧ (M'' →* M ∨ M'' = M) ∧ M'' < M'} do
12:            for all p ∈ P do
13:                mp := M(p)
14:                if M''(p) < M'(p) then
15:                    M'(p) := ω
16:                end if
17:            end for
18:        end for
19:        if M' ∉ V then
20:            V := V ∪ {M'}
21:            for all t ∈ EnabledTransitions(P, T, A, M') do
22:                WorkSet := WorkSet ∪ {⟨M', t⟩}
23:            end for
24:        end if
25:        E := E ∪ {⟨M, t, M'⟩}
26:    end while
27:    return ⟨V, E, v_0⟩
28: end function
```

1.6.2 Různé výsledky grafu pokrytí

Při konstrukci grafu pokrytí záleží v jakém pořadí se hrany přidávají a výsledný graf může mít různý počet vrcholů a hran v závislosti na pořadí přidávání hran. V našem algoritmu využíváme funkce *RandomElement* která vybere náhodný prvek množiny a snažíme se tak tipovat jaké pořadí hran bude ideální pro sestrojení nejmenšího grafu. Pokud bychom chtěli sestrojit minimální graf pokrytí, museli bychom nahradit funkci *RandomElement* nějakou funkcí která by vždy vybrala přechody právě v takovém pořadí aby došlo k sestrojení minimálního grafu.

Že záleží na pořadí v jakém se hrany přidávají si můžeme ukázat na síti v obrázku 4. Zde ale musíme dávat pozor, protože v tomto případě čísla neznačí přechody, ale pořadí ve kterém byly hrany přidány.



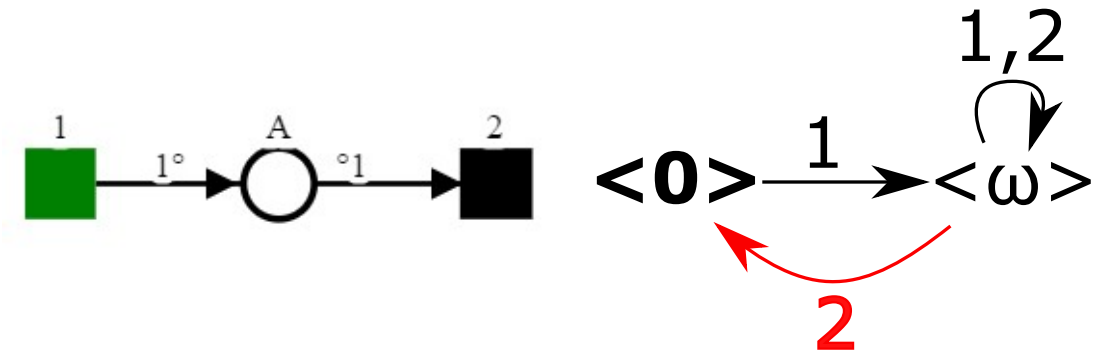
Obrázek 4: Příklad sítě z knihy Understanding petri nets[1](fig. 14.1)

1.6.3 Upravená verze vlastností

Oproti grafu dosažitelnosti náš graf pokrytí tak jak jsme ho sestrojili pomocí algoritmu MakeCoverabilityGraph1 nemusí obsahovat všechny výsledky přechodů které mohou nastat a to znamená že v některých případech některé vlastnosti Petriho sítě nejsme schopni určit, protože nám chybí informace o těchto chybějících hranách grafu pokrytí. Problém je částečně způsobený tím jak máme definovanou hodnotu ω a pokud nějaké místo p je ohodnoceno ω pak už není možné aby z vrcholu s tímto ohodnocením vedla hrana do vrcholu kde místo p nebude mít hodnotu ω.

V Obrázku 5 vidíme červěně zvíražněnou hrana která při použití algoritmu MakeCoverabilityGraph1 chybí. Přitom by tam měla hrana být, protože když budeme neustále opakovat aktivaci přechodu 2 tak se eventuelně dostaneme do ohodnocení kde má místo A hodnotu 0 což je zároveň výchozí ohodnocení. Díky této chybějící hraně bychom síť určili jako že není **vratná** ale přitom ve skutečnosti je. Proto musíme zjistit jestli jsou všechny vlastnosti grafu dosažitelnosti aplikovatlné i na graf pokrytí, případně poupravit nebo rozšířit jejich definici.

Ohraničenost sítě je určená konečností grafu dosažitelnosti. Nekonečný rozvoj je



Obrázek 5: Příklad neohraničené sítě s chybějící hranou v grafu pokrytí

1.7 Příklady sítí

TODO: síť + analýza v programu

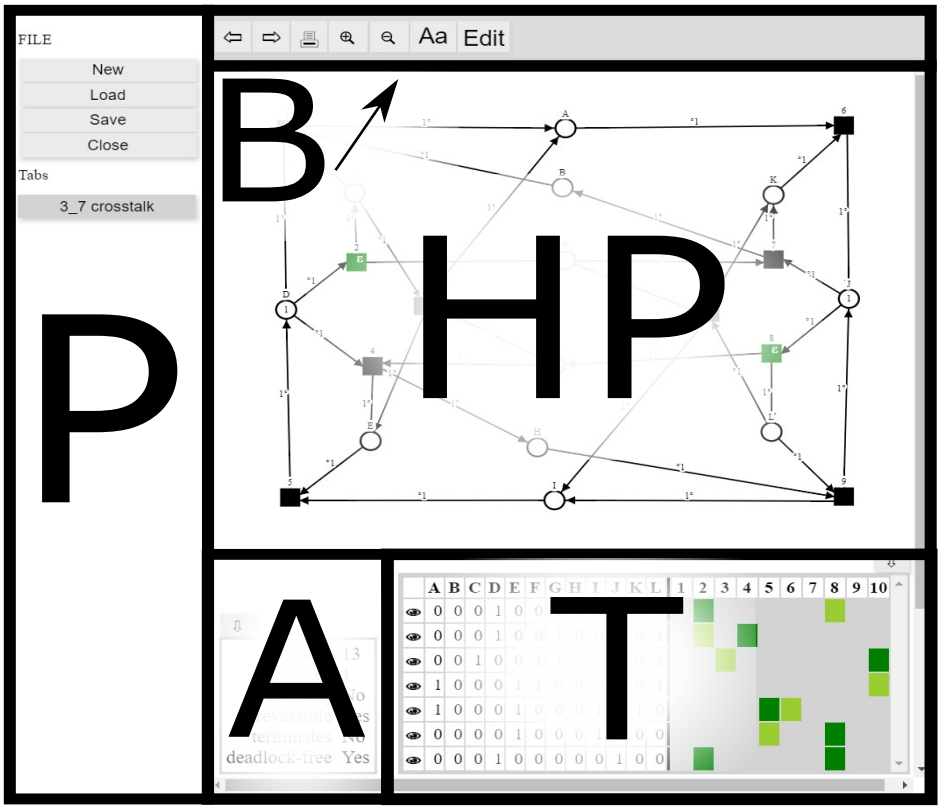
2 Editor

2.1 Systémové požadavky

- Operační systém: Windows 10 (starší verze windows netestovány)
- Ovládání: klávesnice+myš

2.2 Rozložení editoru

Editor je rozložený na několik částí. Všechny tyto části jsou písmeny označené v Obrázku 6 Rozložení editoru.Každá část editoru je popsaná ve své vlastní sekci.



Obrázek 6: Rozložení editoru

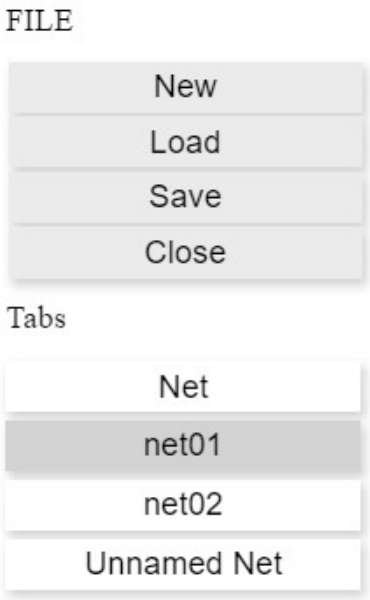
označení	název části editoru	sekce
P	Postranní panel	2.2.1
HP	Hlavní plocha editoru	2.2.2
B	Panel nástrojů editoru	2.2.3
T	Tabulka ohodnocení	2.2.4
A	Výsledky analýzy	2.2.5

2.2.1 Postranní panel

Postraní panel obsahuje tlačítka pro práci se záložkami a samotné záložky. Pod označením **FILE** jsou tlačítka:

New	Vytvoří novou prázdnou záložku
Load	Otevře dialog pro načtení uložené sítě
Save	Uloží síť v otevřené záložce. Pokud byla síť již uložena nebo načtena dialog nabídne uživateli na výběr dvě možnosti. Možnost *yes* uloží a přepíše původní soubor. Možnost *Select file* otevře dialog a uživatel vybere vlastní místo uložoní.
Close	Zavře aktuálně otevřenou záložku

Pod označením **Tabs** jsou pak záložky které se otevírají kliknutím. Záložky jsou nazvané podle jména souboru sítě.



Obrázek 8:
Postranní panel

2.2.2 Hlavní plocha editoru

2.2.3 Panel nástrojů editoru

2.2.4 Tabulka ohodnocení

2.2.5 Výsledky analýzy

2.2.6 Ovládání

2.2.7 Tisk sítě

TODO: obrázek jak vytisknotu do PDF

2.2.8 Klávesové zkratky

3 Použité technologie

3.1 nodejs

Aplikace je psaná za pomoci opensourcové technologie nodeJS, která umožňuje využívat jazyk JavaScript pro psaní serverových aplikací. Cílem platformy nodeJS je vytvořit ekosystém pro jednoduší vývoj webových stránek a aplikací kde stačí pro vyrvaření funkcionality pouze jeden programovací jazyk.

3.2 Typescript

Typescript je opensource programovací jazyk od společnosti Microsoft který je nadstavbou nad jazykem JavaScript. Jelikož je Typescript nadstavbou nad programovacím jazykem JavaScript tak je jakýkoliv validní kód v JavaScriptu automaticky validním kódem v Typescriptu. Typecript se kompiluje do Javascriptu a proto po stránce funkcionality nenabízí nic navíc avšak po stránce vývoje nabízí možnost statické typové kontroly se kterou je spjaté fungování našeptávačů v dnešních textových editorech a také nabízí možnost kompilace do starších verzí JavaScript se simulací funkcionality novejších verzí JavaScriptu. TODO: příklady kódu ?

3.3 Electron

Electron je opensource framework pro vytváření desktopových aplikací pomocí webových technologií JavaScript, HTML a CSS. Využívá NodeJS

3.4 Javascriptová Knihovna Data driven documents (D3)

TODO: příklady kódu ?

3.5 Scalable Vector Graphics (SVG)

TODO: příklady kódu ?

4 Stavba programu

4.1 Třída 1

4.2 Třída 2

4.3 Třída 3

4.4 Třída 4

5 Obsah přiloženého CD/DVD

Na samotném konci textu práce je uveden stručný popis obsahu přiloženého CD/DVD, tj. jeho závazné adresářové struktury, důležitých souborů apod.

`bin/`
: Instalátor Instalator programu, popř. program Program, spustitelné přímo z CD/DVD. / Kompletní adresářová struktura webové aplikace Webovka (v ZIP archivu) pro zkopírování na webový server. Adresář obsahuje i všechny runtime knihovny a další soubory potřebné pro bezproblémový běh instalátoru a programu z CD/DVD / pro bezproblémový provoz webové aplikace na webovém serveru.

`doc/`
: Text práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu, vložené obrázky, apod.

`src/`
: Kompletní zdrojové texty programu Program / webové aplikace Webovka se všemi potřebnými (příp. převzatými) zdrojovými texty, knihovnami a dalšími soubory potřebnými pro bezproblémové vytvoření spustitelných verzí programu / adresářové struktury pro zkopírování na webový server.

`readme.txt`
: Instrukce pro instalaci a spuštění programu Program, včetně všech požadavků pro jeho bezproblémový provoz. / Instrukce pro nasazení webové aplikace Webovka na webový server, včetně všech požadavků pro její bezproblémový provoz, a webová adresa, na které je aplikace nasazena pro účel testování při tvorbě posudků práce a pro účel obhajoby práce.

Navíc CD/DVD obsahuje:

`data/`
: Ukázková a testovací data použitá v práci a pro potřeby testování práce při tvorbě posudků a obhajoby práce.

`install/`
: Instalátory aplikací, runtime knihoven a jiných souborů potřebných pro provoz programu Program / webové aplikace Webovka, které nejsou standardní součástí operačního systému určeného pro běh programu / provoz webové aplikace.

`literature/`
: Vybrané položky bibliografie, příp. jiná užitečná literatura vztahující se k práci.

U veškerých cizích převzatých materiálů obsažených na CD/DVD jejich zahrnutí dovolují podmínky pro jejich šíření nebo přiložený souhlas držitele copyrightu. Pro všechny použité (a citované) materiály, u kterých toto není splněno a nejsou tak obsaženy na CD/DVD, je uveden jejich zdroj (např. webová adresa) v bibliografii nebo textu práce nebo v souboru `readme.txt`.

Literatura

[1] REISIG, Wolfgang. *Understanding petri nets : modeling techniques, analysis methods, case studies*. Berlin: Springer, 2013. ISBN 978-3-642-33277-7.

Rejstřík

dosažitelné ohodnocení, 9, 11
síť živá, 11
síť bez mrtvého bodu, 11
síť ohraničenost, 11
síť skončí, 11
síť slabě živá, 11
síť vratná, 11