VŠB – Technická Univerzita Ostrava
Fakulta Elektrotechniky a Informatiky

DYNAMICKÉ SPOLEHLIVOSTNÍ MODELY NA BÁZI PETRIHO SÍTÍ

DIZERTAČNÍ PRÁCE

2009 Petra Škňouřilová

Dynamické spolehlivostní modely na bázi Petriho sítí
Dizertační práce

Ing. Petra Škňouřilová
VŠB–TU Ostrava
petra.sknourilova@vsb.cz

Školitel:
Doc. Ing. Radim Briš, CSc.
VŠB–TU Ostrava
radim.bris@vsb.cz

Obor:
Technická kybernetika

Fakulta Elektrotechniky a Informatiky
VŠB – Technická Univerzita Ostrava
17. listopadu 15
708 33 Ostrava–Poruba
Česká Republika

http://fei.vsb.cz
http://www.vsb.cz

Typeset by PDFLaTeX

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

NASA	Národní úřad pro letectví a kosmonautiku
F(t)	distribuční funkce
f(t)	funkce hustoty
h(t)	intenzita poruch
MTTF	střední doba do poruchy
MTTR	střední doba do obnovy
R(t)	spolehlivost
A(t)	použitelnost
U(t)	nepohotovost
ŘO	řídící/ochranná jednotka
PN	Petriho síť
GSPN	Zobecněná Stochastická Petriho síť
CPN	Barevná Petriho síť
CET	spojitý strom událostí
CCCM	metoda cell-to-cell zobrazení
MC	simulace Monte Carlo
DFM	dynamické síťové grafy
ESD	diagram sekvence událostí
FMEA	metoda analýzy projevů a důsledků poruch
PRA	pravděpodobnostní analýza rizika
CDF	kumulativní distribuční funkce

Poděkování

Na tomto místě bych ráda poděkovala mému příteli Josefovi za pomoc a podporu během studia. Za hodnotné rady a péči, kterou mi věnoval v průběhu mého doktorandského studia, děkuji mému školiteli Doc.Ing. R.Brišovi,CSc.. Můj nemalý dík náleží i Prof. E.Chateletovi (Univ. of Troyes, Francie), který mi věnoval čas na konzultace a také za umožnění pobytu na výše zmíněné univerzitě. Mé poděkování patří i vedoucímu Katedry matematiky a deskriptivní geometrie VŠB-TUO Doc.RNDr. P.Burdovi, CSc..

Petra Škňouřilová
Ostrava, Česká Republika
31. března 2009

Není běžnějšího omylu než věřit, že když provedeme dlouhé a přesné matematické výpočty, je pak aplikace výsledku na nějaký fakt v přírodě absolutně jistá.

Alfred North Whitehead

.

Kapitola 1

Úvod

Disciplína dynamické spolehlivosti vznikla jako prostředek ke studiu interakcí strojově - softwarových systémů, kdy nebereme v úvahu pouze stochastické nezdary, ale také deterministický vývoj procesních proměnných. Příklady dynamických aspektů, které bychom měli uvážit jsou: zásah vhodně navržených řídících/ochranných systémů v situaci, kdy jedna nebo více proměnných překročí nastavenou prahovou hodnotu s čímž je spojené selhání; vliv vývoje procesních proměnných na stochastické chování systému (např. zvýšení četnosti poruch komponent v důsledku růstu zátěže komponent během nehod); atd..

Je nutno zdůraznit, že v případě reálných systémů bude dynamický přístup k analýze spolehlivosti vyžadovat významné zvýšení výkonu z důvodu přihlédnutí k vývoji procesních proměnných, lidským operátorům a řídícím akcím. V posledních letech umožnil zvýšený počítačový výkon zahrnutí dynamického chování bezpečnostních a spolehlivostních modelů. Obzvláště využití metody Monte Carlo poskytlo možnost efektivně odhadnout spolehlivost systémů obsahujících dynamické prvky. Základy této metody mohou být nalezeny v (Marseguerra 1996).

Dynamická spolehlivost rozšiřuje klasickou metodu stromu poruch/událostí tím způsobem, že bere v úvahu vzájemný vztah mezi hardwarovými komponentami a fyzickým vývojem procesních proměnných. Dynamické aspekty se týkají uspořádání a časování událostí v případě nehody (poruchy), lidských činitelů a řídících akcí. Metody dynamické spolehlivosti jsou založeny na silném matematickém základě schopné zahrnout interakce mezi komponentami a prostředím, ve kterém pracují. Tyto metody provádějí více realistické modelování systému a dále vylepšují kvalitu a přesnost studií zabývajících se hodnocením rizik. Formální přístup k zahrnutí dynamického chování systému byl v analýze rizik zformulován pod názvem Pravděpodobnostní Dynamika (Devooght 1992b). Během posledních deseti let bylo formulováno několik metod pro řešení problému dynamické spolehlivosti (Marseguerra 1996, Cojazzi 1992, Aldemir 1994b, Siu 1994b, Izquierdo 1994, Labeau 1996b).

Cíle výzkumu

Jedním z hlavních problémů v posouzení rizika dynamických systémů je vztah mezi fyzickými parametry (např. teplota, tlak, výška hladiny, . . .) a stavy komponent. Formální popis a výpočet spolehlivosti těchto systémů je ve skutečnosti složitým problémem. Jeden z důvodů spočívá v implementaci dynamického chování. Metody dynamické spolehlivosti neposkytují pro znázornění systémů všeobecně použitelné schéma. Jednou z možností, jak dynamické systémy modelovat jsou Petriho sítě (PN).

Prvním cílem mého výzkumu je aplikovat různé modifikace Petriho sítí (GSPN, CPN) na speciálně zkonstruovaný dynamický problém formulovaný v literatuře a následně vytvořené Petriho sítě využít jako vstupní schéma k simulaci. Získané výsledky budou konfrontovány s přesným analytickým řešením či dalšími použitými metodami pro zjištění, zda zvolený způsob řešení je vhodný či dokonce vhodnější k řešení dynamické spolehlivosti. Dalším krokem je modifikace dynamického systému. První modifikací je zahrnutí akce údržby či opravy do analyzovaného problému a druhou modifikací je rozšíření problému o druhou procesní proměnnou.

Druhým cílem mé práce je využití získaných znalostí a zkušeností z první části k namodelovaní konkrétního reálného systému.

Struktura práce

Z výše uvedených cílů odpovídá i struktura mé dizertační práce. Kapitola 2 se věnuje teorii dynamické spolehlivosti. Kapitola 3 popisuje čtyři techniky využívané k tvorbě modelů pro posouzení spolehlivosti systémů. Čtvrtá kapitola obsahuje výčet a popis metod pro modelování spolehlivosti dynamických procesů. V páté kapitole je stručné shrnutí teorie Petriho sítí. Kapitola 6 a 7 je hlavní částí této práce a týká se aplikace Petriho sítí na zvolené problémy: v šesté kapitole se věnuji řešení benchmarku a v sedmé řešení reálného problému pomocí Petriho sítí. Diskutovány jsou výsledky a závěry vyplývající z dosažených výsledků.

Kapitola 2

Dynamická spolehlivost

Pravděpodobnostní metoda hodnocení rizik, která je také nazývána Kvantitativní analýza rizika, byla aplikována před více než třiceti lety na rozsáhlých komplexních systémech. První aplikací této metody v plném rozsahu bylo studium bezpečnosti reaktoru WASH-1400 (NRC 1975).

Metody hodnocení rizika byly také využity v jiných průmyslových sektorech a armádě. Po rozsáhlém revidování bezpečnostní politiky NASA, které následovalo po nehodě raketoplánu Challenger v roce 1986, NASA využila množství programů kvantitativní analýzy rizika. Příkladem je hodnocení rizika v programu vesmírných letů (Fragola 1995). Poté Úřad pro bezpečnost a zabezpečení mise ve vedení NASA vydal několik příruček pro rozšíření kvalifikace hodnocení rizika v NASA (Stamatelatos 2002).

V některých oblastech jsou techniky hodnocení rizika součástí regulačního frameworku. V situacích, kdy je řízení rizika kritické pro úspěch mise, hrají metody hodnocení rizika důležitou roli při rozhodování a řízení.

Metoda hodnocení rizika se snaží odpovědět na tři otázky položené v (Kaplan 1981), které mohou být reprezentovány množinou trojic: ”scénáře - frekvence - následky”.

Klasický přístup metody hodnocení rizika zahrnuje konstrukci samostatných modelů popisující zranitelnost systému a rizika. Modely jsou typicky prezentovány ve formě stromů poruch/událostí, které jsou grafickou reprezentací Booleovských výrazů popisující kombinace tzv. základních událostí vedoucích k selhání systému. Základní události obecně reprezentují selhání některých komponent nebo subsystémů.

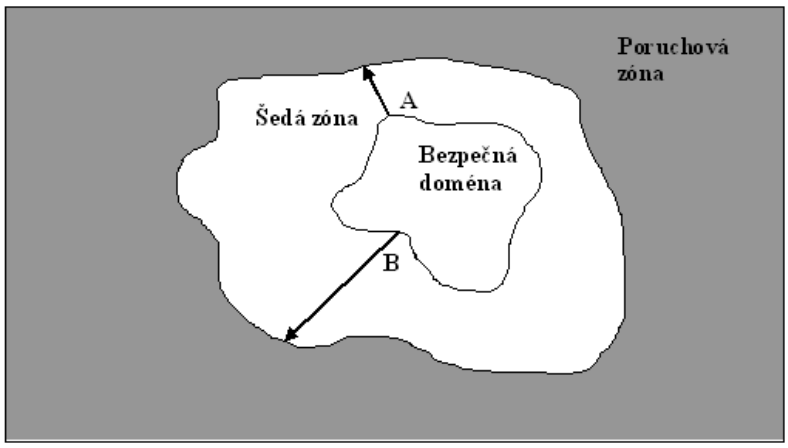
2.1 Teorie dynamické spolehlivosti

2.1.1 Hlavní předpoklady

Fyzický systém, jako například jaderný reaktor, je charakterizován jeho subsystémy a množinou fyzických proměnných (tlak, teplota, atd.). Vektor procesních proměnných je označen $\bar{x}$. Dynamika systému může být popsána soustavou diferenciálních rovnic

$$\frac{d\bar{x}}{dt} = \bar{f}_i(\bar{x}, t). \tag{2.1}$$

Analýza události slouží ke zjišťování podmínek, ve kterých vektor $\bar{x}$ opouští bezpečnou doménu (Obrázek 2.1) a případně k počítání pravděpodobnosti takové události.



Obrázek 2.1: *Nástin oblastí při vývoji procesních proměnných.*

Nehoda se může stát z důvodu, že $\bar{x}$ opustí hranici domény (např. vinou lidského činitele nebo z důvodu selhání komponent). Dynamiky jsou přímo ovlivněny stavem jednotlivých komponent. Přestože selhání některých komponent přímo neovlivní dynamiku, zavádí se index $i = 1, ..., m^N$, kde každá z N komponent systému se může nacházet v jednom z m různých stavů. Jako obvykle může být jakákoliv explicitní časová závislost eliminována pomocí doplňkové proměnné. Historie změn v systému je sled stavů $(\bar{x}_1, i_1, t_1), (\bar{x}_2, i_2, t_2), \ldots, (\bar{x}_k, i_k, t_k), \ldots$, kde v čase přechodu t_k je systém ve stavu $(\bar{x}_k, i_k)$. Mezi t_k a t_{k+1} je vektor $\bar{x}(t)$ řešením

$$\frac{d\bar{x}}{dt} = f_k(\bar{x})$$

s $\bar{x}_k(t_k) = \bar{x}_{k+1}(t_k)$.

Přechod $i_k \rightarrow i_{k+1}$ je chápán jako okamžitý. Tento předpoklad je obvykle dostačující. Změny stavů i nastanou kvůli

1. selhání či poruchy komponenty
2. kontrolním zařízením, které fungují pod vlivem vektoru $\bar{x}$
3. lidskému zásahu.

Pro změnu stavů platí Markovský předpoklad: budoucí vývoj systému závisí pouze na současné hodnotě $(\bar{x}, i)$ a ne na minulosti systému. Je důležité také uvážit lidské

faktory, to znamená zahrnout do stavu systému rozšíření o "stav" operátora. Je důležité si uvědomit, že jakékoliv modelování operátora je v konfliktu s Markovským předpokladem. Provozní čas má exponenciální rozdělení, které ale obvykle nepopisuje reakční dobu operátora.

2.1.2 Matematická formulace

Matematická formulace problému dynamické spolehlivosti byla prvně zmíněna v (Devooght 1992b) a později rozšířená v (Labeau 1996b, Devooght 1992a, Devooght 1996, Izquierdo 2004, Izquierdo 1996, Labeau 1996a). Cílem dynamické spolehlivosti je nalézt pravděpodobnostní funkci hustoty $\pi(\bar{x}, i, t)$, která vyjadřuje pravděpodobnost nalezení systému v bodě $\bar{x}$, ve stavu i a v čase t.

$$\lambda_i(\bar{x})\pi(\bar{x},i,t) = \lambda_i(\bar{x}) \int \pi(\bar{u},i,0)\delta(\bar{x}-\bar{g}_i(t,\bar{u}))(1-F_i(t,\bar{u}))d\bar{u} + \\ + \sum_{j\neq i} \int_0^t \int \lambda_j(\bar{u})\pi(\bar{u},j,t-\tau)[\frac{p(j\rightarrow i|\bar{u})}{\lambda_i(\bar{u})}] \times \delta(\bar{x}-\bar{g}_i(\tau,\bar{u}))dF_i(\tau,\bar{u})d\bar{u}, \quad (2.2)$$

kde $F_i(t,\bar{x})$ je pravděpodobnost, že systém opustí stav (i) před nastáním času t. V Markovském přístupu je pravděpodobnost dána

$$F_i(t,\bar{x}) = 1 - exp[-\int_0^t \lambda_i[\bar{g}_i(s,\bar{x})]ds], \quad (2.3)$$

kde $\bar{g}_i(t,\bar{x}_0)$ je řešení i-té dynamiky

$$\frac{d\bar{x}}{dt} = \bar{f}_i(\bar{x}) \quad (2.4)$$

s počáteční podmínkou $\bar{x}(0) = \bar{g}_i(0,\bar{x}_0) = \bar{x}_0$.

Nyní lze rovnici (2.2) přepsat do tvaru systému parciálních diferenciálních rovnic, tj. do tvaru Chapman-Kolmogorovových rovnic (Gardiner 1985):

$$\frac{\delta}{\delta t}\pi(\bar{x},i,t) + div(\bar{f}_i(\bar{x})\pi(\bar{x},i,t)) + \lambda_i(\bar{x})\pi(\bar{x},i,t) - \sum_{j\neq i}(j\rightarrow i|\bar{x})\pi(\bar{x},j,t) = 0, \quad (2.5)$$

kde λ_i je celková přechodová rychlost ze stavu i taková, že

$$\sum_{j\neq i} p(i\rightarrow j|\bar{x}).$$

Integrální formulace poskytuje semi-Markovské rozšíření (Devooght 1996) a jednotné zpracování přechodů na požádání či přechody v čase. Izquierdo s kolektivem rozšířil tuto teorii o vysvětlení "podnětu", který může spustit automatické či manuální akce (Izquierdo 2004). Stavový prostor obsahuje další rozšíření - bereme v úvahu aktivační stavy podnětu. Podnětem je buď příkaz pro akci od operátora či zařízení automatické kontroly anebo splnění podmínek, které spouští stochastickou událost.

Důležitý fakt, zavedený v rovnicích (2.2) až (2.5), je interakce mezi dynamikami a systémovou strukturou: přechody mezi stavy (selhání nebo oprava subsystému) jsou ovlivněny stavovými proměnnými, které naopak mají dynamiky ovlivněné strukturou (tj. stavem) systému.

V praxi může být zajímavá subdynamika ve smyslu statistické mechaniky:

$$\pi(\bar{x}, t) = \sum_i \pi(\bar{x}, i, t) \tag{2.6}$$

nebo

$$\pi(i, t) = \int \pi(\bar{x}, i, t) d\bar{x}. \tag{2.7}$$

Druhá rovnice je vhodná pro všechny problémy, kde procesní proměnné mají malý nebo žádný vliv na rychlost přechodu, což je běžná situace ve většině spolehlivostních problémů. Avšak neměly bychom zapomínat, že obvykle máme nedokonalou znalost systému, ať už kvůli jeho dynamikám nebo z důvodu jeho přechodových rychlostí. Tato nejistota může být obsažena v parametrech a_i, jejichž rozdělení $p(\bar{a})$ je obvykle získáno z odvození názoru experta. Systémové rovnice (2.2) a (2.5) mají řešení, které je ve skutečnosti podmíněnou pravděpodobností $\pi(\bar{x}, i, t|\bar{a})$ při změně $f_i(\bar{x})$ na $f_i(\bar{x}, \bar{a})$, $\lambda_i(\bar{x})$ na $\lambda_i(\bar{x}, \bar{a})$, atd.. Proto chceme předvídat subdynamiky ve vztahu k

$$\pi(\bar{x}, i, t) = \int \pi(\bar{x}, i, t|\bar{a}) p(\bar{a}) d\bar{a}. \tag{2.8}$$

Parametry $\bar{a}$ by mohly být samotné přechodové rychlosti.

2.1.3 Objasnění hlavních aspektů

Předpokládejme, že systém je v ustáleném stavu $i = 1, \bar{x}_0$, dokud nedojde v čase $t = 0$ k přechodu ze stavu 1 do stavu i, tj. $\bar{x}(t) = \bar{g}_i(t, \bar{x}_0)$. Druhý přechod má za úkol ukončit tuto změnu stavu systému. Z počáteční podmínky:

$$\pi(\bar{u}, i, 0) = \delta(\bar{u} - \bar{x}_0) \tag{2.9}$$

dostaneme z rovnice (2.2)

$$\pi(\bar{x}, i, t) = \delta(\bar{x} - \bar{g}_i(t, \bar{x}_0)).e^{-\int_0^t \lambda_i[\bar{g}_i(s,\bar{x_0})]ds}. \tag{2.10}$$

Následkem $\bar{x}(t) = \bar{g}_i(t, \bar{x}_0)$ ze stavu $\bar{x}_0$ v čase $t = 0$, definujme čas $t_c = t_c(i, \bar{x}_0)$ jako čas prvního přechodu přes bezpečnou hranici Γ, tj. minimální hodnota t taková, že $\bar{g}_i(t, \bar{x}_0) \in \Gamma$. Pokud nenastane situace přechodu přes hranici, je čas $t_c = \infty$. Bezpečná doména D je ohraničená hranicí Γ a její doplněk je $\bar{D}$ s $D \cap \bar{D} = \emptyset$.

Nyní z rovnic (2.2) a (2.10) dostaneme

$$\pi(\bar{x}, j, t) = \int_0^t \int \pi(\bar{u}, i, t - \tau)p(i \to j|\bar{u})\delta(\bar{x} - \bar{g_j}(\tau, \bar{u}))e^{-\int_0^\tau \lambda_i|\bar{g_j}(s,\bar{u})|ds} d\bar{u}d\tau. \tag{2.11}$$

Dosazením (2.10) do (2.11):

$$\begin{aligned}\pi(\bar{x}, j, t) = \int\int \delta(\bar{u} - \bar{g}_i(t - \tau, \bar{x}_0))e^{-\int_0^t \lambda_i|\bar{g}_i(s,\bar{x}_0)|ds} p(i \to j|\bar{u})\delta(\bar{x} - \bar{g}_j(\tau, \bar{u}))\\ e^{-\int_0^\tau \lambda_i|\bar{g}_j(s,\bar{u})|ds} d\bar{u}d\tau = \int p(i \to j|\bar{g}_i(t - \tau, \bar{x}_0))\delta(\bar{x} - \bar{g}_j(\tau, \bar{g}_i(t - \tau, \bar{x}_0)))\\ e^{-\int_0^t \lambda_i|\bar{g}_i(s,\bar{x}_0)|ds - \int_0^\tau \lambda_i|\bar{g}_j(s,\bar{g}_i(t-\tau,\bar{x}_0))|ds} d\tau.\end{aligned} \tag{2.12}$$

Protože předpokládáme, že druhý přechod $i \to j$ ukončí přechodový jev (tj. žádný další přechod po čase j nebude řešen, buď protože hranice Γ překročena, nebo protože j-tý přechodový jev dostane systém zpátky do bezpečné zóny), můžeme položit $\lambda_j = 0$.

K získání pravděpodobnosti přecházení skrz bezpečnou hranici Γ potřebujeme převést tuto hranici na absorbující plochu (tj. $\lambda_i(\bar{r}_s) = 0, \bar{r}_s \in \Gamma$). Potom:

$$\pi_\Gamma(i, t) = \int_{\bar{D}} [\pi(\bar{x}, i, t) + \sum_{j \neq i} \pi(\bar{x}, j, t)]d\bar{x} \tag{2.13}$$

je pravděpodobností přechodu přes hranici před časem t pro přechodový jev vycházející z x_0, i v $t = 0$.

Dostáváme:

$$\int_{\bar{D}} \pi(\bar{x}, i, t)d\bar{x} = H_{\bar{D}}(\bar{g}_i(t, \bar{x}_0))e^{-\int_0^t \lambda_i|\bar{g}_i(s,\bar{x}_0)|ds}, \tag{2.14}$$

kde $H_{\bar{D}}$ je charakteristickou funkcí $\bar{D}$:

$$\begin{aligned} H_{\bar{D}}(\bar{x}) &= 1 \quad \text{pokud} \quad \bar{x} \in \bar{D} \\ &= 0 \quad \text{pokud} \quad \bar{x} \notin \bar{D}. \end{aligned} \tag{2.15}$$

Protože $H_{\bar{D}} = 1$ pro $t > t_c(i, x_0)$, platí

$$\int_{\bar{D}} \pi(\bar{x}, i, t) d\bar{x} = H(t - t_c(i, \bar{x}_0)) e^{\int_0^{t_c} (i,\bar{x}_0)\lambda_i |\bar{g}_i(s,\bar{x}_0)| ds}, \tag{2.16}$$

kde $H(t) = 1$ pro $t \geq 0$ a $H(t) = 0$ pro $t < 0$.

Podobně:

$$\begin{aligned} \int_{\bar{D}} \pi(\bar{x}, j, t) d\bar{x} = & \int_0^{Min(t,t_c)} p(i \to j | \bar{g}_i(t - \tau, \bar{x}_0)) \\ & \times H_{\bar{D}}[\bar{g}_j(\tau, \bar{g}_i(t - \tau, \bar{x}_0))] e^{-\int_0^t \lambda_i |\bar{g}_i(s,\bar{x}_0)| ds} d\tau, \end{aligned} \tag{2.17}$$

protože přechod z $i \to j$ musí nastat před časem t a také před t_c, časem nutným k dosažení hranice Γ z $(i, \bar{x}_0)$.

Nyní přesněji určíme pravděpodobnost přechodu přes bezpečnou hranici:

$$\lim_{t \to \infty} \pi_\Gamma(i, t) = \pi_\Gamma(i, \infty). \tag{2.18}$$

Předpokládejme, že můžeme rozdělit stavy j do dvou tříd:

- $j \in J_-$ mají trajektorii přes hranici Γ směrem dovnitř: $\bar{n}.\bar{f}_j(\bar{x}_s)$ pro všechna $\bar{x}_s \in \Gamma$
- $j \in J_+$ mají trajektorii přes hranici Γ směrem ven: $\bar{n}.\bar{f}_j(\bar{x}_s) > 0$ pro všechna $\bar{x}_s \in \Gamma$.

Toto ale není nezbytně nutné, protože obecně je hranice $\Gamma = \Gamma_+(j) \cup \Gamma_-(j)$, kde $\Gamma_+(j)$ odpovídá $\bar{n}.\bar{f}_j(\bar{x}_s) > 0$, atd..

Záměnou $t - \tau$ a τ zjistíme, že pro dost velké t je $H_{\bar{D}}[\bar{g}_j(t - \tau, \bar{g}_i(\tau, \bar{x}_0))] = 1$ za podmínky, že $j \in J_+$ a $H_{\bar{D}}[\bar{g}_j(t - \tau, \bar{g}_i(\tau, \bar{x}_0))] = 0$, pokud $j \in J_-$; proto tedy

$$\pi_\Gamma(i,\infty) = e^{-\int_0^{t_c}(i,\bar{x}_0)\lambda_i|\bar{g}_i(s,\bar{x}_0)|ds} + \sum_{j\in J_-}\int_0^{t_c(i,\bar{x}_0)} d\tau \frac{p(i\to j|\bar{g}_i(\tau,\bar{x}_0))}{\lambda_i(\bar{g}_i(\tau,\bar{x}_0))}$$
$$\times\, \lambda_i|\bar{g}_i(\tau,\bar{x}_0)|e^{-\int_0^{\tau}\lambda_i|\bar{g}_i(s,\bar{x}_0)|ds} = e^{-\int_0^{t_c(i,\bar{x}_0)}\lambda_i|\bar{g}_i(s,\bar{x}_0)|ds}$$
$$+ \sum_{j\in J_-}\int_0^{t_c(i,\bar{x}_0)} \hat{p}(i\to j|\bar{g}_i(\tau,\bar{x}_0))\times dF_i(\tau,\bar{g}_i(\tau,\bar{x}_0)), \tag{2.19}$$

kde $\hat{p}(i\to j)$ je podmíněná pravděpodobnost taková, že pokud přechod nastane ve stavu i, bude následujícím stavem stav j. Jestliže je $\hat{p}$ nezávislá na $\bar{g}_i(\tau,\bar{x}_0)$, můžeme dále jednodušeji psát

$$\pi_\Gamma(i,\infty) = e^{-\int_0^{t_c(i,\bar{x}_0)}\lambda_i|\bar{g}_i(s,\bar{x}_0)|ds} + \sum_{j\in J_-}\hat{p}(i\to j)\Big[1-e^{-\int_0^{t_c(i,\bar{x}_0)}\lambda_i|\bar{g}_i(s,\bar{x}_0)|ds}\Big]. \tag{2.20}$$

Na závěr lze poznamenat, že data (dynamiky) mohou být nestálá a charakterizována vektorem $\bar{a}$ s rozdělením $p(\bar{a})$. Pravděpodobnost, že přechod nastane v počátečním ustáleném stavu $(x_0,1)$ v čase $t=0$, tj. bezpečná hranice bude překročena v nějakém čase $t>0$, je dána váženým průměrem:

$$\pi_\Gamma \triangleq \sum_i \int \hat{p}(1\to i|\bar{a})\pi_\Gamma(i,\infty|\bar{a})p(\bar{a})d\bar{a}$$
$$= \sum_i \int \hat{p}(1\to i|\bar{a})p(\bar{a})d\bar{a}\Bigg\{e^{-\int_0^{t_c(i,\bar{x}_0.\bar{a})}\lambda_i|\bar{a}.\bar{g}_i(s,\bar{x}_0.\bar{a})|ds}$$
$$+ \sum_{j\in J_-}\hat{p}(i\to j|\bar{a})\Big[1-e^{-\int_0^{t_c(i,\bar{x}_0.\bar{a})}\lambda_i|\bar{a}.\bar{g}_i(s,\bar{x}_0.\bar{a})|ds}\Big]\Bigg\}. \tag{2.21}$$

Fyzický význam (2.20) a (2.21) je zřejmý a znázorňuje důležitost časových proměnných.

2.2 Spolehlivost neopravitelných komponent

V tomto případě uvažujme neopravitelné komponenty. To znamená, že studovaný systém je komponentou, která má tu vlastnost, že při selhání zůstane v tomto stavu napořád. Potom je obvyklou definicí spolehlivosti (Shooman 1968):

2.2.1. DEFINICE. *Spolehlivost komponenty v čase t je pravděpodobnost, že komponenta korektně vykoná určený úkol během intervalu [0,t], daným podmínkami prostředí.*

Všimněme si, že definice uvádí vztah spolehlivosti komponenty k jeho prostředí. To je z toho důvodu, že stejná komponenta bude mít odlišné vlastnosti (spolehlivost) v závislosti na prostředí, ve kterém je umístěna.

Nechť τ je náhodná veličina reprezentující dobu do poruchy studované komponenty. Distribuční funkce veličiny τ je:

$$F(t) = P\{\tau \leq t\}, \tag{2.22}$$

definující pravděpodobnost, že systém selže nejpozději v čase t. Pro $F(t)$ platí následující vlastnosti:

$$\begin{cases} F(0) = 0 \\ \lim\limits_{t\to\infty} F(t) = 1 \\ F(t) \text{ je neklesající v t.} \end{cases} \tag{2.23}$$

Pravděpodobnost bezporuchového chodu je definována jako doplněk $F(t)$:

$$R(t) = P\{\tau > t\} = 1 - F(t), \tag{2.24}$$

a definuje pravděpodobnost, že systém je funkční v čase t. Z důvodu neopravitelnosti systému, funkční systém znamená, že během času t nedojde k žádné chybě. Pro funkci $R(t)$ platí:

$$\begin{cases} R(0) = 1 \\ \lim\limits_{t\to\infty} R(t) = 0 \\ R(t) \text{ je nerostoucí v t.} \end{cases} \tag{2.25}$$

Funkce hustoty pravděpodobnosti veličiny τ je:

$$f(t) = \frac{dF(t)}{dt} = -\frac{dR(t)}{dt}, \tag{2.26}$$

kde $f(t)$ je pravděpodobnost, že veličina τ padne do intervalu $(t, t + dt)$. Navíc:

$$\int_a^b = P\{a < \tau \leq b\} = F(b) - F(a)$$

reprezentuje pravděpodobnost, že k chybě dojde během intervalu $[a, b]$.

Střední hodnota veličiny τ, $E[\tau]$, je nazývána střední dobou do poruchy a značí se zkratkou $MTTF$ (Mean Time To Failure).

Intenzita poruch vyjadřuje pravděpodobnost, že komponenta se porouchá v čase $(t, t + dt)$, pokud do času t k žádné poruše nedošlo.

$$h(t) = P\{t < \tau \leq t + dt | \tau > t\} = \frac{P\{t < \tau \leq t + dt, \tau > t\}}{P\{\tau > t\}}. \tag{2.27}$$

Z (2.27), využitím (2.26), dostáváme:

$$h(t) = \frac{f(t)}{R(t)} = -\frac{1}{R(t)} \frac{dR(t)}{dt} \tag{2.28}$$

a řešením pro $R(t)$ je:

$$R(t) = exp\Big[- \int_0^t h(x)dx \Big]. \tag{2.29}$$

Předpokládejme nyní, že náhodnou veličinou τ je doba korektní funkčnosti systému. Důležitou úlohou je správné určení rozdělení popisující tuto náhodnou veličinu. Můžeme uvažovat dva typy rozdělení: exponenciální a Weibullovo. Hlavní charakteristikou exponenciálního rozdělení je konstantní intenzita poruch. Při konstantní intenzitě poruch, tj. $h(t) = \lambda$, můžeme z (2.28) a (2.26) odvodit:

$$\begin{aligned} F(t) &= 1 - e^{-\lambda t} \\ R(t) &= e^{-\lambda t} \\ f(t) &= \lambda e^{-\lambda t} \\ h(t) &= \lambda. \end{aligned} \tag{2.30}$$

Střední hodnota veličiny τ je převrácená hodnota intenzity poruch, $MTTF = 1/\lambda$. Exponenciální rozdělení je známé jako rozdělení bez paměti, protože spolehlivost komponenty je podmíněná faktem, že komponenta správně fungující po dobu $t = a$ je rovna spolehlivosti v čase $t = 0$. Exponenciální rozdělení popisuje dobře rozdělení doby života zařízení, u kterých dochází k poruše ze zcela náhodných příčin a nikoliv v důsledku opotřebení (mechanické opotřebení, únava materiálu apod.).

Distribuční funkce Weibullova rozdělení je definována:

$$F(t) = 1 - exp\Big[- (t/\eta)^{\beta} \Big],$$

kde $\eta > 0$ je parametr měřítka (posunutí na ose x) a $\beta > 0$ je parametr tvaru. Změnou hodnoty β získáme různé průběhy funkce intenzity poruch:

$$\begin{aligned} \beta < 1 &\Longrightarrow h(t) \text{ je klesající} \\ \beta = 1 &\Longrightarrow h(t) \text{ je konstantní} \\ \beta > 1 &\Longrightarrow h(t) \text{ je rostoucí.} \end{aligned}$$

Všimněme si, že exponenciální rozdělení může být aproximováno Weibullovým rozdělením s parametrem $\beta = 1$. Schopnost popsat různé chování intenzity poruch je hlavní přínos Weibullova rozdělení pro modelování spolehlivosti.

2.3 Spolehlivost opravitelných komponent

Nyní mějme komponentu, která může být opravena v případě selhání (poruchy). Chování opravitelné komponenty je určeno nejen jejím selháním, ale také procesem opravy. Proto můžeme život systému uvažovat jako změnu mezi dvěma stavy: Up (systém je funkční) a Down (systém je v procesu opravy).

Předpokládejme, že období korektní funkčnosti (doba do poruchy) a období nekorektní funkčnosti (doba opravy) jsou popsány náhodnými veličinami. Nechť $\tau_1, \tau_2, \tau_3, \ldots$ jsou náhodné veličiny popisující po sobě jdoucí trvání doby funkčnosti a $\theta_1, \theta_2, \theta_3, \ldots$ příslušné doby oprav. Za předpokladu, že oprava je "regenerační", tj. po opravě je komponenta "jako nová", mají všechny τ_i stejné rozdělení $F(t)$ a všechny θ_i rozdělení $G(t)$. Navíc můžeme popsat chování systému pouze pomocí dvou náhodných veličin τ (trvání stavu Up) a θ (trvání stavu Down). $G(t)$ popisuje pravděpodobnost, že komponenta je opravená v $[0, t]$ a je nazývána udržovatelnost. Podobně jako u $F(t)$ získáme i pro $G(t)$ následující vztahy:

$$\begin{aligned} g(t) &= \tfrac{dG(t)}{dt} \\ H_g(t) &= \tfrac{g(t)}{1-G(t)} \\ MTTR &= \textstyle\int_0^\infty t g(t) dt, \end{aligned}$$

kde $MTTR$ je střední doba do obnovy a $h_g(t)$ (intenzita obnovy) je pravděpodobnost, že oprava je ukončena v intervalu $[t, t + dt]$, jestliže komponenta nebyla opravena v čase t. Pokud předpokládáme, že intenzita opravy $h_g(t)$ je časově nezávislá,tj. $h_g(t) = \mu$, potom udržovatelnost je exponencionální funkcí:

$$G(t) = 1 - e^{-\mu t} \qquad \text{a} \qquad MTTR = \frac{1}{\mu}. \tag{2.31}$$

Předpoklad časové nezávislosti není moc reálný, jelikož obecně platí, že čas potřebný k ukončení opravy závisí na tom, jak dlouho již oprava probíhala, ale přesto je tento předpoklad často využíván, jak v literatuře tak v praxi, pro výhody, které nabízí při procesu řešení.

Je zřejmé, že pokud je systém závislý na poruchách a opravách, tak spolehlivostní funkce $R(t)$ není úplně informativní, protože pro t větší než doba první poruchy je hodnota funkce $R(t)$ blízká 0.

Proto je nutné nadefinovat novou veličinu, nazvanou použitelnost ($A(t)$). $A(t)$ je pravděpodobnost, že systém je ve stavu Up, v čase t.

$$A(t) = P\{\text{v čase } t, \text{ stav = Up}\}. \tag{2.32}$$

Nepohotovost $U(t)$ určuje pravděpodobnost, že stav systému je v čase t Down.

$$U(t) = P\{\text{v čase } t, \text{ stav = Down}\}, \tag{2.33}$$

a protože předpokládáme, že systém je v jednom ze stavů Up nebo Down, platí:

$$A(t) + U(t) = 1. \tag{2.34}$$

Výpočet $A(t)$ a $U(t)$ systému vychází z pozorování, že $A(t)$ $(U(t))$ odpovídá pravděpodobnosti, že systém je ve stavu Up (Down) v čase t. Pravděpodobnost stavu Up může být určena pomocí rovnovážné rovnice, protože

$$\begin{cases} \frac{dA(t)}{dt} = -\lambda A(t) + \mu U(t) \\ \\ \frac{dU(t)}{dt} = \lambda A(t) - \mu U(t). \end{cases} \tag{2.35}$$

Za předpokladu, že v čase $t = 0$ je systém funkční a pracuje, můžeme nastavit $A(0) = 1$ a řešit rovnice (2.35), dostáváme:

$$\begin{aligned} A(t) &= \tfrac{\mu}{\lambda+\mu} + \tfrac{\lambda}{\lambda+\mu} e^{(-\lambda+\mu)t} \\ \\ U(t) &= \tfrac{\lambda}{\lambda+\mu} - \tfrac{\lambda}{\lambda+\mu} e^{(-\lambda+\mu)t}, \end{aligned}$$

a obdržíme:

$$A(0) = 1 \qquad ; \qquad lim_{t\to\infty} = A_\infty = \frac{\mu}{(\lambda+\mu)}. \tag{2.36}$$

Protože u opravitelného systému je $MTTF \gg MTTR$ a tedy $\lambda \ll \mu$, příspěvek přechodové doby velmi rychle slábne a proto je spolehlivost často určena jejím asymptotickým chováním v (2.36).

Jestliže A_∞ je asymptotickou spolehlivostí, můžeme psát

$$A_\infty = \frac{\mu}{\lambda+\mu} = \frac{1/\lambda}{1/\lambda + 1/\mu} = \frac{MTTF}{MTTF + MTTR}.$$

I když tento výraz může být určen pouze za předpokladu konstantní intenzity poruchy a obnovy, v (Cox 1962) bylo dokázáno, že toto platí pro jakékoli rozdělení $F(t)$ a $G(t)$, pokud $MTTF$ je střední hodnota veličiny $F(t)$ a $MTTR$ je střední hodnotou veličiny $G(t)$.

2.4 Formální popis analýzy

Formálně může být dynamická spolehlivost brána jako součást struktury Pravděpodobnostní Dynamiky, která umožňuje vztah mezi spojitým dynamickým vývojem systému a diskrétními přechody mezi stavy systému. Charakteristické dynamické rysy vývoje zařízení mohou být zobecněny zavedením odpovídajících fyzických modelů, přičemž každý odpovídá konkrétní konfiguraci systému.

Uvažujme systém skládající se z N_C komponent. Každá komponenta má několik stavů charakterizované přechodovými stavy mající exponenciální rozdělení (toto předpokládáme z důvodu snadnější názornosti a navíc systém může být snadno převeden do modelu Monte Carlo simulace). Systém je navíc vybaven řídící/ochrannou jednotkou (ŘO) zasahující na podnět. Konfiguraci systému lze snadno popsat vektorem $(j_i, ..., j_{N_C})$, kde j_i je celočíselná hodnota označující konfiguraci i-té komponenty. V případě dvoustavové komponenty může být hodnota j_i =1 pro pohotovostní stav komponenty a j_i =2 pro stav selhání (např. pro systém dvou-stavových komponent (up, down) může základní konfigurace vypadat následovně: $j_1 = 1, j_2 = 1, ..., j_i = 2, ..., j_{N_C} = 1$). Alternativou označení může být setřídění a pojmenování všech konfigurací systému k=1,2,... . Mějme navíc $x(t) \in \mathbf{R}^n$ množinu procesních proměnných popisujících stav systému v čase t.

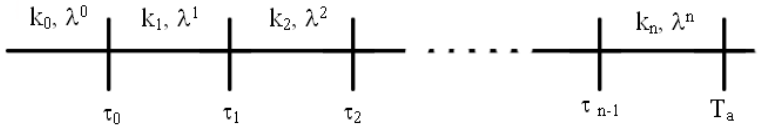
Chování systému může být popsáno následovně: začneme z počáteční události ($t_{k_0} = 0$, systém je určen počátečním stavem k_0 a vektorem procesních proměnných x_0), systém se bude deterministicky vyvíjet podle odpovídajícího dynamického modelu m_{k_0}, daného rovnicí

$$\frac{dx(t)}{dt} = m_{k_0}(x(t), t). \tag{2.37}$$

Tato rovnice popisuje časový vývoj systému (v pracovním stavu) do události (stochastické či uskutečněné ŘO) v čase t_{k_1}, která posune systém do stavu k_1. Poté se systém bude vyvíjet podle nové dynamiky m_{k_1}, atd..

Během vývoje systému může bod P, představující vektor x, úspěšně přejít přes tři různé oblasti fázového prostoru: bezpečná doména, šedá zóna a poruchová zóna. Bezpečná doména je relativně malá a ohraničená přednastavenými prahy zakročení ŘO. V tomto regionu se může procesní proměnná x významně měnit, např. během spouštění systému, ale trvání této změny je ohraničeno nadefinovanými prahy. Šedá zóna je mezi bezpečnou doménou a poruchovou zónou, přičemž poruchová zóna je ta, ve které se nachází systém v neobnovitelném stavu. Po události, v případě selhání ŘO, P vstoupí do šedé zóny a pokračuje k poruchové zóně. Během této cesty mohou být spuštěny vhodné akce obnovy k zastavení nehody. Je důležité poznamenat, že čas strávený v šedé zóně silně závisí na rychlosti a délce cesty z P skrz tento region. Tato situace je naznačená na Obrázku 2.1, z kterého je zřejmé, že větší pravděpodobnost

selhání je při vstupu P do šedé zóny z bodu A než z bodu B (předpokládejme, že P se pohybuje s konstantní rychlostí). Vypadá to, že určitá selhání jsou více kritická a zaslouží si více pozornosti. Navíc, v šedé zóně se může $x(t)$ značněji vychýlit ze své původní hodnoty a některé z přechodových rychlostí mohou prudce vzrůst.



Obrázek 2.2: *Časová linie stochastických/deterministických přechodů systému.*

Nyní se podívejme na vývoj nehody (Obrázek 2.2). Před nehodou je systém v činném stavu v počáteční konfiguraci k_0, vektor procesních proměnných je x_0 (uvnitř bezpečné domény) a součet všech možných přechodových rychlostí vedoucích z k_0 je λ^0. V čase τ_0 dojde ke stochastické chybě a poté:

1. při chybě se změní konfigurace systému z k_0 na k_1 a součet přechodových rychlostí z k_1 je nyní λ^1

2. vektor x procesních proměnných se začne měnit podle modelu popsaného rovnicí (2.37) a bod P se začne pohybovat pryč z jeho počáteční pozice (x_0, k_0); dříve či později P dosáhne hranice bezpečné domény a τ_1 je čas této události, jak je určeno integrací rovnice (2.37) s počáteční podmínkou x_0 a s m_{k_1}, což je model náležící konfiguraci k_1

3. v čase τ_1 je zařízení ŘO požádáno k zasažení, tj. modifikovat hardwarovou konfiguraci systému do nové konfigurace k_2 a stejně tak k přivedení bodu P zpátky do bezpečné domény. Prozatím předpokládejme, že ŘO je bezporuchové zařízení. V tomto případě přejde hardwarová konfigurace deterministicky do k_2 s rychlostí λ^2

4. začínajíc z $x(\tau_1)$ je rovnice (2.37) integrována s m_{k_2} do času τ_2, ve kterém P opět narazí na hranici bezpečné domény

V případě, že ŘO je bezporuchové zařízení, může popsaná posloupnost u bodu P, odrážejícího se od hranice bezpečné domény, pokračovat po celou dobu poruchy trvající T_a a navíc předpokládáme, že během tohoto časového období nastane $n-1$ zásahů ŘO zařízení v časech $\tau_1, \ldots, \tau_{n-1}$, přičemž poslední zásah opustí systém v konfiguraci k_n s celkovou přechodovou rychlostí rovnou λ^n.

Ve výše popsané posloupnosti je nejvíce nereálným bodem to, že předpokládáme bezporuchové ŘO zařízení, tj. systém, který funguje úspěšně s pravděpodobností rovnou jedné. Tento předpoklad může být zmírněn zavedením množiny pravděpodobností q_k, popisující pravděpodobnosti, že v časech τ_k, kdy systémové konfigurace jsou dány posloupností k, komponenty ŘO zařízení (při požádání k zásahu) úspěšně zareagují na dané požadavky. Pak pro odpovídající čas τ_k je dána také pomocná pravděpodobnost selhání ŘO při požadavku : $p_k = 1 - q_k$. V této více reálné situaci, kdy selhání zásahu při požadavku nastane v τ_k, pokračujeme únikem z P (se stejnou konfigurací k jako před neúspěšným zásahem) daným integrací rovnice (2.37) do šedé zóny. Pokud ŘO-zařízení pokračuje v selhání, pak po deterministickém šedém období τ^{gp} přejde bod P poslední hranici poruchové zóny a nastane totální porucha systému. Připomeňme, že během šedé zóny se mohou některé procesní proměnné významně měnit a to může vést ke způsobení podstatného kolísání v některých přechodových rychlostech. V tomto případě musí λ odpovídat vývoji procesních proměnných. Nejlepší způsob k realizaci zmíněné závislosti je předpokládání postupných změn v souladu s množinou dříve spočtených hodnot procesních proměnných: jakmile je jedna z těchto hodnot překročena, pak jsou λ postupně modifikovány. Po tomto je možno sledovat vývoj systému za bezpečnou doménou a také vzít v úvahu další možné stochastické selhání během šedého období při zachování formální jednoduchosti při zacházení s konstantami λ.

Kapitola 3

Přístupy využívané k modelování spolehlivosti systémů

Pro modelování systémů mohou být využity např. tyto metody:

- Markovské modely (Smith 2000)
- Metoda dynamických síťových grafů - DFM (Garrett 2002)
- Bayesovské metody (Zhang 2002, Pai 2002)
- Metody Petriho sítí

Zmíněné metody a citace nejsou kompletní, pouze reprezentují metody používané při modelování spolehlivosti systémů.

3.1 Markovské modely

Markovské modely jsou vhodným nástrojem pro analýzu komplexních systémů. Například v oblasti spolehlivosti může být činnost systému reprezentována stavovým diagramem, který znázorňuje stavy a intenzity dynamického systému. Tento diagram obsahuje uzly (představující možný stav systému, který je určen stavy jednotlivých komponent) spojené orientovanými hranami. Možné události (např. selhání nebo oprava komponenty) určují přechody. Činnost systému může být analyzována užitím Markovského modelu.

Analýza markovským modelem poskytuje spoustu užitečných vlastností popisujících činnost systému. Tyto vlastnosti jsou:

- spolehlivost systému
- střední dobu do poruchy
- udržovatelnost

- průměrný počet výskytů v daném stavu během daného časového intervalu

a mnoho dalších.

Pojem Markovský model je odvozen z předpokladu, který umožňuje analyzovat daný systém: Markovská vlastnost. Markovská vlastnost popisuje charakteristickou závislost vývoje - vývoj závisí pouze na předcházejícím stavu, budoucí vývoj systému je nezávislý na jeho historii. Markovská vlastnost je zajištěna, jestliže pravděpodobnosti přechodů jsou dány exponenciálním rozdělením s konstantní intenzitou poruchy či obnovy. V tomto případě se jedná o stacionární Markovský proces. Tento model je vhodný pro popis elektronických systémů s opravitelnými komponentami. Například Smith použil Markovský přístup pro určení spolehlivosti digitálních systémů (Smith 2000).

Předpoklady Markovského modelu mohou být modifikovány, aby bylo možno analyzovat složitější systémy. Markovské modely jsou aplikovatelné na systémy s běžnými chybami jako je např. elektrický šok, který může způsobit blesk elektrickému zařízení. Markovské modely mohou také popisovat komplexní proces opravy, komponenty s mnoha operačními stavy, závislé poruchy a další sekvenčně závislé události.

3.2 Metoda dynamických síťových grafů

Garret a Apostolakis popsali použití DFM pro spolehlivost I&C systémů (Garrett 2002). Přístup spojil I&C systém a další fyzické komponenty s procesními aspekty systému. Model zobrazil proměnné do konečného počtu stavů. Vlivy provozního chování komponent (včetně poruch) na výkon systému jsou reprezentovány pomocí rozhodovacích tabulek.
Základní problémy, které mohou mít dopad na efektivnost tohoto přístupu, zahrnují složitost výběru správné množiny stavů pro každou proměnnou a přesnost zkonstruovaných rozhodovacích tabulek. Je samozřejmě nezbytné zvolit kompromis mezi přesností modelu na jedné straně a velikostí a komplexností modelu na straně druhé. Jak již bylo uvedeno, některé aplikace DFM indikovaly, že kvantifikace modelu je možná i s chybovými daty (Guarro 2004). Jeho začlenění do teorie spolehlivosti vyžaduje další testování.

3.3 Bayesovské metody

Bayesovské metody využívají Bayesovský teorém a přístupy vyvinuté Thomasem Bayesem v 18. století. Metody se odlišují od klasických metod v aspektu jak praktickém,

tak i v aspektu základních teorií. Praktický rozdíl spočívá v tom, že Bayesovské metody připouští kombinaci technických dat s dalšími významnými informacemi, což je užitečné pro studie spolehlivosti. Možnými zdroji informací mohou být např. konstrukční návrh a testování dat, technická data z různých prostředí nebo osobní zkušenost s obdobným zařízením. Rozdíl v teorii spočívá v interpretaci pravděpodobnosti.

Informace, která je dostupná na začátku analýzy se nazývá *dřívější informace*. Problémem je získání odhadu parametru spolehlivosti, který kombinuje dřívější informaci (ve formě dřívějšího rozdělení) s experimentální informací. Toto je možno vyřešit využitím Bayesova teorému, který dřívější informaci transformuje na pozdější rozdělení.

Martz a Waller zpracovali v (Martz 1982) přehled týkající se použití Bayesovského přístupu v oblasti spolehlivosti. Zhang a Golay (Zhang 2002) popsali užití Bayesovských metod k určení spolehlivosti softwaru.

3.4 Metody Petriho sítí

Petriho sítě (Murata 1989) si zaslouží speciální zmínku z toho důvodu, že jsou často uvažovány jako hlavní alternativa k Booleovským modelům (chybové stromy a blokové diagramy) pro studie spolehlivosti.

Peterson (Peterson 1977) popisuje Petriho síť jako grafický modelovací jazyk, který je podobný konečnému automatu s přechody, hranami a uzly. Hrany spojují přechody s uzly (místy). Dále je využíváno tokenů, které se mohou pohybovat při spuštění Petriho sítě. Token se pohybuje z uzlu (místa) a je pohlcen odpovídajícím přechodem. Když je přechod uschopněn a následně proveden, produkuje tokeny do výstupních míst a pohlcuje token z každého místa, které je vstupním daného přechodu. Aby byl přechod uschopněn, každé jeho vstupní místo musí obsahovat alespoň jeden token. Petriho sítě byly široce používány při modelování počítačového hardwaru a softwaru (Peterson 1981). Jejich potenciál leží v jejich schopnosti uvážit synchronizace a paralelismy.

Marsan a G.Conte (Marsan 1984b) představují Zobecněné Stochastické Petriho Sítě (GSPNs) jako Petriho síť s přidanou množinou přechodů, které jsou uschopněny v náhodných časech. GSPNs jsou reprezentantem semi-Markovských procesů (Marsan 1995). Stochastické sítě mohou být použity pro kvalitativní popis dynamických systémů. Modely procesní proměnné mohou být spojeny s Petriho sítí a kvantitativní analýza Petriho sítě může být vykonána například pomocí Monte Carlo simulace. Petriho sítě mají také možnost využít hierarchické modelování, kde Petriho síť nejvyšší

úrovně může být rozložena na podsítě. Cordier a kol. (Cordier 1996) poskytují stručný popis těchto konceptů spolu s jejich aplikacemi. Různé aplikace Petriho sítí zabývající se procesními proměnnými a řešené pomocí simulace Monte Carlo jsou popsány v literatuře (Dutuit 1997, Chabot 1998).

Liu a Chiou (Liu 1997) popisují, jak mohou být Petriho sítě užity k přímé simulaci stromu poruch, kde uzly a přechody reprezentují různé typy logických bran, zahrnující inhibiční brány, brány zpoždění a M z N brány. Navíc je představen algoritmus pro generování souboru minimálních řezů a drah a nová modifikace Petriho sítí nazvaná Duální Petriho sítě. Duální Petriho sítě jsou používány k přímé konstrukci stromu poruch. Využitím těchto metod můžeme Petriho sítě vhodně využít k detekci chyb podobným způsobem jako bychom zapojili metodu stromu poruch.

Balakrishan a Trivedi (Balakrishman 1996) charakterizují příklad pro aplikaci stochastických Petriho sítí (SPN) v oblasti směrování sítě. Tento typ Petriho sítě, stejně jako GSPN, je použit k výpočtu Markovkých řetězců z modelu. Takovéto Petriho sítě mohou být využity pro vysokoúrovňový popis systému.

V (Rauzy 2002) je popsáno zobecnění Petriho sítí: mode automat. Mode automat je vstupně-výstupní automat s konečnou množinou stavů nazývanými módy. S některými omezeními a s určitou ztrátou informace mohou být tyto automaty transformovány na chybové stromy. V každém okamžiku je automat pouze v jednom módu. Jakmile nastane nějaká událost, tak se změní mód automatu. V každém módu je dána přechodová funkce, která spočítá hodnoty výstupů z hodnot vstupních toků. Můžeme ověřit vlastnosti dosažitelnosti, uváznutí, živosti, atd., stejně jako u Petriho sítí. Když je automat přeložen podle Booleovských formulí, pořadí událostí je odstraněno ve smyslu sloučení všech událostí do jedné. Modelovací síla základního mode automatu je větší než u Turingova stroje.

Goddard v (Goddard 1996) ukazuje, jak mohou být Petriho sítě spolu s metodou analýzy projevů a důsledků poruch (FMEA) aplikovány v požadavcích na určení chybějících bezpečnostních požadavků či nejistot a rozporuplnosti v bezpečnostních požadavcích. Tato metoda používá standardní Petriho sítě s přidáním inhibičních hran a podmíněných míst. Podmíněná místa jsou stejná jako normální místa s tím rozdílem, že nemohou přijímat či ztratit token během realizace Petriho sítě. Tato podmíněná místa jsou užitečná pro modelování spínačů nebo podmínek, které nejsou kontrolovány modelovaným systémem. Bezpečnostní požadavky jsou poté přeloženy do formy Petriho sítě a ta je následně provedená.

Petriho sítě mohou být také použity během návrhu k ověření, že jsou bezpečnostní požadavky vyhovující. Pokud Petriho síť ukáže, že porucha může nastat s méně než předurčeným počtem nezávislých selhání, potom může být návrh systému změněný tím způsobem, že snížíme počet nezávislých selhání nutných pro vyskytnutí poruchy. Touto cestou můžeme oddálit možnou příčinu selhání. Petriho síť je užívaná pro ověření návrhu systému a následně i k zajištění mimořádných podmínek.

Barevné Petriho sítě (CPNs)(Jensen 1991, Jensen 1997) jsou modelovací nástroj, které poskytují modelování, formální analýzu a ověření (simulační technikou) bezpečnostní procedury v reálných systémech. Kromě modelování a simulace byly CPNs a z nich odvozené Markovské grafy úspěšně využity pro analýzu spolehlivosti hybridních systémů (Schoeniga 2006). Navíc, jak je ukázáno v (Gerzson 1995), může být model procesu (ve formě kvalitativní diferenciální algebraické rovnice) reprezentován jako Barevná Petriho síť. Barevné Petriho sítě jsou úspěšně aplikovány v oblasti analýzy spolehlivosti, stejně jako pro modelování, řízení komponent a ověření bezpečnostních procedur. Vývojový nástroj založený na Barevné Petriho síti a zahrnující bázi znalostí pro ověření dynamického poplašného systému byl popsán v (Park 2002). V (Son 2003) je popsána metoda kombinace Barevných Petriho sítí a systému PVS (Prototype Verification System) pro ověření požadavků bezpečnostního softwaru. Fuzzy Barevné sítě byly aplikovány pro automaticky operující systém (Lee 2004). Dokonce i lidský faktor jako jsou vlastnosti a dynamičnost postřehu a akcí operátora může být popsán užitím CPNs (Kim 2007). Zavedení barev umožňuje někdy velmi výrazně stručnější zápis modelu. Analýza nad CPN, a to zejména formální analýza, je komplikovanější než nad klasickými Petriho sítěmi. Volba, zda užít Petriho sítě či CPN, pak záleží na konkrétní situaci - rozhoduje komplikovanost modelu, existence překladače z nějakého vhodného vyššího modelovacího jazyka do Petriho sítí (jež může nahradit použití mechanismů nabízených CPN), dostatečnost simulační analýzy nebo potřeba formální analýzy, dostupnost nástrojů pro formální analýzu apod.. Skutečnost, že rozhodnutí použít CPN může skutečně často převážit, lze dokumentovat řadou průmyslových studií, ve kterých byly CPN aplikovány (z nich mnohé jsou popsány ve třetím díle monografie (Jensen 1997)). Patří mezi ně například: komunikační protokoly a sítě, software (části SW Nokia, bankovní transakce, distribuované algoritmy,. . .),hardware nebo řídící a vojenské systémy.

Kapitola 4

Metody pro modelování spolehlivosti dynamických procesů

Protože většina aplikací řešila specifický problém, byly počítačové kódy napsané pouze pro testování návrhu aplikace. Z toho plyne, že popis systému byl buď napevno zakódován do programu nebo byl poskytnut ve formě textového souboru, který měl specifický formát pro každou verzi kódu. To znamená, že nebylo věnováno dostatek pozornosti pro určení spočitatelného kvalitativního schématu, které by využívalo metod dynamické spolehlivosti. Tento nedostatek uživatelské přívětivosti je jedním z hlavních důvodů pro omezení využití metod dynamické spolehlivosti. Vstupní schéma pro znázornění systému může být využito k provedení kvalitativní analýzy systému, která je základem nejen pro ověření vnitřní konzistence modelu a jeho aplikovatelnosti na modelování systému, ale také pomáhá samotnému modelovacímu procesu. V posledních letech se objevila snaha nasměrovat vstupní schéma tak, aby zachycovalo dynamické chování systému. Problémem je, že metody dynamické spolehlivosti obecně neposkytují pro znázornění systémů všeobecně použitelné schéma. Jednou z možností pro vytvoření vhodného vstupního schématu jsou grafické modely.

Grafy poskytují intuitivní znázornění logiky systému. Aby se využilo skutečnosti, že analytici jsou seznámeni s klasickou analýzou Stromu poruch/událostí, byly pro zachycení dynamického vývoje systémů navrženy rozšířené modifikace metody stromu poruch. Další grafické nástroje, vhodné pro práci s dynamickými systémy, které byly úspěšně aplikovány v různých inženýrských odvětvích jsou například: Petriho Sítě (Chatelet 1998, Dutuit 1997, Malhotra 1995, Tombuyses 1999, Vernez 2003, Volovoi 2004), Metody dynamických síťových grafů (Houtermans 2000, Houtermans 2002, Kaufman 2000),GO-FLOW (Matsuoka 2004, Matsuoka 1988) a Sekvenční diagram dynamické události (Swaminathan 1999a, Swaminathan 1999b, Swaminathan 1999c). Grafická reprezentace často slouží jako vstupní schéma numerické či matematické proceduře, např. Markovský řetězec, jehož řešením získáme numerický odhad systému. Techniky Petriho sítí, Go-flow a Sekvenčních diagramů rozšiřují schopnost vypořádat se s problémem dynamické spolehlivosti či s problémem analýzy rizika. Samozřejmě i tyto přístupy mají své nevýhody. Jednou společnou nevýhodou je to, že při exponenciálním růstu stavo-

vého prostoru, grafy rostou neúměrně rychle. Dalším omezením u některých grafových schémat je, že spoléhají na Markovský předpoklad, který u většiny reálných systémů nemusí platit a následně využitá Markovská aproximace může generovat nepřesné odhady.

Statistická metoda stromu událostí/poruch nezachází s časově závislými interakcemi mezi fyzickými procesy zařízení (např. zahřívání, natlakování) a spuštěnými nebo stochastickými logickými událostmi (např. otevření ventilu, nastartování pumpy) během nehody, což může vést k propojení těchto událostí skrz řídící systém. Dynamické metody, které mohou být užity k modelování těchto interakcí mohou být rozděleny do třech kategorií: časově spojité metody, časově diskrétní metody a metody s vizuálním rozhraním. Zatímco metody s vizuálním rozhraním jsou také buď spojité či diskrétní, důvodem proč jsou zmíněny zvlášť je schopnost vizuálního zobrazení, což je většinou považováno za uživatelsky přívětivější.

Časově spojité metody jako např. spojitý strom událostí (CET) (Devooght 1992b) poskytují pravděpodobnost nalezení systému v určité lokaci stavového prostoru, v určitém čase a konfiguraci. V CET je tato pravděpodobnost vypočítána z integrální rovnice, jejíž vstupy jsou modely fyzických procesů v diferenciální či integrální formě a přechodové rychlosti mezi jednotlivými stavy systémového hardwaru. Diskrétní verze CET stavového prostoru je metoda spojitého cell-to-cell zobrazení (CCCM) (Tombuyses 1996). CCCM definuje stavy systému tak, že se skládají z hardwarových konfigurací a uživatelem definovaných intervalů fyzických procesních proměnných. Pravděpodobnost vývoje systémových stavů je modelována s užitím spojité časové Markovské representace. Přechodové rychlosti stavů jsou získány z uživatelem poskytnutých systémových modelů a Chapman-Kolmogorovovy rovnice.

Diskrétní metody zahrnují následující:

- DYLAM (Dynamical Logical Methodology) (Amendola 1984, Cojazzi 1996), v principu se jedná o simulační zařízení schopné generovat větvení (scénáře) vývoje systému v uživatelem specifikovaných časových intervalech a schopného simulovat každou větev. Pro každý scénář je vyhodnocena časově závislá pravděpodobnost. Jakékoli nežádoucí následky jsou identifikovány z generovaných scénářů a jejich pravděpodobnost je zahrnuta do pravděpodobnosti incidentních větví.
- DETAM (Dynamic Event Tree Analysis Method) (Acosta 1993), metoda DDET (Dynamic Discrete Event Tree) (Acosta 1993) a ADS (Accident Dynamic Simulator) (Kae-Sheng 1996) jsou tři varianty systému DYLAM, které mohou dynamicky generovat v každém časovém kroku všechny možné stromy událostí.

- Simulační přístup Monte-Carlo (MC) z (Labeau 1998, Marchand 1998) používá diskrétní vzorkování ke zjištění možnosti vzniku větvení při vývoji systému kvůli porouchané komponentě a sleduje větve k určení pravděpodobnosti/frekvence nežádoucích událostí. Zatímco MC přístup z (Labeau 1998, Marchand 1998) může být také označen jako technika generování dynamického stromu události jako DYLAM, DETAM, DDET a ADS, liší se od těchto přístupů v tom, že MC přístup vybírá okamžiky větvení stochasticky místo užití deterministických pravidel.

- Hybridní simulace DDET/MC, která je popsána v (Marseguerra 1996), generuje větvení pomocí DDET a vybírá větve, které bude sledovat pomocí Monte-Carlo přístupu.

- CCMT (Cell-to-Cell Mapping Technique)(Aldemir 1991) je založena na diskrétní verzi CCCM a sleduje pravděpodobnostní vývoj systému s využitím Markovského řetězce.

Metody s vizuálním rozhraním zahrnují např. Petriho sítě (Gribaudo 2003, Dutuit 1997), dynamické síťové grafy (DFM) (Gardiner 1985, Gardiner 1985), dynamické stromy poruch (Andrews 1999, Cepin 2001), diagram sekvence událostí (ESD) (Swaminathan 1999c) a metodu GO-FLOW (Matsuoka 1991, Matsuoka 1988).

Podobným způsobem jako u analýzy chybového stromu mohou být i vizuální modely založené na Petriho sítích (Dutuit 1997, Gribaudo 2003) použity pro reprezentaci vztahů příčina-následek mezi událostmi a poskytnutí minimální množiny řezů. Ale narozdíl od analýzy chybového stromu, modely vytvořené pomocí techniky Petriho sítí umožňují také explicitní reprezentaci časového elementu ve vývoji systému s využitím dynamického modelu systému a následnou simulaci.

DFM (Gardiner 1985) je technika založená na orientovaných grafech. Procesní proměnná je reprezentována uzlem diskretizovaným do konečné množiny stavů. Dynamika systému je reprezentována pomocí vztahů příčina-následek mezi těmito stavy. Místo minimální množiny řezů poskytuje DFM primární implikanty pro systém. Primární implikant je jakýkoliv jednočlen (konjunkce primárních událostí), který je dostatečný k tomu, aby způsobil vrcholovou událost, ale neobsahuje žádné kratší konjunkce stejných událostí, které jsou dostatečné ke způsobení vrcholové události.

Metoda dynamických stromů poruch používá ke znázornění časově proměnných závislostí mezi základními událostmi časové vnější události (Cepin 2001) a nebo funkčně závislé hradla (Andrews 1999). Takové závislosti se mohou objevit kvůli propojení hardware skrz dynamik systému (Siu 1994a, Aldemir 1994a) (konkrétně v řídících

systémech), změně konfigurace (Cepin 2001) (např. kvůli údržbě) nebo číslicovému řízení (Andrews 1999). Kvantifikace dynamických stromů poruch je provedena s využitím časově závislé Booleovské logiky (Cepin 2001) nebo Markovských modelů (Andrews 1999).

Přístup ESD (Swaminathan 1999c) využívá ke znázornění pravděpodobného vývoje systému šestici událostí (např. inicializační, rozhodující, zpoždění), podmínek (omezení času, konkurence či přepínací podmínka), hradel (složená vstupní a výstupní AND/OR), množiny procesních parametrů, omezení a pravidla závislostí. Události představují přechody mezi stavy systému. Pravděpodobnostní přístup je rozšířením přístupu CET (Devooght 1992b) a je založen na Chapman-Kolmogorovově rovnici.

Metodologie GO-FLOW je technika pro analýzu úspěchově-orientovaných systémů, schopná vyhodnocení spolehlivosti a pohotovosti systému. Modelovací technika vytvoří GO-FLOW graf, který je složený z operátorů a signálových čar, reprezentující funkci systému. Signály znázorňují určité fyzické veličiny nebo informace. Výstup metodologie GO-FLOW obsahuje spolehlivost/pohotovost časově závislého systému, množiny řezu a dále analýzu nejistot. GO-FLOW je také vhodná pro modelování detailní dynamiky systému, jak je ukázáno v (Matsuoka 1991).

V literatuře (Siu 1994a, Belhadj 1992) se objevují případy, kde přesné modelování dynamik pravděpodobnostního systému může být důležité pro správné vyhodnocení frekvencí konkurenčních vrcholových událostí, obzvláště existuje-li nejistota v parametrech modelu (Aldemir 1996).

Kapitola 5

Petriho sítě

Petriho sítě hrají důležitou úlohu v životním cyklu distribuovaných systémů, začínající z počáteční fáze návrhu, přes vývoj systému, do údržby během jeho fáze provozu. Petriho sítě (PNs) byly poprvé představeny v doktorské práci Carla A. Petriho (Petri 1966), kde byly využity pro popis souběžných systémů kvůli vztahu příčina-následek bez přesného časového vyjádření. O několik let později byly zavedeny pomocí různých přístupů do modelu Petriho sítě pojmy týkající se času (Ramchandani 1974, Merlin 1976, Sifakis 1978). Následovalo velké množství různých prací, většinou založených na využití deterministického časování. V (Symons 1978, Florin 1985, Molloy 1982) byl poprvé zaveden pojem PN se stochastickým časem. Tyto studie umožnily použití Petriho sítí na poli hodnocení výkonu tradičně využívající přístup stochastického modelování. Takové modely jsou nyní souhrnně označovány jako Stochastické Petriho sítě (SPNs). Rozšíření přístupu navrženého v (Molloy 1982) bylo popsáno v (Marsan 1984a), kde náhodné časování bylo kombinováno s deterministickým nulovým zpožděním. Tzn., že časový a logický vývoj systému mohl být popsán jedním modelem. Takový modelovací nástroj je označován jako Zobecněná stochastické Petriho sítě (GSPNs).

Petriho sítě (PNs) jsou grafický nástroj pro formální popis systému, jehož dynamiky jsou charakterizovány souběžností, synchronizací a konfliktem, který je typickým rysem distribuovaných prostředí. Petriho sítě spojují pojem (distribuovaného) stavu a pravidla pro změnu stavu tak, že jim umožňují zachytit statické a zároveň i dynamické charakteristiky reálného systému. PNs mohou být úspěšně aplikovány v různých technických odvětvích - např. počítačová architektura, hardwarové a softwarové komponenty, algoritmy, komunikační protokoly, řídící systémy, přeprava, bankovnictví či organizace práce.

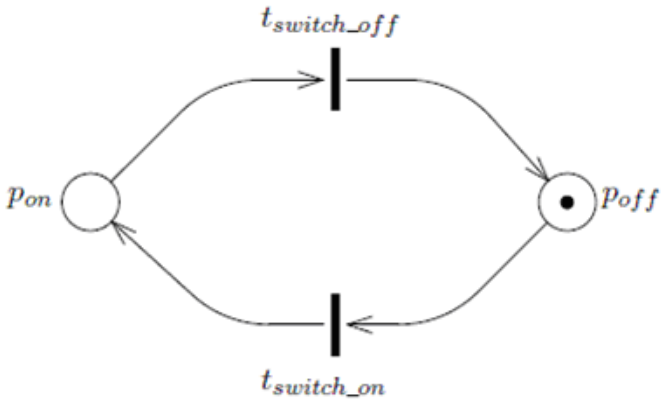
5.1 Co je Petriho síť?

Petriho síť je systém propojených míst a aktivit. Dále jsou dána pravidla, která určují, kdy může aktivita nastat a specifikují, jaké nastanou změny stavů míst. Petriho sítě mohou být použity k modelování a simulaci systémů různých typů. Jsou obzvláště

užitečné pro návrh a analýzu komplexních distribuovaných systémů.

PN zahrnuje místa, přechody a hrany. Zmíněné objekty definují strukturu sítě. Místa jsou využívaná pro popis možných stavů systému. Přechody popisují události, které mohou modifikovat stav systému a hrany specifikují vztah mezi stavy a událostmi a to dvěma způsoby: a) popisují stavy, ve kterých se mohou objevit dané události; b) změny stavu způsobené událostmi. Dalším důležitým prvkem je token. Tokeny jsou značky v místech a slouží ke specifikaci stavu sítě (tzv. značení Petriho sítě). Pokud místo popisuje podmínku (tudíž obsahuje maximálně jeden token), bude tato podmínka pravdivá v případě, že token je přítomen v místě. Pokud v místě token není obsažen, je podmínka nepravdivá. Další možností je, že místo definuje situaci a v tomto případě, počet tokenů obsažených v daném místě je použit ke specifikaci této situace.

Model Petriho sítě je graficky reprezentován orientovaným bipartitním grafem, ve kterém jsou místa znázorněna pomocí kružnic, příp.elips, a přechody pomocí obdélníků, příp.úseček. Ukázka jednoduché Petriho sítě je na Obrázku 6.13, který je převzat z (Marsan 1995).



Obrázek 5.1: *Jednoduchá Petriho síť popisují činnost vypínače.*

Tokeny jsou zakresleny jako černé body uvnitř míst. Dynamika sítě se řídí pomocí dvou pravidel:

- pravidlo stanovující podmínky proveditelnosti přechodu - *enabling rule*
- pravidlo popisující změnu stavu sítě po provedení přechodu - *firing rule*

Přechod může být proveden, jestliže všechna vstupní místa přechodu obsahují nejméně jeden token. V tomto případě lze říci, že přechod je uschopněn. Provedení uschopněného

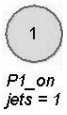
přechodu znamená, že se odstraní token z každého jeho vstupního místa a přidá se token do všech výstupních míst. V případě, že váha hrany je větší než jedna, počet tokenů potřebných v každém vstupním místě pro uschopnění přechodu a počet tokenů přidaných do každého výstupního místa je určen váhou hrany spojující místo a přechod. Provedení přechodu je atomickou operací.

5.2 Zobecněné stochastické Petriho sítě: stručné shrnutí

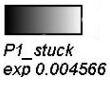
Modely a systémy Petriho sítě, které byly dosud uvažovány, neobsahovaly pojem času. Pojem času byl v práci C.A. Petriho (Petri 1966) záměrně vynechán, a to z toho důvodu, že časování může mít vliv na chování Petriho sítí. Spojení časových omezení s aktivitami znázorněnými v modelu nebo systému Petriho sítě může zabránit v provedení přechodů, čili vyvrácení významného předpokladu, že všechna možná chování reálného systému jsou znázornitelná pomocí struktury Petriho sítě.

Pokud provedení přechodu nenastane okamžitě a čas provedení přechodu je řízen podle exponenciálního rozdělení, síť se stane stochastickou Petriho sítí (SPN). Jestliže je přípustné použití okamžitých i časovaných přechodů, dostáváme Zobecněnou stochastickou Petriho síť (Balbo 1987). Pokud jsou uschopněny oba typy přechodů (časované i okamžité), mohou být provedeny pouze okamžité přechody. Zobecněné stochastické Petriho sítě (GSPNs) byly poprvé definovány v (Marsan 1984a). Definice byla později v (Marsan 1987) doplněna o využití strukturálních vlastností v modelování.

GSPNs jsou tedy rozšířením Petriho sítě obsahující také časované přechody, jejichž čas je určen náhodnou veličinou. Pro tvorbu modelu GSPN se používá následujících prvků (obrázky vytvořeny v programu Moca-RP):



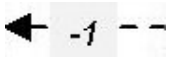
Místo 1, pojmenované P1_on, obsahující 1 token.



Stochastický přechod, jehož doba přechodu se řídí podle exponenciálního rozdělení.

Okamžitý přechod, je proveden okamžitě po uschopnění.

Orientovaná hrana, která je použita pro přenos tokenu v případě provedení přechodu. Hrana může být i násobná.



Inhibiční hrana. Přechod odpovídající k inhibiční hraně je uschopněn tehdy, pokud vstupní místo obsahuje méně tokenů než je váha inhibiční hrany (v absolutní hodnotě).

Obrázek 5.2: *Objekty využívané v GSPN.*

5.3 Barevné Petriho sítě

Pojem Petriho sítě byl postupně obohacován a zobecňován tak, aby jeho modelovací schopnost vyhověla praktickým potřebám. Ukázalo se totiž, že narážejí na pár závažných nedostatků. Jedním z problémů je to, že některé modely Petriho sítě jsou příliš rozsáhlé, jelikož všechny operace s daty musí být prováděny přímo na struktuře sítě. Z toho důvodu začal v 80. letech minulého století vývoj *Petriho sítí vyšší úrovně* (high-level PNs (Jensen 1991)), odstraňující výše zmíněný nedostatek. Výsledkem vývoje byly barevné Petriho sítě zavedené K. Jensenem (Jensen 1997), které umožňují stručnější zápis modelů.

Původní koncept černých tokenů nahrazují tokeny různých barev, což znamená zavedení typů do Petriho sítě. Jensen rozšířil Petriho sítě i o další prvky, jako proměnné, deklarace typů, inskripční výrazy hran, atd.. V místech se mohou nacházet multimnožiny tokenů různých barev, ale vždy jediného typu přiřazeného danému místu. Přechody mohou být ohodnoceny podmínkami nutných k provedení přechodu. Hrany jsou ohodnoceny multimnožinami tokenů, jejichž barvy patří do množiny barev přiřazené místu, které je s danou hranou incidentní.

Barevné Petriho sítě (CP-nets nebo CPNs) (Jensen 1997, Jensen 1998) poskytují framework pro tvorbu a analýzu distribuovaných a souběžných systémů. CPN-model systému popisuje stavy, ve kterých se systém může nacházet a přechody mezi těmito stavy. Barevné Petriho sítě byli často využívány v aplikační oblasti a také v průmyslu, např. v oblasti komunikačních protokolů (Floreani 1996, Huber 1991), audio/video systémů (Christensen 1997), operačních systémů (Cherkasova 1993), návrhu hardwaru (Genrich 1992, Shapiro 1991),vestavěných systémů (Rasmussen 1996), návrhu softwa-

rových systémů (McLendon 1992, Scheschonk 1994) a zdokonalování podnikových procesů (Mortensen 1994, Pinci 1991).

Vývoj barevných Petriho sítí byl veden potřebou vyvinout průmyslový robustní modelovací jazyk, který by byl současně všestranný a dobře teoreticky podložený, tak aby mohl být použit v praxi pro systémy, které svou velikosti a složitostí odpovídají typickým průmyslovým projektům. Aby toho bylo možno dosáhnout, zkombinovala se síla PNs (Murata 1989) se sílou programovacích jazyků. PNs poskytují prostředky pro popis synchronizace souběžných procesů, zatímco programovací jazyk poskytuje prostředky pro definici datových typů (barevných množin) a manipulaci s datovými proměnnými.

Modely CPN mohou být strukturovány do mnoha souvisejících modulů. Toto je obzvláště důležité, pokud pracujeme s modelem CPN velkého systému. Modulový koncept CPNs je založen na hierarchickém strukturování. Nové moduly mohou být vytvořeny z již existujících modulů a navíc mohou být použity v několika částech modelu CPN. Pomocí prostředků strukturování je možno zachytit ve stejném modelu CPN různé úrovně abstrakce modelovaného systému. Model CPN, který reprezentuje vysokou úroveň abstrakce, je typicky vytvořen v raném stadiu návrhu či analýzy. Tento model je postupně zpřesňován tak, aby poskytoval stále detailnější a přesnější popis systému.

Model CPN je proveditelný. To vyplývá z toho, že je možné studovat chování systému pomocí simulací modelu CPN. Cílem simulací je velmi často kontrola návrhu systému. Simulace mohou také sloužit jako základ pro zkoumání výkonu uvažovaného systému.

Vizualizace je technika, která je blízká simulaci modelu CPN. Pozorování každého jednotlivého simulačního kroku je často příliš detailní pro pochopení celého systému. To zahlcuje pozorovatele přílišným množstvím detailů obzvláště u rozsáhlých modelů CPN. Informace o funkčnosti systému lze získat i mnohem lépe, a to pomocí výsledků získaných ze simulací. Další důležitou aplikací vizualizace je schopnost prezentování návrhů a výsledků s použitím konceptů aplikačních domén. To je důležité hlavně v diskusi s lidmi a kolegy neorientujícími se v problematice barevných Petriho sítí.

Čas hraje významnou roli v oblasti distribuovaných a souběžných systémů. Správné fungování mnoha systémů závisí na čase, který zaberou jisté aktivity. Různý způsob jejich začlenění do návrhu může mít významný dopad na výkon systému. Časované modely CPN, spolu se simulací, mohou být použity pro analýzu výkonu systému, např. zkoumání kvality služeb (zpoždění) nebo počet služeb (např. výkonnost) poskytovaných systémem. Časový koncept CPNs je vhodný hlavně pro zkoumání systému pomocí simulace. Toto odlišuje analytické přístupy od analýzy výkonu (Marsan 1995) a modelovacích jazyků zaměřených na model kontrolování časovaných

(Clarke 1986, McMillan 1993) a hybridních systémů (Larsen 1997).

Využití stavového prostoru umožňuje CPNs zkontrolovat a ověřit správnou funkčnost systému. Metoda stavového prostoru se opírá o spočtení všech dosažitelných stavů a stavových změn systému a je založena na explicitním stavovém vyčíslení (Holzmann 1991, Huber 1986, Jensen 1997). Pomocí zkonstruovaného stavového prostoru lze ověřit vlastnosti chování systému. Příklady takových vlastností jsou např. nepřítomnost deadlocku v systému, možnost vždy dosáhnout daný stav a záruka zajištění dané služby. Metodu stavového prostoru CPNs lze také aplikovat na časované CPNs. Lze tedy dále ověřit správnou funkci modelovaného systému i pomocí časovaných CPNs.

Barevné sítě umožňují systémovým designérům a analytikům převést často složitý či téměř nemožný úkol, pracovat přímo s reálným systémem, na mnohem příjemnější a stravitelnější formu počítaných modelů, simulací a analýz. Tímto způsobem lze řešit velké množství problémů, avšak samozřejmě ne úplně všechny. I samotný model komplikovaného systému může být sám o sobě složitý. Na to, aby mohl být takovýto model vytvořen a efektivně využíván, je zapotřebí celé řady metod pro jeho efektivní zpracování. Dále je třeba si uvědomit, že počítačový model je stále počítačovým programem a jako takový se může potýkat se stejnými problémy jaké řeší i jiné konvenční programy (syntaktické chyby, sémantické chyby, chyby v návrhu ...). Jestliže má model správně plnit svoji funkci, je nezbytné tyto chyby detekovat a opravit.

Kapitola 6

Aplikace I. - Řešení benchmarku užitím Petriho sítí

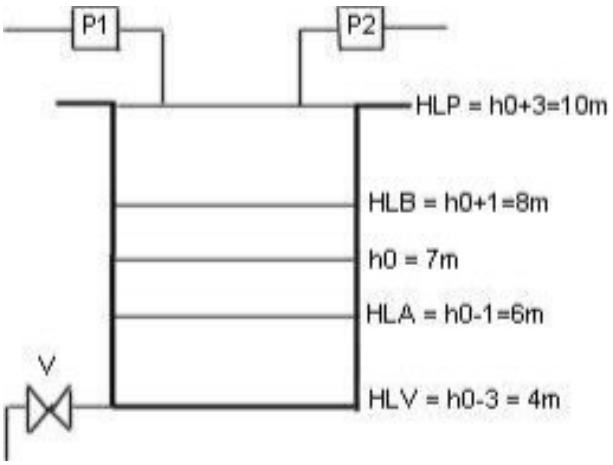
Jako první se budu zabývat řešením benchmarku převzatého z literatury (Marseguerra 1996). Konkrétně se jedná o přepouštěcí nádrž, která je široce užívaná v systémech různých průmyslových odvětví. Navíc řízení výšky hladiny tekutiny v nádrži je jedním z nejstarších řídících problémů, což dělá tento benchmark zajímavým a důležitým. V literatuře o dynamické pravděpodobnostní analýze rizika (PRA) již bylo diskutováno mnoho alternativ zmiňovaného problému nádrže. Aldemir použil problém nádrže jako příklad pro analýzu dynamik systému pomocí dynamického přístupu založeném na Markovském řetězci (Aldemir 1987). V (Deoss 1989) byl stejný problém studován užitím DYMCAM (Dynamic Monte Carlo Availability Model). Problém znázorňující různé metody dynamické PRA byl studován v (Siu 1994b). Cojjazi aplikoval DYLAM ke studiu podobného problému (Cojazzi 1996).

6.1 Popis a analýza řešeného systému

Uvažovaný systém (Obrázek 6.1) je složen z nádrže obsahující tekutinu, ze dvou pump (*P1*-hlavní pumpa a *P2*-rezervní pumpa), které slouží k plnění nádrže a z ventilu (*V*). Dále je součástí systému kontrolér monitorující výšku hladiny (*h*) a působící na komponenty systému (*P1*,*P2* a *V*). V tomto případě se jedná o dynamický systém s jednou procesní proměnnou (výška hladiny tekutiny v nádrži - *h*).

Komponenty systému se mohou nacházet ve třech stavech - ON (komponenty jsou otevřené), OFF (komponenty jsou zavřené) a STUCK (komponenty jsou porouchané). Jako počáteční hodnotu proměnné *h* uvažuji hodnotu rovnou 0 ($h_0 = 7m$), kdy *P1* a *V* jsou ve stavu ON a rezervní pumpa *P2* je ve stavu OFF. Protože všechny komponenty mají stejnou průtokovou rychlost (0.6 m/h), je zřejmé, že dokud trvá počáteční stav, výška hladiny se nemění. Příčinou kolísání hladiny (proměnné h) je výskyt poruchy jedné z komponent. Pravděpodobnost výskytu poruchy se řídí podle exponenciálního rozdělení. V Tabulce 6.1 je definována intenzita poruchy pro jednotlivé komponenty.

Tabulka 6.2 ukazuje závislost proměnné h na aktuálním stavu komponent. Korektní



Obrázek 6.1: *Schéma systému.*

komponenta	intenzita poruchy
$P1$	0.004566 l/h
$P2$	0.005714 l/h
V	0.003125 l/h

Tabulka 6.1: *Intenzity poruch.*

činnost systému je v případě, že se proměnná h pohybuje mezi hodnotami $(h_0 - 1)$ a $(h_0 + 1)$. Pokud h dosáhne hodnoty HLB $(h_0 + 1)$, nastává riziko přetečení nádrže. Tato událost nastane, jestliže h překročí výšku hladiny označenou jako HLP $(h_0 + 3)$. S cílem zabránit nechtěného stavu *přetečení* je nutné, aby kontrolér přepnul pumpy do stavu OFF a ventil do stavu ON.

Další nechtěnou události je stav, kdy proměnná h je pod hodnotou HLV (h_0-3). Tento stav pojmenuji jako *vysušení*. Pokud h dosáhne HLA $(h_0 - 1)$ musí kontrolér, s cílem zvýšení hladiny v nádrži, zapnout obě pumpy a vypnout ventil. Tabulka 6.3 definuje řídicí pravidla s ohledem na proměnnou h. K selhání celého systému nastane tehdy, pokud dojde alespoň k jedné ze dvou nechtěných událostí (*vysušení* nebo *přetečení*).

Z předcházejícího popisu systému můžu nadefinovat následující souhrn předpokladů:

- Průtok z ventilu a pump je identický, roven 0.6 m^3/h. Objem nádrže je 1 m^3. Z toho plyne, že můžu průtok znázornit jako hodinovou rychlost změny výšky hladiny rovnou 0.6 m/h.

P1	P2	V	změna h
ON	OFF	OFF	stoupá s rychlostí 0.6 m/h
ON	ON	OFF	stoupá s rychlostí 1.2 m/h
ON	ON	ON	stoupá s rychlostí 0.6 m/h
ON	OFF	ON	zůstává
OFF	OFF	OFF	zůstává
OFF	ON	OFF	stoupá s rychlostí 0.6 m/h
OFF	ON	ON	zůstává
OFF	OFF	ON	klesá s rychlostí 0.6 m/h

Tabulka 6.2: *Závislost výšky hladiny na stavu komponent.*

Výška hladiny	P1	P2	V
$h < h_0 - 1$	ON	ON	OFF
$h_0 - 1 \leq h \leq h_0 + 1$	ON	OFF	ON
$h > h_0 + 1$	OFF	OFF	ON

Tabulka 6.3: *Řídicí pravidla.*

- Počáteční podmínkou (v čase t=0) je, že výška hladiny je h_0 = 7 m, hlavní pumpa s ventilem je ve stavu ON a rezervní pumpa je ve stavu OFF

- Senzory jsou umístěny v různých úrovních nádrže a detekce nadefinovaných omezení pro výšku hladiny slouží řídicímu mechanismu pro řízení komponent podle Tabulky 6.3.

- Komponenty jsou navzájem nezávislé a nejsou opravitelné.

- Uvažuji následující poruchové situace: komponenty ($P1, P2, V$) se porouchají buď ve stavu ON, nebo ve stavu OFF.

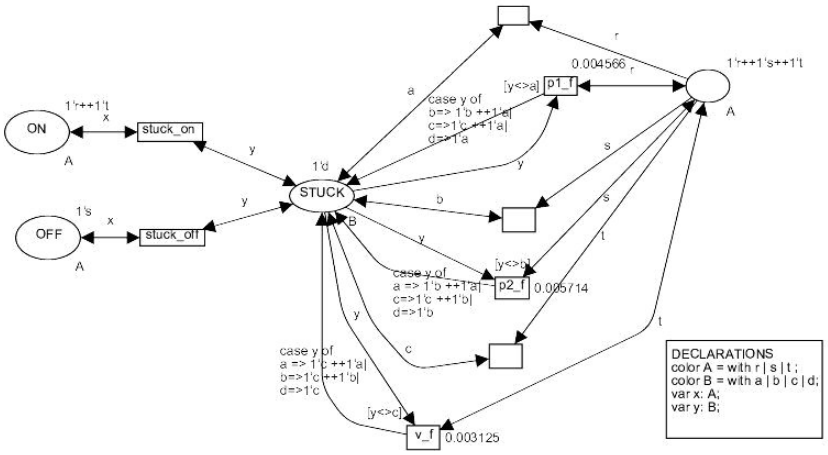
- Výskyt poruchy komponent se řídí podle exponenciálního rozdělení s rychlostmi poruchy α_1=$1/219h$ pro $P1$, α_2=$1/175h$ pro $P2$ a α_3=$1/320h$ pro ventil V.

- Mohou nastat dva nechtěné stavy vedoucí k selhání celého systému: 1) výška hladiny je větší než 10 m - dojde ke stavu *přetečení*, 2) výška hladiny v nádrži je menší než 4 m - nastane stav *vysušení*.

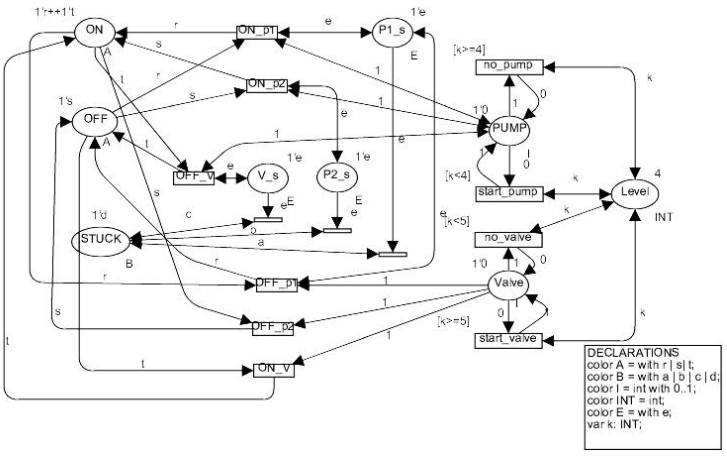
6.2 Tvorba modelu systému s využitím Barevných Petriho sítí

K tvorbě modelu systému jsem použila nástroje Barevných Petriho sítí. Hlavním důvodem využití tohoto typu Petriho sítí je, že Barevné Petriho sítě dosud v oblasti dynamické spolehlivosti (řešení zadaného benchmarku) nebyly aplikovány.

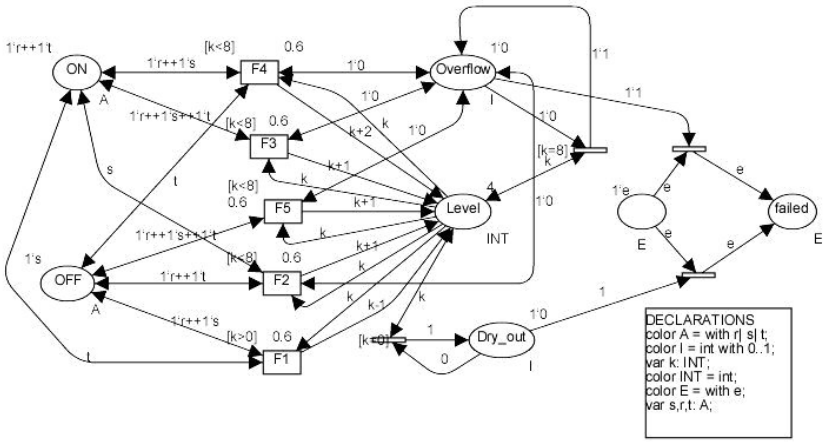
Pro přehlednost rozdělím model systému do tří subsítí. První (Obrázek 6.2) modeluje poruchy komponent, druhá (Obrázek 6.3) řízení komponent a třetí (Obrázek 6.4) je modelem pro stavy systému.



Obrázek 6.2: *Porucha komponent.*



Obrázek 6.3: *Řízení komponent.*



Obrázek 6.4: *Stavy systému.*

Tabulka 6.4 definuje místa využitá v modelu.

místo	popis
ON	komponenta je otevřená
OFF	komponenta je zavřená
STUCK	komponenta je porouchaná
Level	výška hladiny (počet tokenů)
PUMP	slouží k řízení pump
VALVE	slouží k řízení ventilu
Overflow	přetečení nádrže
Dry_out	nádrž je prázdná
Failed	selhání celého systému

Tabulka 6.4: *Popis míst.*

Síť na Obrázku 6.2 ukazuje vhodnost použitého přístupu Barevných Petriho sítí. Chování všech tří komponent systému je modelováno současně. To je umožněno použitím "barev" tokenů. "Barva" *r* reprezentuje hlavní pumpu, "barva" *s* rezervní pumpu a "barva" *t* představuje ventil v systému. *r, s, t* jsou datového typu A. Datové typy jsou uvedeny v DECLARATIONS u každé podsítě.

Stav komponenty je modelován pomocí tří míst - ON, OFF a STUCK. Místo STUCK je datového typu B a místa ON, OFF datového typu A. Místo STUCK se odlišuje z důvodu použití pomocné proměnné v případě, že žádná komponenta není porouchaná.

Jinak barva *a=r* a reprezentuje hlavní pumpu. Analogicky barva *b=s* a *c=t*.

Stav komponenty (např. hlavní pumpy) je modelován pomocí přítomnosti tokenu $1`r$. Pokud místo ON obsahuje $1`r$, znamená to, že hlavní pumpa je ve stavu otevřená. Stejné je to u místa OFF. Místo STUCK je označen $1`a$, pokud je hlavní pumpa porouchaná. Samozřejmě může nastat i situace, že všechny tři komponenty budou porouchané. Pak bude situace taková, že místo ON a OFF bude neoznačené a místo STUCK bude obsahovat multimnožinu $1`a++1`b++1`c$. Změna stavu komponenty *P1* kvůli poruše je modelována pomocí časovaného přechodu *p1_f* a okamžitého přechodu *stuck_on* nebo *stuck_off* a dalších pomocných nepojmenovaných přechodů. Přechod *p1_f* je stochastický a jeho doba přechodu je rovna intenzitě poruchy komponenty *P1*. Po provedení přechodu *p1_f* přejde komponenta ze stavu ON či OFF do stavu STUCK.

Porucha rezervní pumpy a ventilu je modelována stejným způsobem.

Druhou částí je modelování řízení komponent Obrázek 6.3 a popisuje aktuální stav nádrže, stav komponent a řídicí systém. Výška hladiny v nádrži může být diskretizována: devět možných stupňů můžu znázornit pomocí množiny tokenů v místě označeném Level. Datový typ místa Level je INT a počet tokenů v místě Level koresponduje s výškou hladiny (proměnnou *h*) - Tabulka 6.5.

počet tokenů	*h*	stav systému
8	>10 m	přetečení
7	10 m	HLP
6	9 m	
5	8 m	HLB
4	7 m	korektní činnost
3	6 m	HLA
2	5 m	
1	4 m	HLV
0	<4 m	vysušení

Tabulka 6.5: *Souvislost mezi h a počtem tokenů v místě Level.*

Řidící akce změny stavu komponent s ohledem na h je modelována pomocí dvou okamžitých přechodů pro každou komponentu (např. *ON_p1*, *OFF_p1*) a dalšími přechody *start_pump(valve)*, *no_pump(valve)*.

Pokud h klesne pod 7 metrů ($h \leq HLA$ a v místě Level je počet tokenů < 4) je nutné (podle Tabulky 6.3) přepnout pumpy do stavu ON a zavřít ventil. Tato procedura nazvaná *start_pump* je simulována pomocí přechodů *start_pump*,*ON_p1*, *ON_p2* a *OFF_V*.

Přechod *start_pump* je uschopněn v případě, že je splněna podmínka provedení přechodu $[k < 4]$. Analogicky je modelována procedura nazvaná *start_valve*. Ta nastane tehdy, že místo Level obsahuje více než 4 tokeny ($h > 7m$). Procedura *start_valve* znamená, že je nutné vypustit kapalinu otevřením ventilu a vypnutím pump.

Přechody *on_p1* a *off_p1* nemohou být uschopněny v případě, že je komponenta *P1* je ve stavu STUCK (místo P1_s je neoznačeno). Stejně i pro komponentu *P2* a *V*.

Poslední část (Obrázek 6.4) reprezentuje změnu výšky hladiny v nádrži, jak je definováno v Tabulce 6.2. Ke znázornění těchto změn jsem použila pět časovaných přechodů (*F1, F2, F3, F4, F5*). Všechny tyto přechody jsou prováděny se zpožděním 0.6 . Každý z těchto přechodů může být proveden pouze v situaci, dokud trvá daná konfigurace stavu komponent. Efektem provedení přechodu je přidání či odebrání tokenu z místa Level. Touto cestou modeluji zvyšování či snižování hladiny (h). Po provedení přechodů *F2, F3 a F5* se počet tokenů v místě Level zvýší o 1. V případě provedení přechodu *F1* se sníží o 1. Jestliže je uschopněn a následně proveden přechod *F4*, počet tokenů se zvýší o dva, což reprezentuje rychlejší růst výšky hladiny.

To, zda nastane jeden ze dvou nechtěných stavů (přetečení, vysušení) je detekováno pomocí dvou okamžitých přechodů: *overflow* a *dryout*. První je proveden tehdy, pokud místo Level obsahuje osm tokenů ($h > 10m$) a stav přetečení je dán výskytem tokenu $1`1$ v místě Overflow. Přechod *dryout* je uschopněn a proveden, pokud místo Level obsahuje nula tokenů ($h < 4m$). Po provedení přejde do místa Dry_out token $1`1$, což znamená, že nastal stav vysušení.

Jakmile dojde k označení místa failed, nastala porucha celého systému.

6.3 Modifikace systému I. - návrh údržby

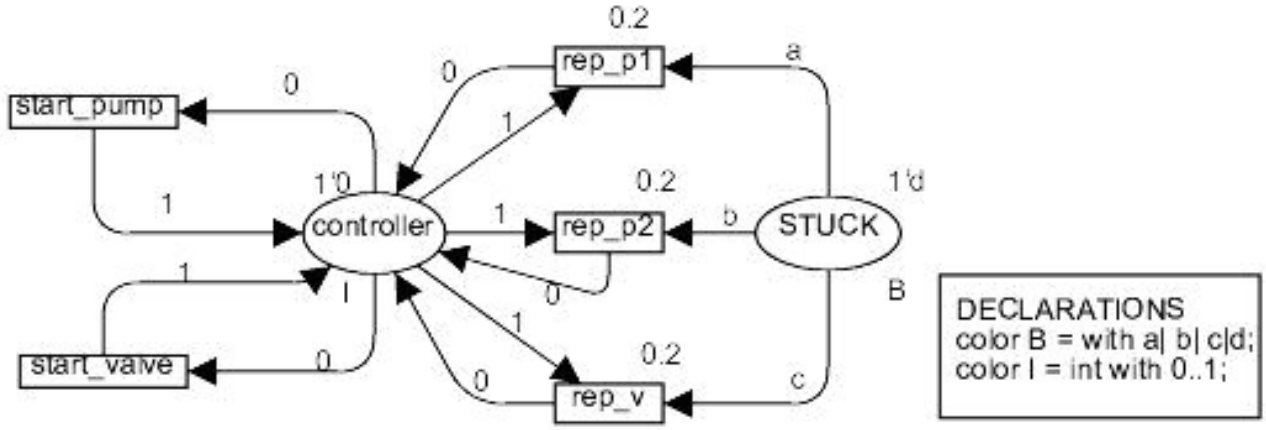
Dosud jsem předpokládala, že komponenty jsou neopravitelné. To znamenalo, že jakmile se komponenta porouchala a přešla do stavu STUCK, nebylo možné pomocí této porouchané komponenty regulovat výšku hladiny a došlo k jedné z uvažovaných událostí, která vyvolala poruchu systému. Toto je důvod, proč je nutné modifikovat dříve definovaný systém. Modifikace spočívá v tom, že komponenty systému se staly opravitelnými. Takto modifikovaný systém jsem řešila dvěmi způsoby navržení údržby komponent. Oba způsoby jsem opět namodelovala užitím Petriho sítí a postupně zakomponovala do již vytvořeného PN-modelu systému.

První způsob spočíval ve vytvoření nové komponenty nazvané kontrolor, která detekuje možnou poruchu komponenty pozorováním změny výšky hladiny v nádrži. Pokud se výška hladiny nenachází uvnitř oblasti korektní funkce systému, má kontrolor podezření na výskyt poruchy a to spustí proces opravy pro porouchanou komponentu.

V případě zvýšení hladiny (tzn. je uschopněn přechod *start_pump*) kontrolor před-

pokládá, že by mohlo dojít k přetečení nádrže, a proto zkontroluje, zda se některá z komponent nachází ve stavu zaseklá. Pokud ano, spustí opravu zaseklé komponenty. Doba trvání opravy je pro všechny komponenty shodná. K tomuto slouží časované přechody *rep_p1*, *rep_p2* a *rep_v*.

Pokud dojde ke snižování hladiny (tzn. je uschopněn přechod *start_valve*) dochází ke stejnému procesu. Procedura opravy pomocí kontrolora je namodelována pomocí Petriho sítí (Obrázek 6.5).



Obrázek 6.5: *Petriho síť popisují opravu komponent.*

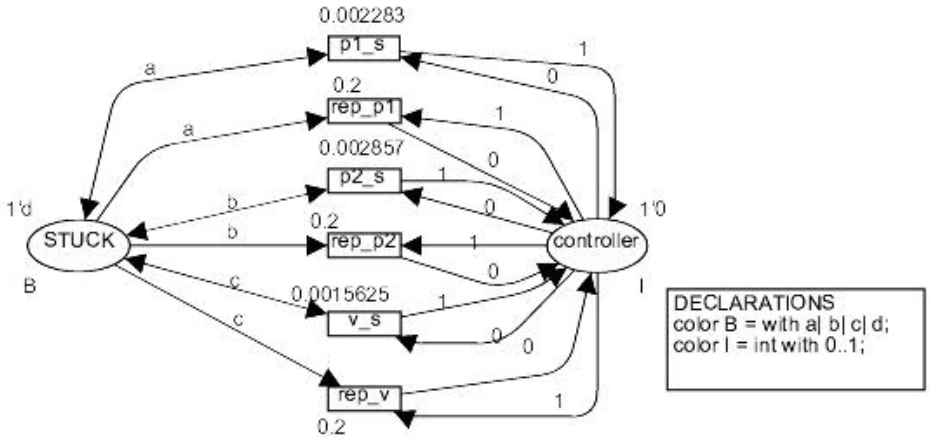
Druhou alternativou řešení je návrh preventivní periodické údržby. Ta spočívá v tom, že komponenta kontrolor periodicky po určité době (pro každou komponentu se jedná o čas shodný s polovičním časem intenzity poruchy dané komponenty) kontroluje, zda je některá komponenta zaseklá. Pro namodelování tohoto procesu je využito tří stochastických přechodů *p1_s*, *p2_s* a *v_s*. Když zjistí, že některá komponenta je ve stavu STUCK, spustí proces opravy (se stejnou dobou trvání opravy jako v případě první alternativy řešení) dané komponenty.

Například, při poruše komponenty $P1$ je v místě STUCK token $1`a$. Během procesu kontroly je uschopněn přechod *p1_s* a kontrolor nastartuje proces opravy komponenty $P1$ pomocí přechodu *rep_p1*.

Model preventivní periodické údržby je znázorněn na Obrázku 6.6.

6.4 Získané výsledky

Pro určení spolehlivosti systému jsem musela určit kumulativní distribuční funkci (cdf) pro stav *vysušení* a *přetečení* vytvořeného modelu Petriho sítě. To znamenalo spočítat

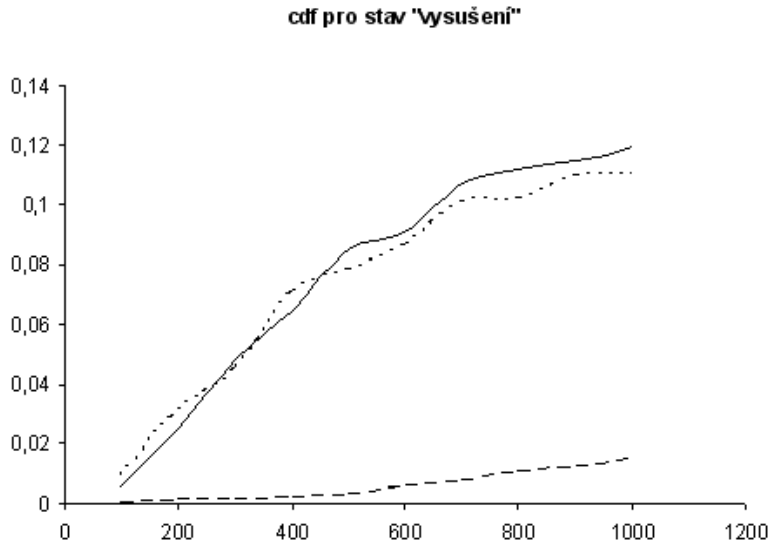


Obrázek 6.6: *Petriho síť modelující preventivní periodickou údržbu.*

pravděpodobnost, že systém se nachází v jednom z nechtěných stavů. Konkrétně, kumulativní distribuční funkce stavu *vysušení* byla spočítána jako pravděpodobnost přítomnosti tokenu 1‘1 v místě Dry_out. Stejně jsem postupovala u stavu *přetečení*: cdf stavu *přetečení* je pravděpodobností přítomnosti tokenu 1‘1 v místě Overflow. Postupně jsem počítala výše zmíněné cdf pro původní systém a pro modifikovaný systém (opravitelné komponenty - obě varianty). Výsledky byly spočteny pro simulační čas v rozpětí 0 až 1000 hodin. Získané výsledky jsou uvedeny v Tabulce 6.6 a Tabulce 6.7 a graficky znázorněné na Obrázcích 6.7 a 6.8.

hodiny	neopravitelné komponenty	oprava komponent	preventivní údržba
100	0.00524	0.0092	0.00006
200	0.0251	0.0312	0.00095
300	0.0486	0.045	0.00124
400	0.0645	0.0712	0.0019
500	0.085	0.0781	0.00323
600	0.0911	0.087	0.00554
700	0.1068	0.101	0.0075
800	0.112	0.102	0.0105
900	0.115	0.1095	0.0124
1000	0.1191	0.11	0.0153

Tabulka 6.6: *cdf pro stav vysušení.*



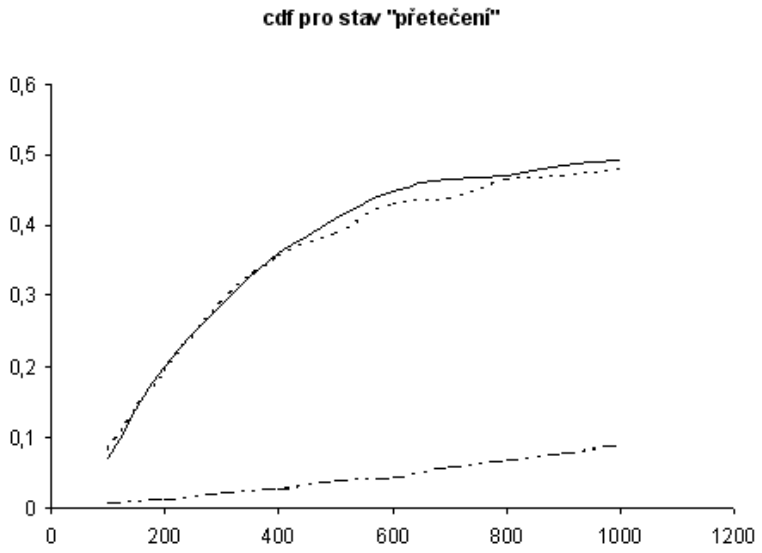
Obrázek 6.7: *cdf pro stav vysušení.*

hodiny	neopravitelné komponenty	oprava komponent	preventivní údržba
100	0.0712	0.0821	0.0042
200	0.1981	0.192	0.00945
300	0.289	0.291	0.021
400	0.361	0.356	0.0265
500	0.409	0.389	0.0361
600	0.4492	0.4286	0.0413
700	0.4651	0.4371	0.056
800	0.471	0.461	0.067
900	0.486	0.468	0.076
1000	0.493	0.4765	0.086

Tabulka 6.7: *cdf pro stav přetečení.*

Celá čára na Obrázcích 6.7 a 6.8 znázorňuje cdf pro systém bez opravy, přerušovaná čára cdf pro systém s procesem opravy a tečkovaná čára je použita pro znázornění cdf v systému s preventivní údržbou.

Pro ověření správnosti vytvořeného modelu systému jsem porovnala (Tabulka 6.8) mnou určené hodnoty cdf s hodnotami cdf získané simulací modelu systému vytvořeného pomocí černobílých Petriho sítí (výsledky jsem získala v (Raiteri 2005)).



Obrázek 6.8: *cdf pro stav přetečení.*

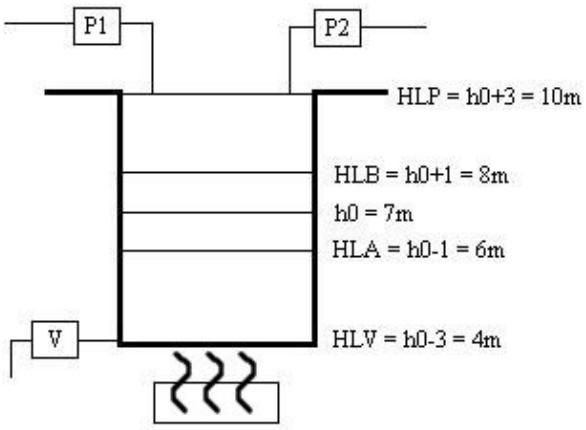
hodiny	cdf (barevné PNs)	cdf (černobílé PNs)
100	0.00524	0.004463
200	0.0251	0.022077
300	0.0486	0.044846
400	0.0645	0.065827
500	0.085	0.082568
600	0.0911	0.095014
700	0.1068	0.103939
800	0.112	0.110227
900	0.115	0.114622
1000	0.1191	0.117689

Tabulka 6.8: *Porovnání výsledků.*

Závěrem mohu konstatovat, že použití Barevných Petriho sítí je vhodnou volbou k vytvoření modelu dynamického systému. Pro určité typy systémů (podobná činnost různých komponent v systému, rozsáhlé systémy, hierarchické systémy, atd.) dokonce volbou vhodnější při využití vlastností barevných Petriho sítí. Dalším závěrem je konstatování, že využití preventivní údržby je lepší volbou než použití procesu opravy až při selhání komponenty. Výsledky byly postupně publikovány například v [6], [7], [8] a [10].

6.5 Modifikace systému II. - přidání druhé procesní proměnné

V tomto případě jsem uvažovala systém se dvěma proměnnými. Přidanou proměnnou je teplota kapaliny v nádrži ($\Theta(t)$). Schéma modifikovaného systému je na Obrázku 6.9.



Obrázek 6.9: *Systém se dvěma proměnnými.*

Počáteční teplota kapaliny (v čase t=0) je $\Theta(0) = 50°$C. Teplo dodávané ze zdroje je rovno $W = 753.48MJ/h$ a kapalina proudící z pump *P1* a *P2* má teplotu $\Theta_{P1(2)} = 15°$C.

Zachování energie je dáno následující diferenciální rovnicí:

$$\frac{d\Theta(t)}{dt} = \alpha_1 q_1 \Theta_{P1} + \alpha_2 q_2 \Theta_{P2} - \alpha_3 q_3 \Theta_t + W, \tag{6.1}$$

kde

α_i = 1, pokud komponenta $i = 1, 2, 3$ je ve stavu ON

α_i = 0, pokud komponenta i je ve stavu OFF

q_i představuje průtokovou rychlost komponenty i

Θ_{Pi} je teplota kapaliny přitékající do nádrže z pump

$\Theta(t)$ je teplota kapaliny v nádrži v čase t.

U systému se dvěma proměnnými může dojít ke třem nechtěným stavům (vrcholovým událostem) vedoucím k poruše systému: *vysušení*, *přetečení* a *teplota*. Stav *teplota* nastane v případě, že $\Theta(t) \geq 100°$C a může nastat v jedné ze tří možných oblastí, které specifikují výšku hladiny kapaliny v nádrži (Tabulka 6.9).

h	*P1*	*P2*
$h < h_0 - 1$	OFF	OFF
$h < h_0 - 1$	ON	OFF
$h < h_0 - 1$	OFF	ON
$h_0 - 1 \leq h \leq h_0 + 1$	OFF	OFF
$h_0 - 1 \leq h \leq h_0 + 1$	ON	OFF
$h_0 - 1 \leq h \leq h_0 + 1$	OFF	ON
$h > h_0 + 1$	OFF	OFF
$h > h_0 + 1$	ON	OFF
$h > h_0 + 1$	OFF	ON

Tabulka 6.9: *Konfigurace pro stav teplota.*

Modifikovaný systém jsem opět namodelovala pomocí Petriho sítí. Pro srovnání, respektive posouzení vhodnosti využití Barevných Petriho sítí, jsem systém namodelovala pomocí černobílých i Barevných Petriho sítí.

V obou variantách je nutné přidat model Petriho sítě znázorňující chování systému pro procesní proměnnou *teplota*.

6.5.1 Využití Barevných Petriho sítí

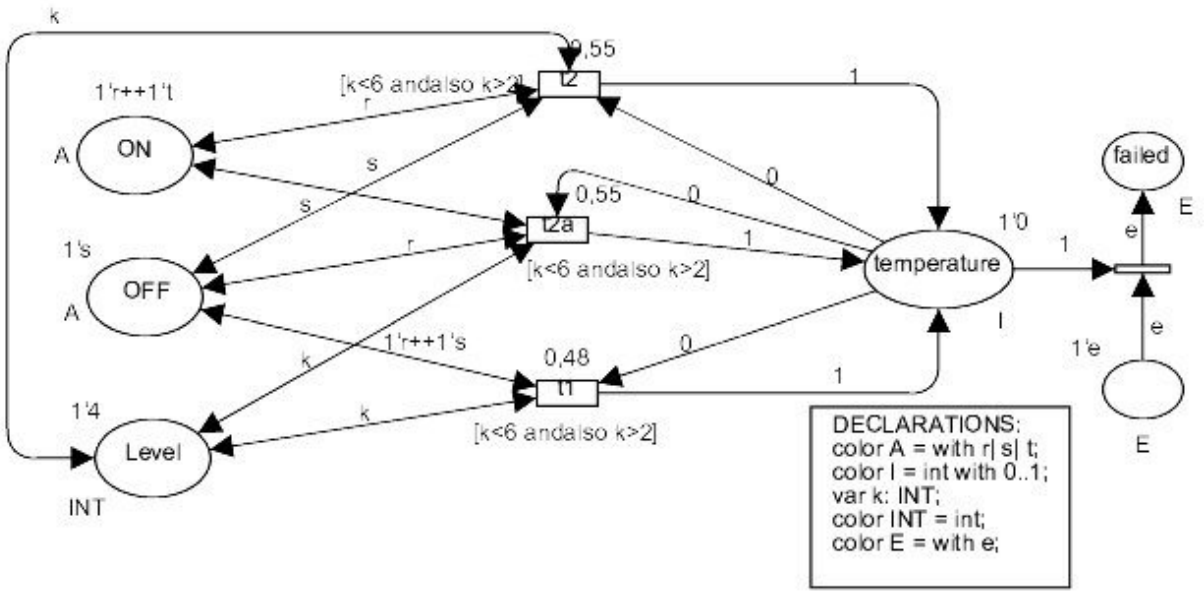
Pro tvorbu PN modelu systému se dvěma proměnnými jsem využila již vytvořený model popsaný v podkapitole 6.2. Navíc jsem přidala místo temperature pro stav *teplota* a model znázorňující dynamiku systému definovanou v Tabulce 6.9. Pro lepší pochopení jsem síť tvořila separátně a rozdělila do tří sítí pro každou uvažovanou oblast. Na Obrázku 6.10 je uvedena Petriho síť modelující změnu stavu komponent pro druhou oblast (4-6 řádek v Tabulce 6.9).

Přechod *t2* je časovaný, kde zpoždění přechodu je určeno z podmínek vedoucí k výskytu stavu *teplota*, a je uschopněn (může být proveden), pokud hlavní pumpa *P1* je ve stavu ON (místo ON obsahuje token $1`r$), rezervní pumpa ve stavu OFF (v místě OFF je token $1`s$), počet tokenů v místě Level je 3, 4 nebo 5 ($6 \leq h \leq 8$) a místo temperature obsahuje $1`0$.

Po provedení výše zmíněného přechodu je přidán token $1`1$ do místa temperature a ve stejný moment je z místa temperature odebrán token $1`0$. Tato situace modeluje situaci, kdy dojde k vrcholové události *teplota* a následně nastane porucha celého systému.

Stejným způsobem jsem zpracovala všechny další varianty.

Po vytvoření a verifikaci modelu CPN jsem opět určila kumulativní distribuční funkci



Obrázek 6.10: *CPN-model pro druhou oblast.*

pro všechny tři uvažované vrcholové události vedoucí k selhání systému jako pravděpodobnost přítomnosti tokenu 1‘1 v místě Dry_out (respektive Overflow či temperature). Spočtené hodnoty jsou graficky znázorněny na Obrázku 6.11.

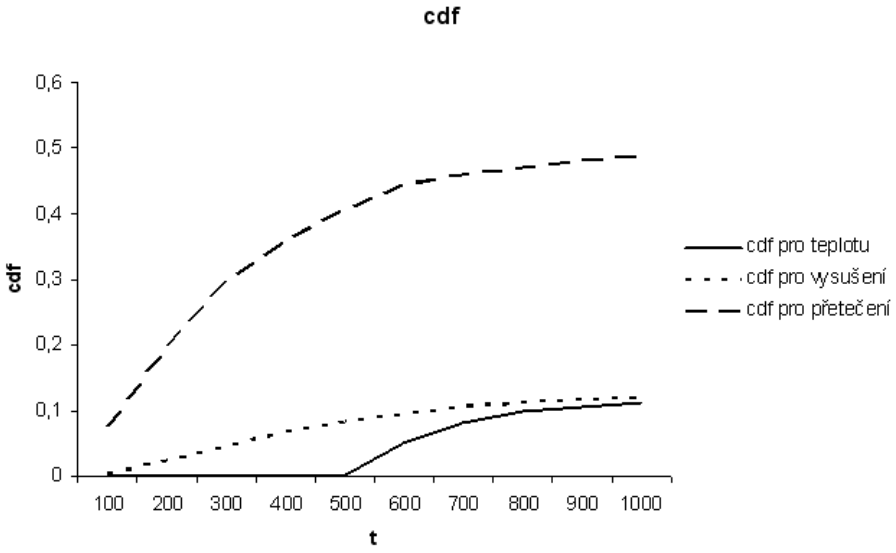
Pro systém se dvěma proměnnými jsem opět uvažovala i variantu s opravitelnými komponentami. Proto jsem do navrženého modelu CPN zahrnula model CPN preventivní údržby a následně určila cdf pro všechny tři nechtěné stavy (Obrázek 6.12).

Celá čára na Obrázku 6.11 a Obrázku 6.12 znázorňuje cdf pro stav *teplota*, přerušovaná čára cdf pro stav *přetečení* a tečkovaná čára je použita pro znázornění cdf pro stav *vysušení*.

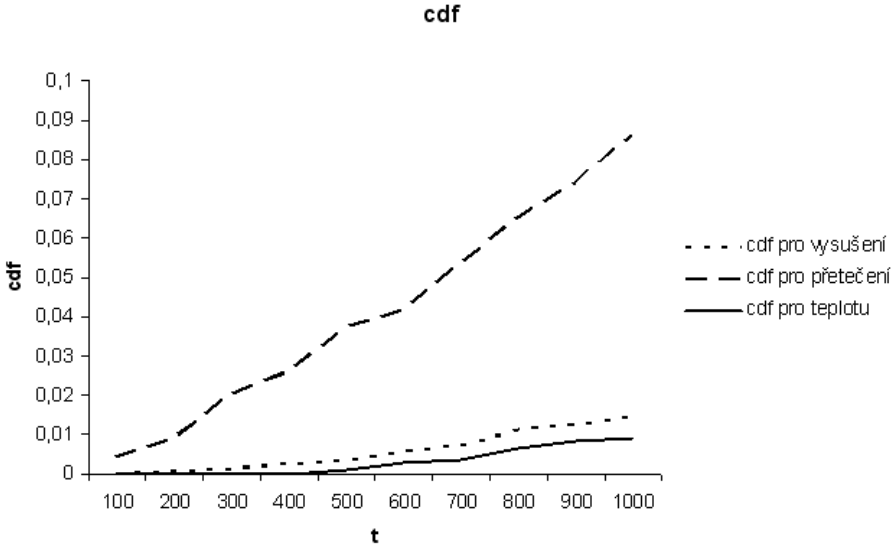
6.5.2 Využití černobílých Petriho sítí

Pro srovnání a kontrolu výsledků u použitého modelu barevné PN jsem vytvořila model systému se dvěma proměnnými s použitím černobílých Petriho sítí. Pro tvorbu modelu PN jsem použila stejnou analýzu systému, jak je postupně definována a popsána v kapitolách 6.1, 6.2, 6.5 a 6.5.1.

Na Obrázku 6.13 je ukázán celý vytvořený model PN.

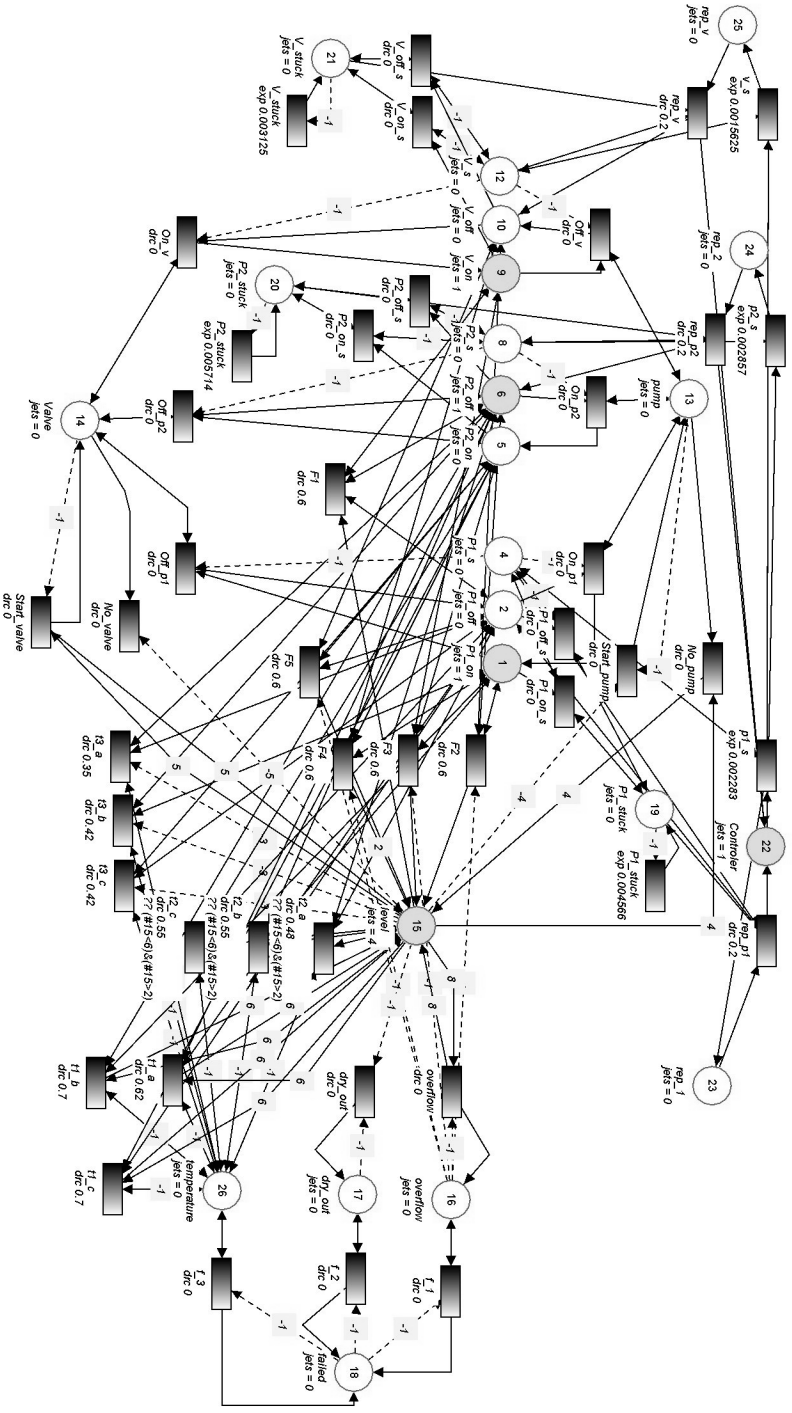


Obrázek 6.11: *Spočtené cdf pro neopravitelné komponenty.*



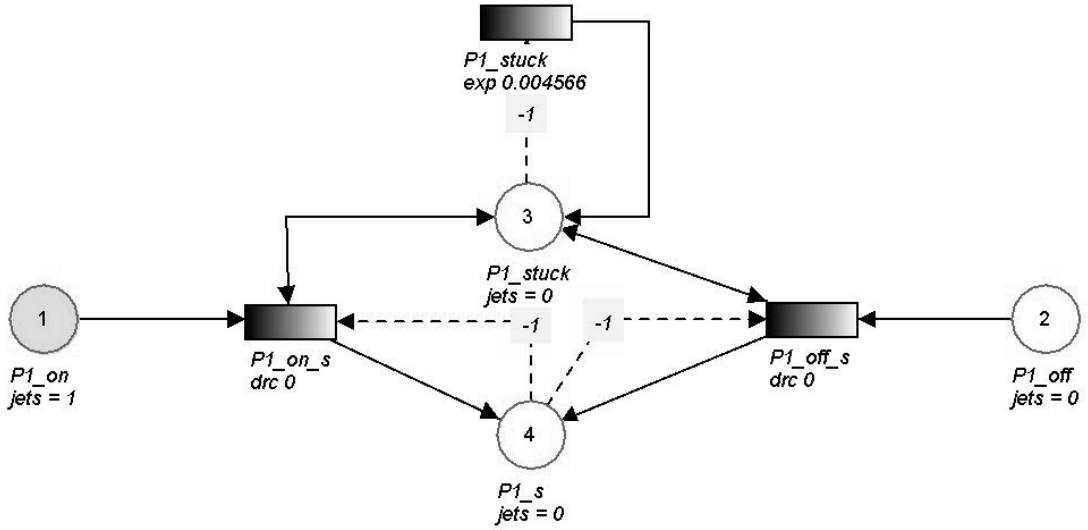
Obrázek 6.12: *Spočtené cdf pro opravitelné komponenty.*

Opět pro lepší znázornění rozdělím model na části. Na Obrázku 6.14 je ukázána Petriho síť modelující poruchu hlavní pumpy. Počáteční stav komponenty *P1* je stav ON, tzn. v místě P1_on je obsažen jeden token. Přechody *P1_on_s* a *P1_off_s* jsou okamžité



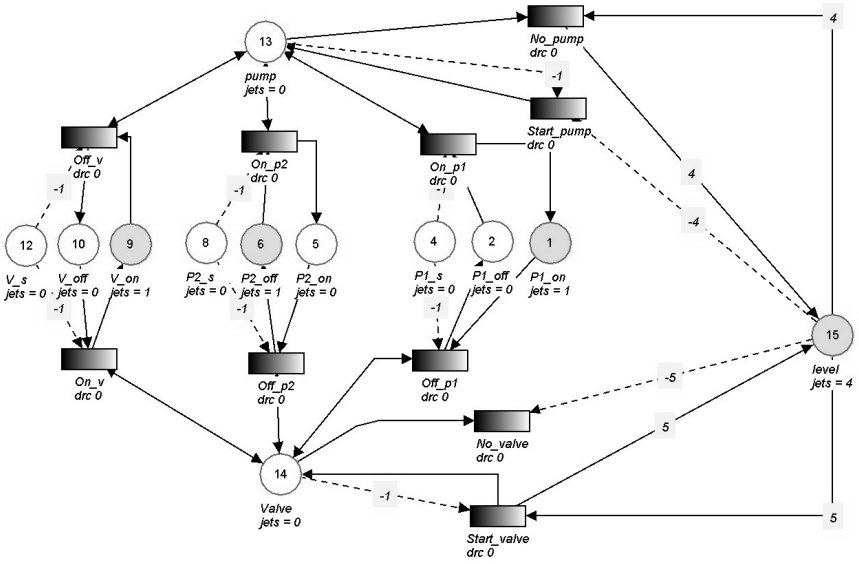
Obrázek 6.13: *Model PN pro systém se dvěma proměnnými.*

a přechod *P1_stuck* je stochastický. Jakmile je přechod *P1_stuck* proveden (dojde k poruše komponenty), místo P1_stuck se označí. Následně je uschopněn jeden z přechodů *P1_on_s* a *P1_off_s* (záleží na aktuálním stavu komponenty) a token se objeví v místě P1_s, což značí, že komponenta *P1* je ve stavu STUCK. Analogicky je namodelován proces poruchy rezervní pumpy a ventilu.



Obrázek 6.14: *Model PN pro proces poruchy P1.*

Na Obrázku 6.15 je model Petriho sítě popisující proces řízení komponent. Místo level slouží ke znázornění aktuální výšky hladiny v nádrži (může obsahovat 0-8 tokenů, jak je definováno v Tabulce 6.5). U modelu černobílé PN jsem využila inhibičních hran. Např. přechod *Start_pump* je uschopněn v případě, že hladina v nádrži je < 7m (v místě level je méně než 4 tokeny). Ukončení přechodu *Start_pump*, tj. objevení se tokenu v místě pump, znamená spuštění operace změna stavu komponent vedoucí ke zvýšení hladiny komponent. Tato činnost je modelována pomocí tří okamžitých přechodů *Off_v*, *On_p2* a *On_p1*. Pokud je jedna z komponent ve stavu STUCK (místo V_s, P2_s nebo P1_s obsahuje token), nemůže být provedena změna stavu komponenty (modelováno pomocí inhibiční hrany). Po ukončení procesu Start_pump (místo level obsahuje 4 tokeny) je uschopněn přechod *No_pump* a z místa pump je odebrán token umožňující proces změny stavu komponent s efektem zvýšení hladiny v nádrži. Podobným způsobem jsem navrhla i operaci řízení komponent vedoucí ke snížení hladiny kapaliny (označená jako Start_valve).

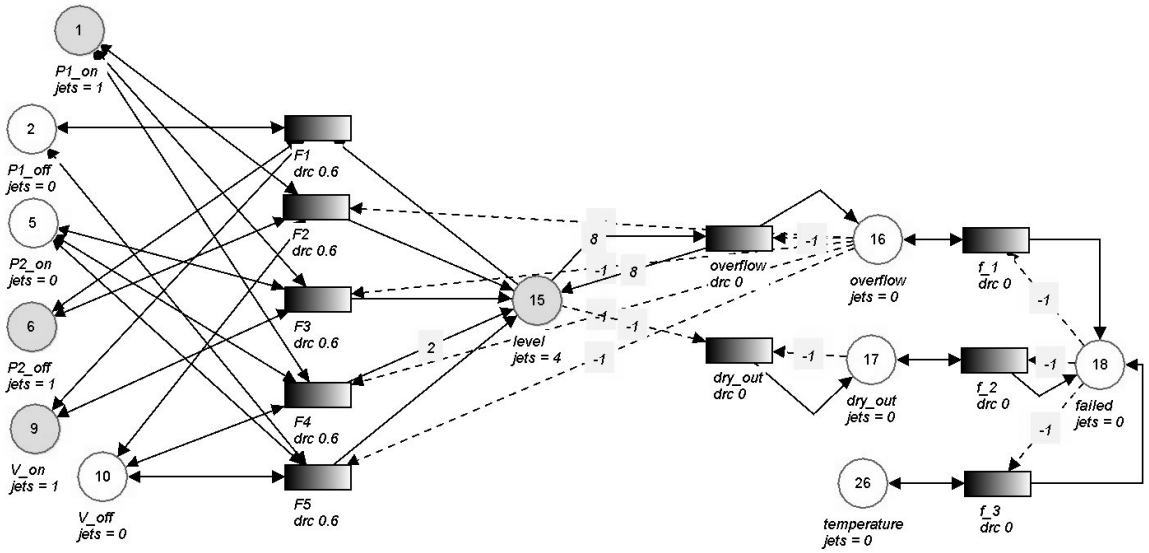


Obrázek 6.15: *PN-model pro proces řízení komponent.*

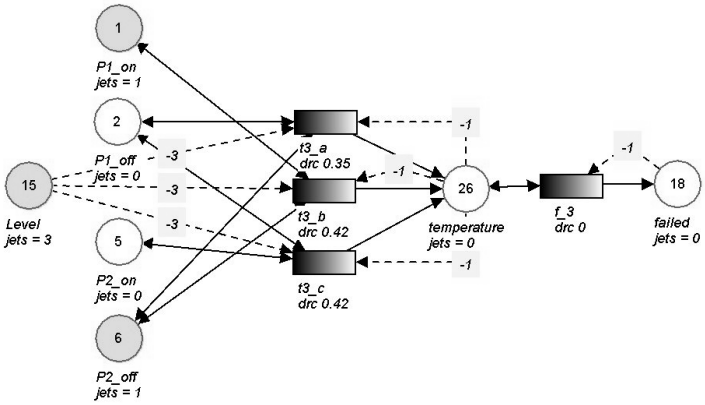
Další částí je PN-model popisující změnu výšky hladiny v nádrži a výskyt poruchových stavů systému (Obrázek 6.16). Proces změny výšky hladiny jsem namodelovala pomocí pěti časovaných přechodů. Při provedení přechodu *F1* dojde k odebrání tokenu z místa level (sníží se výška hladiny) a provedení přechodů *F2, F3, F4, F5* vede ke zvýšení počtu tokenů v místě level (Tabulka 6.2).

V situaci, že místo level obsahuje 8 tokenů, dojde k uschopnění a provedení přechodu *overflow*, což má za následek výskyt tokenu v místě overflow a tudíž poruchu celého systému (místo failed je označeno). Výskyt stavu *vysušení* je určen uschopněním přechodu *dry_out* a následnou přítomností tokenu v místě dry_out. Přechod *dry_out* je uschopněn v případě, že místo level neobsahuje žádný token (použita inhibiční hrana). Poslední vrcholovou událostí vedoucí k poruše systému je stav *teplota*. K vytvoření modelu vedoucí k tomuto stavu jsem využila Tabulky 6.9. PN-modely pro všechny tři uvažované oblasti jsou znázorněny postupně na Obrázcích 6.17, 6.18 a 6.19.

U tvorby GSPN-modelu systému se dvěma proměnnými jsem uvažovala pouze variantu s opravitelnými komponentami. Proto jsem do modelu zahrnula proces preventivní kontroly komponent. Tuto variantu údržby jsem použila z toho důvodu, že při vyhodnocení a simulaci modelu systému s jednou proměnnou se projevila jako výhodnější a vhodnější. Model Petriho sítě pro preventivní údržbu komponent je na Obrázku 6.20. Navržený model odpovídá popisu a analýze popsané v kapitole 6.3.



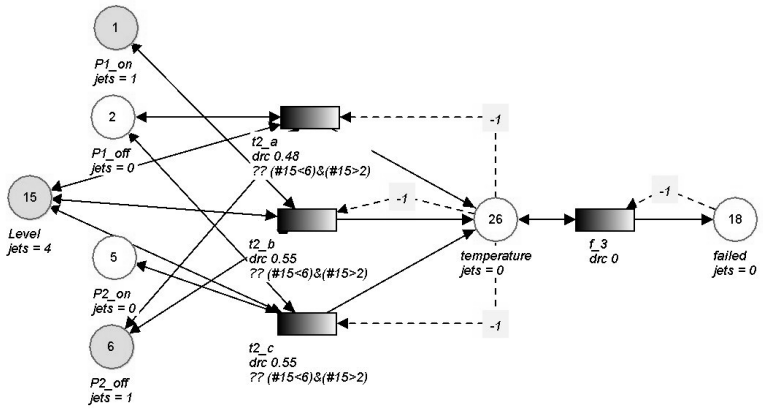
Obrázek 6.16: *PN-model popisující změnu proměnné h.*



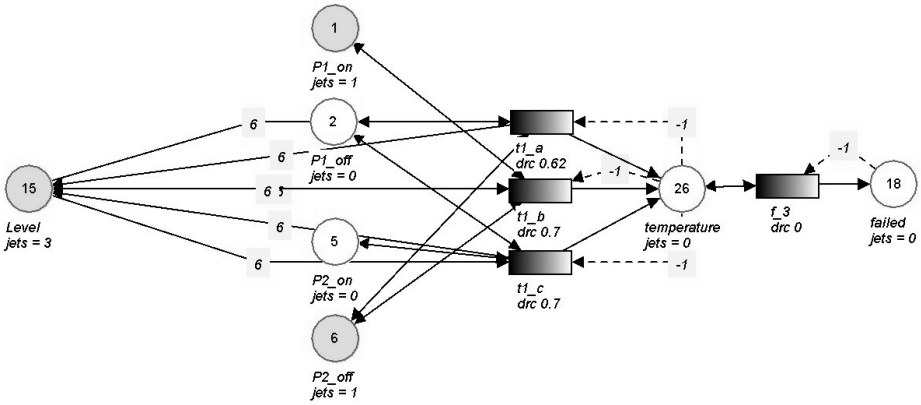
Obrázek 6.17: *PN-model pro první oblast.*

6.5.3 Výsledky simulací

Vytvořený model systému se dvěma proměnnými (s opravitelnými komponentami) jsem opět využila jako vstupní schéma do simulací. Postupně jsem použila jak model vytvořený pomocí barevných Petriho sítí, tak i GSPN-model. Cílem simulací bylo určení kumulativní distribuční funkce (cdf) pro všechny tři uvažované vrcholové události a následné porovnání použitých přístupů k tvorbě modelů systému.



Obrázek 6.18: *PN-model pro druhou oblast.*

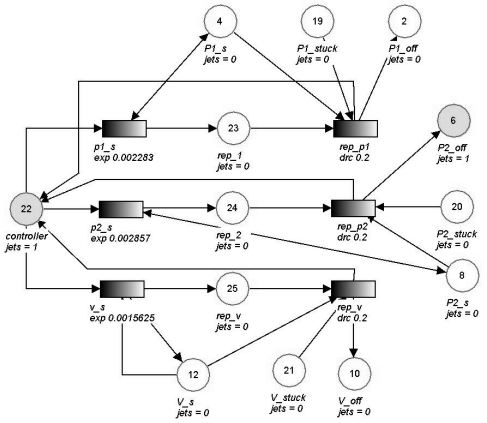


Obrázek 6.19: *PN-model pro třetí oblast.*

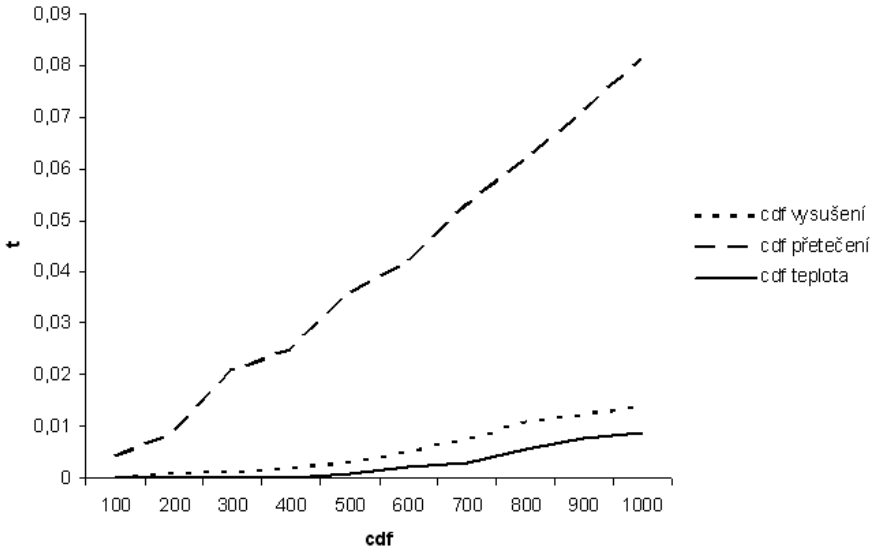
Určit hodnoty cdf znamenalo spočítat pravděpodobnost výskytu tokenu (v případě CPN-modelu 1‘1 tokenu) v místě charakterizující daný stav. Např. pro nechtěný stav *teplota* jsem počítala pravděpodobnost výskytu tokenu v místě temperature.

Obrázek 6.21 ukazuje vývoj hodnot cdf určené pro GSPN-model. Z důvodu srovnání výsledků obou použitých přístupů jsou na Obrázcích 6.22, 6.23 a 6.24 graficky znázorněny vypočtené hodnoty cdf během simulací pro oba modely. Na Obrázku 6.22 se jedná o spočtené cdf pro stav *teplota*. Obrázek 6.23 se týká stavu *vysušení* a Obrázek 6.24 popisuje výsledky simulací pro vrcholovou událost *přetečení*.

Je vidět, že oba použité přístupy jsou vhodné pro vytvoření spolehlivostního modelu a popisu dynamických systémů. Jako další kritérium porovnání přístupů jsem porovnala



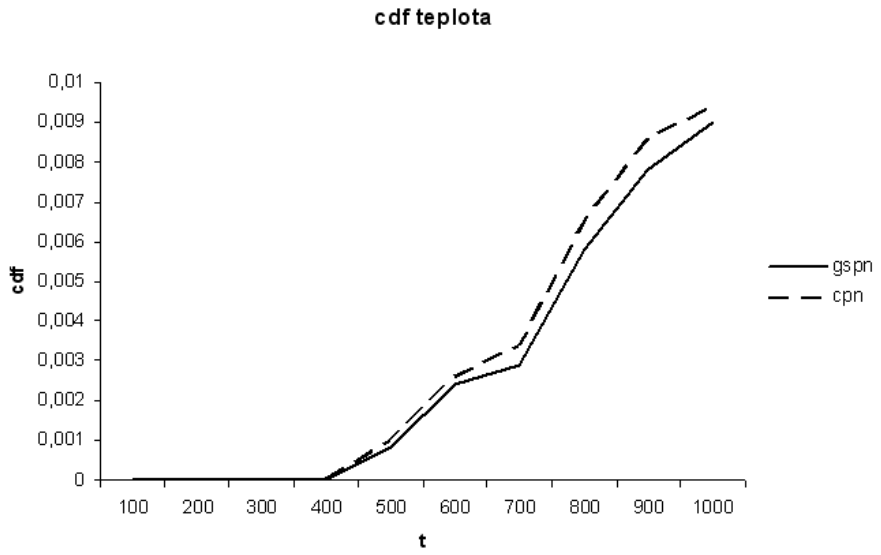
Obrázek 6.20: *PN-model procesu údržby.*



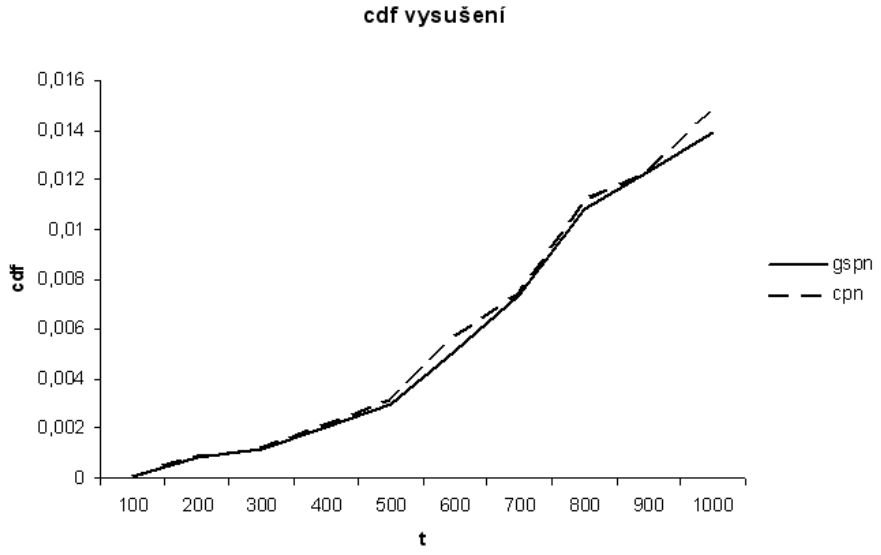
Obrázek 6.21: *GSPN-model: vývoj cdf.*

CPU čas pro simulace. Cílem bylo zjistit, zda původní domněnka, že použití Barevných Petriho sítí bude pro tvorbu modelu výhodnější, je oprávněná. Tabulka 6.10 ukazuje CPU-čas (Intel 2.8 GHz Northwood) Monte Carlo simulace.

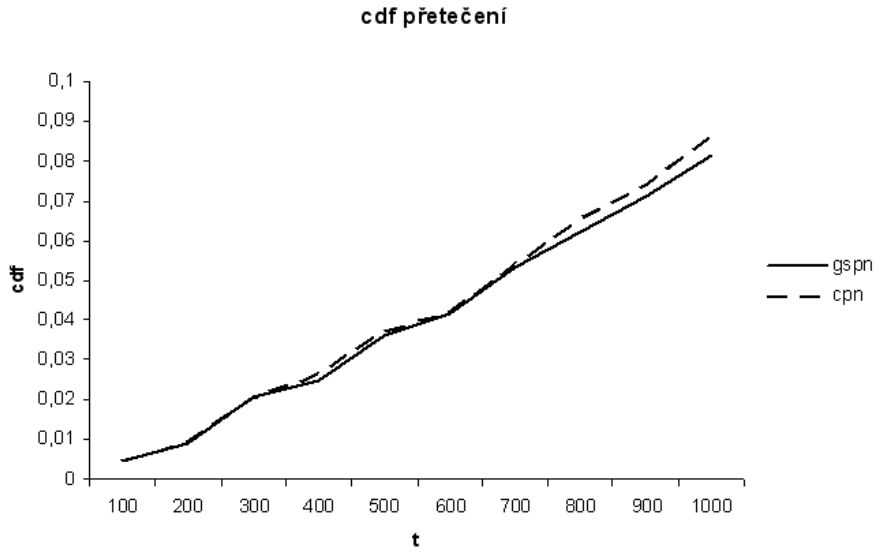
Výsledky jsem postupně prezentovala na konferencích (např. Mathematical Methods in Reliability: Theory, Methods, Applications, 2007) a publikovala např. v [1],[3].



Obrázek 6.22: *Spočtené cdf pro stav teplota u systému se dvěma proměnnými.*



Obrázek 6.23: *Spočtené cdf pro stav vysušení u systému se dvěma proměnnými.*



Obrázek 6.24: *Spočtené cdf pro stav přetečení u systému se dvěma proměnnými.*

počet historií	CPU-čas (GSPN)	CPU-čas(CPN)
10^3	0.94s	0.88s
10^5	1min38s	1min21s

Tabulka 6.10: *CPU-čas Monte Carlo simulace.*

Kapitola 7

Aplikace II. - Řešení reálného dynamického systému

Poslední část mé práce se týká nalezení vhodného matematického modelu pro konkrétní reálný systém. Pro tvorbu modelu a popsání dynamických procesů systému jsem použila nástroje Petriho sítí. V předcházející kapitole jsem ověřila a ukázala, že zvolený přístup bude vhodný. Zkoumaným zařízením je vzduchový kompresor sloužící jako zdroj tlakového vzduchu. Stlačený vzduch je nositelem energie potřebné pro chod zařízení zajišťujících výrobu produktů na výrobních linkách závodu. Kompresorová jednotka byla pro model dynamického systému vybraná záměrně, nebot'je zde jasně patrná zpětná vazba z výrobního procesu na zdroj energie, kde jako medium slouží tlakový vzduch a jasné dynamické reakce na požadované procesní změny parametrů. Důležitou roli zde sehrála i dostupnost požadovaných informací z reálného provozu.

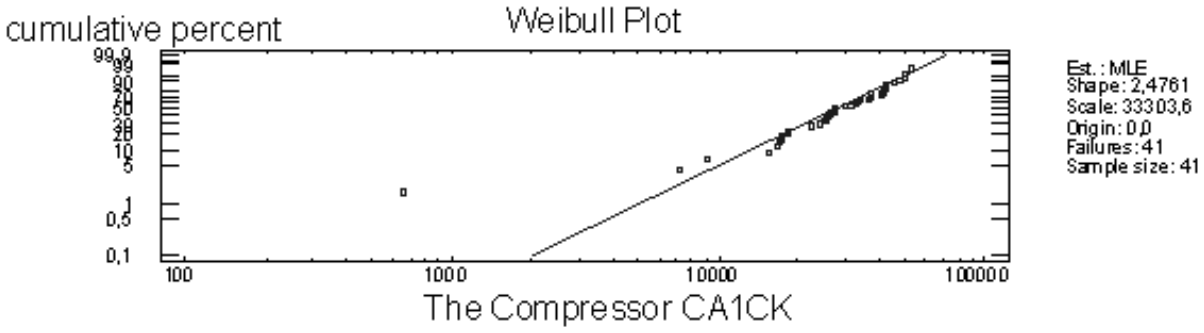
7.1 Popis dynamického systému

Reálný dynamický systém se skládá ze tří základních částí - výrobní, přenosová a spotřebiče vzduchu. Výrobní částí je myšlena kompresorová stanice, přenosová část je systémem potrubí tlakového vzduchu a spotřebičem vzduchu můžou být např. pneumatická vrata, zavírací a otevírací mechanismy, atd.. Pro další postup byly spotřebiče chápány jako vnější vlivy na změny sledované procesní proměnné (tlak v potrubí) a z toho důvodu se dynamický systém skládá pouze ze dvou částí. Pro správnou funkci systému potrubí a kompresorů se pro zavádění modelu dynamického systému vychází z následujících základních předpokladů:

- Systém potrubí pracuje v bezporuchovém stavu - jelikož neslouží k rozvodu agresivních látek, jeho zabezpečení proti možnému přetlaku jsou naddimenzovaná a jiná namáhání než tlakovým pnutím lze vyloučit, z toho důvodu lze potrubní rozvod v modelovém řešení považovat za bezporuchový.
- Poruchovost kompresorů - systém nepracuje ve správném stavu z důvodu nezajištění dodávky tlakového vzduchu správných parametrů z kompresorové stanice.

Současné vytížení kompresorové stanice se pohybuje v mezích 30-35%. K dlouhodobému výpadku by tak mohlo dojít pouze v případě nakumulování několika závažných poruch nebo v případě selhání lidského faktoru. Nejčastějším jevem je pokles tlaku vzduchu v potrubí v důsledku nadměrného odběru nebo náhlým odlehčením ve výrobním procesu. V takovémto případě musí kompresory v kompresorové stanici na základě impulzu z určeného regulačního zařízení okamžitě zareagovat a zkorigovat tlak v potrubí do pracovních mezí. Toto je typický příklad dynamického systému.

Pro analýzu dat je nejdříve nutné vytvořit databázi událostí v softwarovém prostředí. Jako výchozí materiály slouží knihy záznamů servisních prací na jednotlivých kompresorech. Analyzovaný údaj je doba do poruchy kompresoru. Časovou jednotkou jsou tzv. "moto hodiny", které udávají dobu provozu kompresoru. Jestliže u kompresorů vyselektujeme doby do poruchy, můžeme pak v dalším kroku datové analýzy určit hledané rozdělení poruch v čase, což je důležitým faktorem pro vytvoření modelu stárnutí kompresoru. Tato skutečnost se pak bude implementovat do celkového modelu dynamického systému. Po analýze, kdy byla využita metoda maximální věrohodnosti s využitím modifikovaného Kaplan-Meierova odhadu parametrů, bylo zjištěno, že doba do poruchy kompresorů se řídí podle Weibullova rozdělení s parametry: SHAPE = 2,4761 a SCALE = 33 303 (viz Obrázek 7.1).
Další informace o analýze systému je možno najít v [2] nebo (Gala 2007).



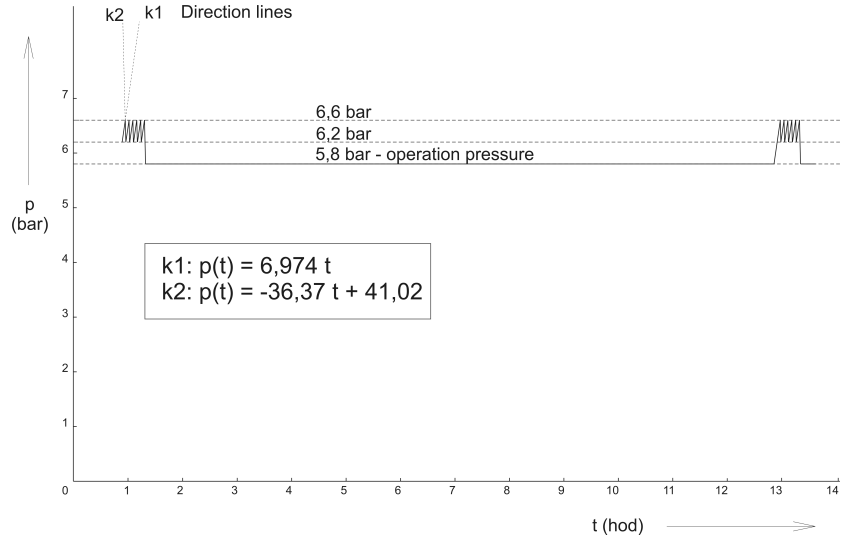
Obrázek 7.1: *Weibullovo rozdělení pro kompresor CA1CK.*

7.2 Modelová zjednodušení

Skutečné hodnoty z reálného systému se zavedou do matematické modelu dynamického systému. Vzhledem ke složitosti a nevyzpytatelnosti reálného systému není

zcela možné věrně zachytit veškeré skutečnosti provozu, což je také důvodem k zavedení určitých zjednodušení v modelovém řešení dynamických jevů. Základním předpokladem pro tvorbu modelu je matematický popis změny tlaku vzduchu v potrubí v závislosti na reálném čase.

Pro modelové řešení byl vybrán stav systému při odlehčené zátěži u nějž byla vysledována periodicita opakování 12 hodin (střídání pracovních směn) s pravděpodobností výskytu dynamických jevů 70% (Obrázek 7.2). To je stav, ve kterém se tlak v potrubí pohybuje v pásmu 6,2 - 6,6 bar, což je až 0,8 bar nad pracovní oblastí. Mezi jednotlivými dynamickými přechody uvažujeme tlak v potrubí konstantní.



Obrázek 7.2: *Dynamické stavy systému.*

7.3 Návrh modelu reálného problému

Shrnutím předcházejících analýz o systému můžu definovat matematický model. Dynamický systém je složen ze dvou komponent - hlavní kompresor (*C1*) a zálohový kompresor (*C2*). Obě komponenty se mohou nacházet v jednom ze tří stavů - zapnutý (ON), vypnutý (OFF) a porouchaný (FAIL). Výskyt poruchy obou komponent se řídí podle Weibullova rozdělení.

Sledovanou procesní proměnnou je tlak vzduchu v systému potrubí (p). Mezní hodnoty procesní proměnné jsem shrnula v Tabulce 7.1.

Uvažuji dva poruchové stavy systému. První nastane, pokud tlak v potrubí klesne pod 5 bar a tuto vrcholovou událost označím jako *podtlak*. Druhou vrcholovou událostí

mezní hodnota	tlak (bar)
horní kritická mez	7
horní mez	6.6
odlehčený stav	6.2-6.6
pracovní oblast	5.8-6
dolní mez	5.5
dolní kritická mez	5

Tabulka 7.1: *Mezní hodnoty.*

je *přetlak* ($p > 7$ bar).

Tabulka 7.2 definuje řídicí pravidla s ohledem na proměnnou p. V případě, že tlak klesne pod 6.2 bar znamená to, došlo k poruše hlavního kompresoru. Zálohový kompresor, který slouží jako studená záloha, se zapne, jakmile je tlak v potrubí roven 5.5 bar a začne se modelovat situace, kdy se tlak pohybuje v rozmezí 5.5-5.8 bar. V této situaci může dojít k *podtlaku*.

Tlak v potrubí (p)	C1	C2
6.6	OFF	OFF
6.2	ON	OFF
5.8	OFF	OFF
5.5	OFF	ON

Tabulka 7.2: *Řídicí pravidla pro řízení kompresorů.*

Pro tvorbu spolehlivostního modelu analyzovaného reálného systému jsem využila Zobecněných stochastických Petriho sítí. Nástroj GSPN jsem zvolila z toho důvodu, že analyzovaný systém není příliš rozsáhlý a v systému jsou pouze dvě komponenty s podobnými vlastnostmi.

Na Obrázku 7.3 je GSPN-model systému. Tabulka 7.3 definuje místa využitá v GSPN-modelu. Ke znázornění aktuální hodnoty tlaku v potrubí jsem využila diskretizaci (místo p obsahuje 0-20 tokenů, viz. Tabulka 7.4). Počáteční stav je tlak v potrubí je roven 6.2 bar (místo p obsahuje 12 tokenů), hlavní kompresor je zapnutý (místo C1_on je označeno) a rezervní kompresor je vypnutý (token je v místě C2_off).

Jelikož modelovaný reálný systém je velmi podobný řešenému benchmarku v kapitole 6, použila jsem obdobný postup při tvorbě modelu. Příkladem může být část

Petriho sítě modelující poruchu hlavní komponenty (Obrázek 7.4). Místa C1_on, C1_off a C1_fail jsou využita pro znázornění stavu, ve kterém se komponenta nachází. Po provedení stochastického přechodu *C1_fail* přejde komponenta *C1* do stavu porouchaná (v místě C1_fail se objeví token). Jakmile dojde k situaci, že hlavní kompresor je porouchaný, dojde k zablokování (pomocí inhibičních hran) okamžitých přechodů *C1_on* a *C1_off*.

místo	popis
C1_on (C2_on)	kompresor C1 (C2) je zapnutý
C1_off (C2_off)	kompresor C1 (C2) je vypnutý
C1_fail (C2_fail)	kompresor C1 (C2) je porouchaný
p	hodnota tlaku v systému potrubí
system_fail	porucha systému

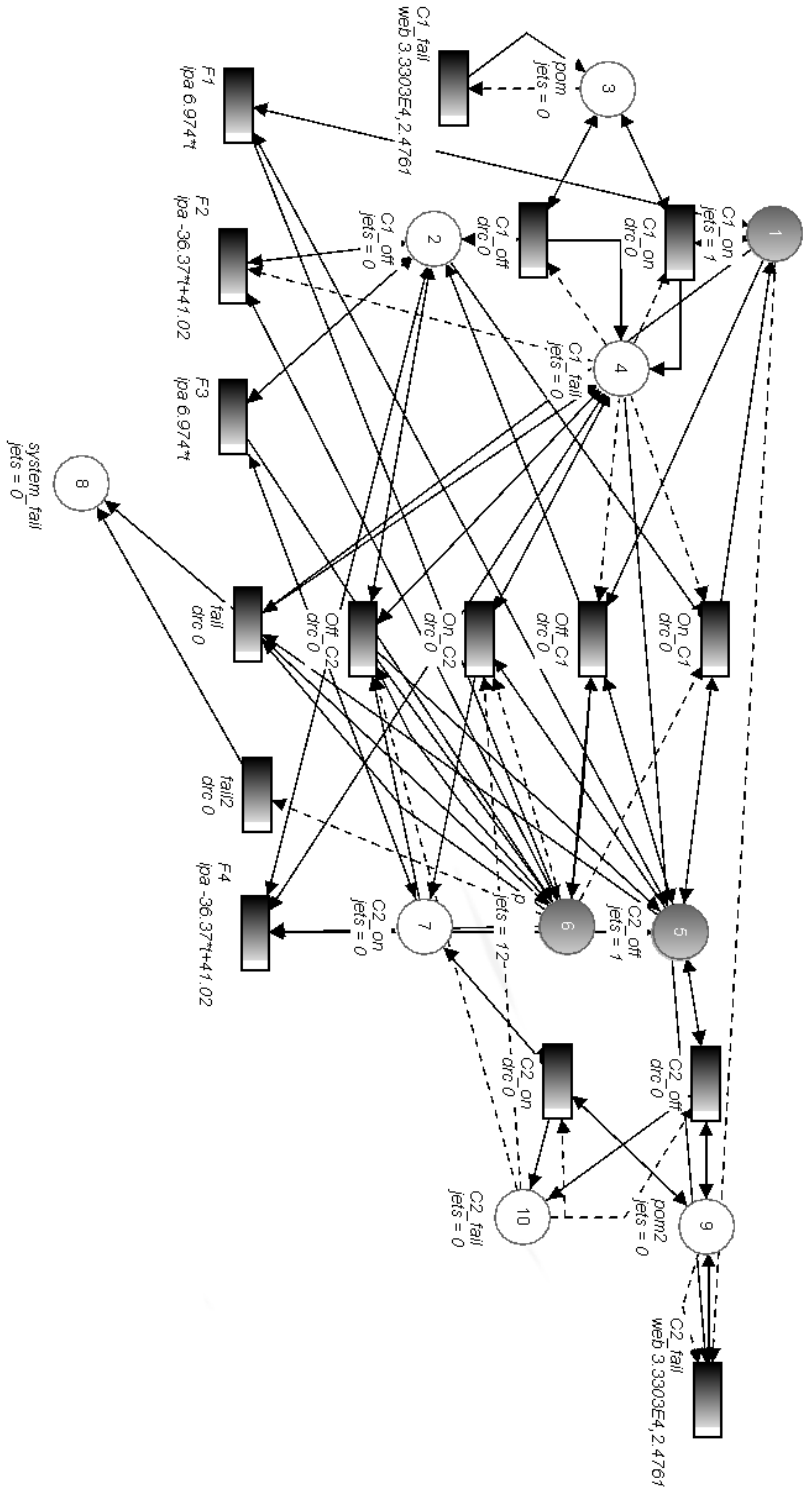
Tabulka 7.3: *Popis míst v GSPN-modelu.*

počet tokenů	p	stav systému
20	>7 bar	přetlak
16	6.6 bar	horní hranice odlehčeného stavu
12	6.2 bar	dolní hranice odlehčeného stavu
0	<5 bar	podtlak

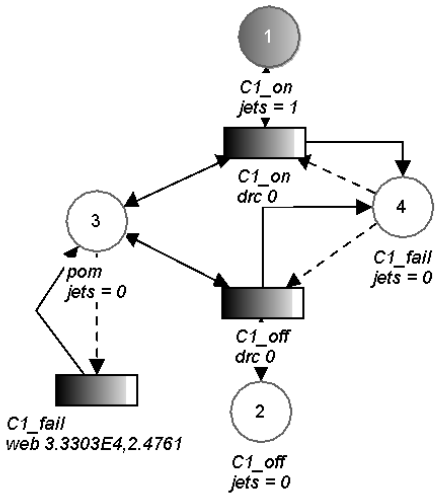
Tabulka 7.4: *Souvislost mezi hodnotou tlaku a počtem tokenů v místě p.*

Řidící akce změny stavu kompresoru s ohledem na hodnotu proměnné p jsem znázornila pomocí dvou okamžitých přechodů pro každou komponentu (např. *On_C1*, *Off_C2*). Při hodnotě p = 6.2 bar (v místě p je počet tokenů 12) je nutné přepnout hlavní kompresor do stavu ON. Jakmile hodnota tlaku dosáhne 6.6 bar, je potřeba snížit tlak v systému potrubí. To docílím vypnutím hlavního kompresoru (přechod *Off_C1*). Touto cestou modeluji odlehčený stav. Během období odlehčené zátěže je zálohový kompresor ve stavu OFF. Zálohový kompresor se zapne až v případě, že dojde k poruše hlavního kompresoru a tlak klesne na 5.5 bar. K modelování této situace jsem použila přechodu *On_C2* a začínám modelovat stav, kdy se tlak v systému potrubí pohybuje v rozmezí 5.5-5.8 bar.

Dynamika systému spočívá ve změně hodnoty tlaku v potrubí. Pro znázornění zmíněné dynamiky jsem použila čtyři časované přechody (*F1, F2, F3, F4*). Efektem



Obrázek 7.3: *GSPN-model reálného systému.*



Obrázek 7.4: *GSPN pro znázornění poruchy kompresoru.*

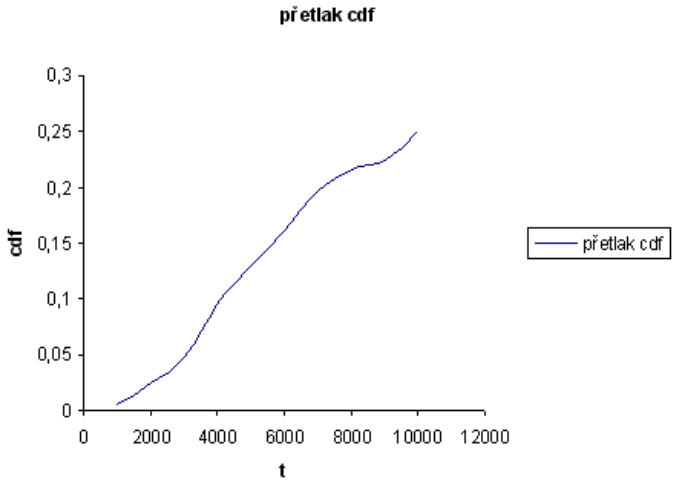
provedení těchto přechodů je změna počtu tokenů v místě p. Po uschopnění přechodů *F1, F3* se počet tokenů v p bude zvyšovat (tlak roste) a při uschopnění přechodů *F2, F4* se počet tokenů v místě p sníží (tlak klesá).

Token v místě system_fail představuje, že nastala jedna z vrcholových událostí a došlo k poruše (selhání) systému. Přechod *fail* znázorňuje situaci, kdy v systému nastane přetlak ($p > 7$bar). *fail* je uschopněn tehdy, jakmile se v místě p objeví 20. token a místa C1_on, C1_fail a C2_off jsou označeny.

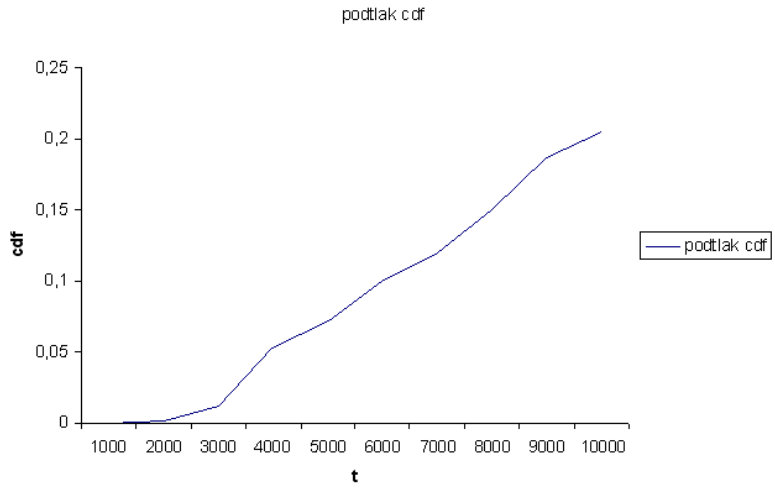
Druhým přechodem, po jehož provedení přejde token do místa system_fail, je přechod nazvaný *fail2*. Tento přechod je uschopněn, pokud místo p neobsahuje žádný token ($p < 5$bar). Došlo k podtlaku a tím i k poruše systému.

7.4 Výsledky simulace

Vytvořený GSPN-model jsem použila jako vstupní schéma pro proces simulace. Cílem simulace bylo určení kumulativní distribuční funkce (cdf) pro nadefinované vrcholové události, vedoucí k poruše systému (přetlak, podtlak). Spočtení cdf pro přetlak (podtlak) znamenalo určit pravděpodobnost přítomnosti tokenu v místě system_fail při nadefinovaném stavu přetlak (podtlak). Nadefinování bylo provedeno pomocí přechodů vedoucí k poruše. Výsledky byly určeny pro simulační čas v rozpětí 0 až 10000 hodin. Získané výsledky jsou graficky znázorněné na Obrázcích 7.5 a 7.6.



Obrázek 7.5: *cdf pro stav přetlaku.*



Obrázek 7.6: *cdf pro stav podtlaku.*

Výsledky byly publikovány např. v časopise Automa [2].

Kapitola 8

Závěr

Většina průmyslových systémů je dynamických a proto hraje studium spolehlivosti dynamických systémů důležitou úlohu. Jeden z největších problémů v oblasti hodnocení rizik je ten, jak efektivně a reálně namodelovat dynamické chování systémů. V současné době je všeobecně známo, že většina metod používaných pro tuto úlohu je nevhodných v případech, kdy fyzikální chování systémů nemůže být odděleno od pravděpodobnostního chování, což je právě případ systémů s dynamickými procesy. Jednou z možných metod pro modelování vztahů mezi vývojem procesních proměnných a stochastickými změnami jsou stochastické Petriho sítě.

V práci byly dosaženy následující hlavní výsledky:

1. Ověření vhodnosti použití Petriho sítí na popis dynamických systémů:
   - analýza dynamického benchmarku s jednou proměnnou a vytvoření modelu s využitím barevných Petriho sítí
   - návrh údržby, tvorba CPN-modelu
   - přidání druhé proměnné a vytvoření CPN a GSPN modelů modifikovaného systému
   - použití vytvořených modelů jako vstupní schémata do simulací s cílem určení kumulativní distribuční funkce
   - porovnání výsledků u obou použitých typů Petriho sítí
2. Aplikace nástroje Petriho sítí na konkrétní reálný problém:
   - analýza dynamického systému
   - definice modelových zjednodušení
   - vytvoření modelu systému s využitím Zobecněných Stochastických Petriho sítí
   - simulace modelu a spočtení cdf

Zjistila jsem, že pro určité typy systémů je vhodné využití Barevných Petriho sítí z důvodů větší kompaktnosti a celkové přehlednosti vytvořeného modelu. Důležitou úlohou v teorii spolehlivosti je i problém údržby. I tento problém jsem úspěšně vyřešila a namodelovala dvěma přístupy. Z analýzy výsledků vyplynulo, že vhodnějším přístupem je použití preventivní údržby. Získané zkušenosti jsem aplikovala na konkrétní reálný problém.

Dosažené výsledky byly během mého doktorandského studia publikovány nejen ve sbornících řady mezinárodních i národních konferencí, ale také v časopise Journal of Risk and Reliability a v časopise Automa. Zvláště prvně jmenovaný patří mezi jeden z nejprestižnějších časopisů v oboru teorie spolehlivosti a publikuje zde celá řada předních odborníků. V současné době připravuji další dvě publikace na mezinárodní konference (ESREL 2009, ICCC 2009).

Tento výzkum byl podporován grantovým projektem AV ČR (grant číslo T401940412).

Literatura

Acosta, C., S. N.: 1993, Dynamic event trees in accident sequence analysis: Application to steam generator tube rupture, *Reliab. Eng & System Safety* **41**, 135–154.

Aldemir, T.: 1987, Computer-assisted markov failure modeling of process-control systems, *IEEE Transactions On Reliability* **36**(1), 133–149.

Aldemir, T.: 1991, Utilization of the cell-to-cell-mapping technique to construct markov failure models for process control systems, *Probabilistic Safety Assessment and Management PSAM1-New York Elsevier* pp. 1431–1436.

Aldemir, T.: 1994a, Dynamic approaches - applications: An overview, *Reliability and Safety Assessment of Dynamic Process Systems* **120**, 81–84.

Aldemir, T., B. M. D. L.: 1996, Process reliability and safety under uncertainties, *Reliab. Eng & System Safety* **52**, 211–225.

Aldemir, T., S. N. M. A.-C. P. C. G. B. G.: 1994b, Reliability and safety assesment of dynamic process system nato-asi series f, *Springer-Verlag* .

Amendola, A., R. G.: 1984, Dylam-1, a software package for event sequence and consequence spectrum methodology, *ISPRA: Commission of the European Communities* .

Andrews, J. D., D. J. B.: 1999, Dependency modeling using fault-tree analysis, *Proceedings of the 17th International System Safety Conference* pp. 67–76.

Balakrishman, M., T. K.: 1996, Stochastic petri nets for reliability analysis of communication network applications with alternate routing, *Reliab. Eng & System Safety* **53**.

Balbo, G., C. G. F. G.-R. G. M.: 1987, Generalized stochastic petri nets for the performance evaluation of fms, *Proc. Int. Conf. on Robotics and Automation* pp. 1013–1018.

Belhadj, M., H. M. A. T.: 1992, On the need for dynamic methodologies in risk and reliability studies, *Reliab. Eng & System Safety* **38**, 219–236.

Cepin, M., M. B.: 2001, A dynamic fault-tree, *Reliab. Eng & System Safety* **75**, 93–91.

Chabot, J. L.: 1998, Hybrid monte-carlo simulation using petri-nets and fire code for analysing fire scenarios in nuclear power plants, *Proceedings of the PSAM 4* .

Chatelet, E., C. J. L. D.-Y.: 1998, Event representation in dynamic reliability analysis using stochastic petri nets, *Proceedings of the Fifth International Workshop on Dynamic Reliability: Future Directions* .

Cherkasova, L., K. V. R. T.: 1993, On net modelling of industrial size concurrent systems, *Proceedings of ICATPN´93,LNCS 691* pp. 552–561.

Christensen, S., J. J. N.: 1997, Analysis of bang and olufsen's beolink audio/video system using coloured petri nets, *Proceedings of ICATPN´97,LNCS 1248* pp. 387–406.

Clarke, E. M., E. E. A.-S. A. P.: 1986, Automatic verification of finite state concurrent systems using temporal logic, *ACM Transactions on Programming Languages and Systems* **8**(2), 244–263.

Cojazzi, G.: 1996, The dylam approach for the dynamic reliability analysis of systems, *Reliab. Eng & System Safety* **52**(3), 279–296.

Cojazzi, G., I. J. M. M.-E. S.-P. M.: 1992, The reliability and safety assesment of protection systems by the use of dynamic event trees (det). the dylam-treta package, *Proc. XVIII annaul meeting Spanish Nuclear Society* .

Cordier, C., F. M. L. A.-P. A.: 1996, Integration of process simulations in availability studies, *Reliab. Eng & System Safety* **55**, 106–116.

Cox, D.: 1962, Renewal theory, *Chapman & Hall* .

Deoss, D. L., S. N.: 1989, A simulation model for dynamic system availability analysis, *Massachusetts Institute of Technology* .

Devooght, J., S. C.: 1992a, Probabilistic reactor dynamics 3. a framework for time-dependent interaction between operator and reactor during a transient involving human error, *Nucl.Sci. Engng* **112**(2), 101–113.

Devooght, J., S. C.: 1992b, Probabilistic reactor dynamics i. the theory of continuous event trees, *Nucl.Sci. Engng* **111**(3), 229–240.

Devooght, J., S. C.: 1996, Probabilistic dynamics as a tool for dynamic psa, *Reliab. Eng & System Safety* **52**, 185–196.

Dutuit, Y., C. E. S. J. P. T.-P.: 1997, Dependability modelling and evaluation by using stochastic petri nets: Application to two test cases, *Reliab. Eng & System Safety* **55**, 117–124.

Floreani, D. J., B. J. D. A.: 1996, Designing and verifying a communications gateway using coloured petri nets and design/cpn, *Proceedings of ICATPN´96. LNCS 1091* .

Florin, G., N. S.: 1985, Les reseaux de petri stochastiques, *Technique et Science Informatiques* **4**(1).

Fragola, J. R.: 1995, Probabilistic risk assessment of the space shuttle, *No. SAIC doc. no. SAICNY95-02-05* .

Gala, J.: 2007, Data analysis for a real dynamic system., *Electrical Power Engineering* .

Gardiner, C. W.: 1985, Handbook of stochastic methods.

Garrett, C. J., A. G. E.: 2002, Automated hazard analysis of digital control systems, *Reliab. Eng & System Safety* **77**, 1–17.

Genrich, H. J., S. R. M.: 1992, Formal verification of an arbiter cascade, *Proceedings of ICATPN´92* pp. 205–223.

Gerzson, M., H. K. M.: 1995, Analysis of controlled technological systems using high level petri nets, *Proceedings of the ESCAPE/95 Conference BLED* pp. 531–536.

Goddard, P. L.: 1996, A combined analysis approach to assessing requirements for safety critical real-time control systems, *Hughes Aircraft Company, IEEE Proceedings Annual Reliability Maintainability Symposium* .

Gribaudo, M.: 2003, Fluid petri nets and hybrid model-checking: a comparative case study, *Reliab. Eng & System Safety* **81**, 269–280.

Guarro, S., Y. M. O. S.: 2004, Conditional risk model concept for critical space systems software, *Probabilistic Safety Assessment and Management: PSAM7-ESREL/04* pp. 158–163.

Holzmann, G. J.: 1991, Design and validation of computer protocols, *Prentice-Hall International Editions* .

Houtermans, M., A. G. B. A. K. D.: 2000, Programmable electronic system design & verification utilizing dfm, *Proceedings of Computer Safety, Reliability And Security* **1943**, 275–285.

Houtermans, M., A. G. B. A. K. D.: 2002, The dynamic flowgraph methodology as a safety analysis tool: programmable electronic system design and verification, *Safety Science* **40**, 813–833.

Huber, P., J. A. M. J. L. O.-J. K.: 1986, Reachability trees for high-level petri nets, *Theoretical Computer Science* **45**, 261–292.

Huber, P., P. V. O.: 1991, A formal executable specification of the isdn basic rate interface, *Proceedings of ICATPN´91* pp. 1–21.

Izquierdo, J., L. P. E.: 2004, The stimulus-driven theory of probabilistic dynamics as a framework for probabilistic safety assessment, *Proceedings of the PSAM 7 , ESREL* .

Izquierdo, J. M., H. J. S.-P. M.-M. M.: 1994, Automatic generation of dynamic event trees: A tool for integrated safety assesment (isa), *Springer-Verlag* .

Izquierdo, J. M., M. E. D.-J.: 1996, Relationship between probabilistic dynamics and event trees, *Reliab. Eng & System Safety* **52**, 197–209.

Jensen, K.: 1997, Coloured petri nets. volumes: Basic concepts, analysis methods, and industrial case studies, *Monographs in Theoretical Computer Science, Springer-Verlag* .

Jensen, K.: 1998, An introduction to the practical use of coloured petri nets, *Lectures on Petri Nets II, LNCS 1492, Springer-Verlag* pp. 237–292.

Jensen, K., R. G.: 1991, High-level petri nets: Theory and application, *Springer-Verlag* .

Kae-Sheng, H., M. A.: 1996, The development and application of the accident dynamic simulator for dynamic probabilistic risk assessment of nuclear power plants, *Reliab. Eng & System Safety* **52**, 297–314.

Kaplan, S., G. B. J.: 1981, On the quantitavive definition of risk, *Risk Analysis* **1**, 11–27.

Kaufman, L. M., B. S. J.-B. W.: 2000, Modeling of common-mode failures in digital embedded systems, *In Annual Reliability And Maintainability Symposium* pp. 350–357.

Kim, M.C., S. P. H.: 2007, A method for identifying instrument faults in nuclear power plants possibly leading to wrong situation assessment, *Reliab. Eng & System Safety* **10**.

Labeau, P. E.: 1996a, A monte carlo estimation of the marginal distributions in a problem of probabilistic dynamics, *Reliab. Eng & System Safety* **52**, 65–75.

Labeau, P. E.: 1996b, Probabilistic dynamics: Estimation of generalized unreliability throught efficient monte carlo simulation, *Annals of Nuclear Energy* **23**(17), 1355–1369.

Labeau, P. E.: 1998, A survey on monte carlo estimation of small failure risks in dynamic reliability, *Int. J Electron Commun* **3**(52), 205–211.

Larsen, K. G., S. S. W.-C. A.-R. H. T. A.: 1997, Special section on timed and hybrid systems, *Software Tools for Technology Transfer* (1-2).

Lee, S. J., S. P. H.: 2004, Development of automated operating procedure system using fuzzy colored petri nets for nuclear power plants, *Annals of Nuclear Energy* **31**, 849–869.

Liu, T. S., C. S. B.: 1997, The application of petri nets to faiulre analysis, *Reliab. Eng & System Safety* **57**.

Malhotra, M., T. K. S.: 1995, Dependability modeling using petri-nets, *Ieee Transactions On Reliability* **44**, 428–440.

Marchand, S., T. B. L. P.: 1998, Ddet and monte carlo simulation to solve some dynamic reliability problems, *Proceedings of PSAM 4* **3**, 2055–2060.

Marsan, M. A., B. G. C.-G.: 1984a, A class of generalized stochastic petri nets for the performance analysis of multiprocessor systems, *ACM Transactions on Computer Systems* **2**(1).

Marsan, M. A., B. G. C.-G. C.-G.: 1987, Generalized stochastic petri nets revisited: Random switches and priorities, *Proc. Int. Workshop on Petri Nets and Performance Models* pp. 44–53.

Marsan, M. A., B. G. C.-G. D.-S. F. G.: 1995, Modeling with generalized stochastic petri nets, *Series in Parallel Computing. Wiley* .

Marsan, M., C. G.: 1984b, A class of generalized stochastic petri nets for the performance evaluation of multiprocessor systems, *ACM Transactions on Computer Systems* **2**(2).

Marseguerra, M., Z. E.: 1996, Monte carlo approach to psa for dynamic process systems, *Reliab.Eng & System Safety* **52**, 227–241.

Martz, H. F., W. R. A.: 1982, Bayesian reliability analysis, *John Willey & Sons* .

Matsuoka, T.: 2004, Improvement of the go-flow methodology, *Proceedings of the PSAM 7 ESREL* .

Matsuoka, T., K. M.: 1988, Go-flow: A new reliability analysis methodology, *Nuclear Science and Engineering* **98**, 64–78.

Matsuoka, T., K. M.: 1991, An analysis of a dynamic system by the go-flow methodology, *Probabilistic Safety Assessment and Management* **96**, 1547–1436.

McLendon, W. M., V. R. F.: 1992, Analysis of an ada system using coloured petri nets and occurence graphs, *Proceedings of ICATPN´92,LNCS 616* pp. 384–388.

McMillan, K. L.: 1993, Symbolic model checking, *Kluwer Academic Publishers* .

Merlin, P. M., F. D. J.: 1976, Recoverability of communication protocols: Implications of a theoretical study., *IEEE Transactions on Communications* **24**(9), 1036–1043.

Molloy, M. K.: 1982, Performance analysis using stochastic petri nets, *IEEE Transaction on Computers* **31**(9), 913–917.

Mortensen, K. H., P. V.: 1994, Modelling the work flow of a nuclear waste management program, *Proceedings of ICATPN´94,LNCS 815* pp. 376–395.

Murata, T.: 1989, Petri nets: properties, analysis and applications, *Proceedings of the IEEE* **77**(4), 541–580.

NRC: 1975, Reactor safety study: an assessment of accident risks in us commercial nuclear power plants, *N. R. Commission* .

Pai, G., D. S. D. J.: 2002, Estimating software reliability from process and product evidence, *PSAM6: Proceedings of the 6th International Conference on Probabilistic Safety Assessment and Management* **CDROM Version**.

Park, J. H., S. P. H.: 2002, An integrated knowledge base development tool for knowledge acquisition and verification for npp dynamic alarm processing systems, *Annals of Nuclear Energy* **29**.

Peterson, J. L.: 1977, Petri nets, *Computing Surveys* **9**(3).

Peterson, J. L.: 1981, Petri net theory and the modeling of systems, *Englewood Cliffs, NJ:Prentice-Hall* .

Petri, C. A.: 1966, Communication with automata, *Technical Report RADC - TR-65-377, NY* .

Pinci, V. O., S. R. M.: 1991, An integrated software development methodology based on hierarchical coloured petri nets, *Proceedings of ICATPN´91,LNCS 524* pp. 227–252.

Raiteri, D. C., B. A.: 2005, Solving dynamic reliability problems by means of ordinary and fluid stochastic petri nets., *proceedings of ESREL* pp. 381–389.

Ramchandani, C.: 1974, Analysis of asynchronous concurrent systems by timed petri nets, *Ph.D. thesis, Cambridge, MA* .

Rasmussen, J. L., S. M.: 1996, Designing a security system by means of coloured petri nets, *Proceedings of ICATPN´96,LNCS 1091* pp. 400–419.

Rauzy, A.: 2002, Mode automata and their compilation into fault trees, *Reliab. Eng & System Safety* **78**.

Scheschonk, G., T. M.: 1994, Simulation and analysis of a document storage system, *Proceedings of ICATPN´94,LNCS 815* pp. 454–470.

Schoeniga, R., A. J. F. C.-T. H.-T.: 2006, An aggregation method of markov graphs for the reliability analysis of hybrid systems, *Reliab. Eng & System Safety* **91**.

Shapiro, R. M.: 1991, Validation of a vlsi chip using hierarchical coloured petri nets, *Journal of Microelectronics and Reliability* .

Shooman, M. L.: 1968, Probabilistic reliability: an engineering approach, *Mc Graw Hill* .

Sifakis, J.: 1978, Petri nets for performance evaluation, *Proc. 3rd Intern. Symp. IFIP* pp. 75–93.

Siu, N.: 1994a, Dynamic approaches , issues and methods: An overview, *Reliability and Safety Assessment of Dynamic Process Systems* **120**, 3–7.

Siu, N.: 1994b, Risk assesment for dynamic systems: An overwiev, *Reliab. Eng & System Safety* **43**(1), 43–73.

Smith, D. T., D. T. A.-J. B. W.: 2000, A safety assessment methodology for complex safety-critical hardware/software systems, *International Topical Meeting on Nuclear Plant Instrumentation, Controls, and Human-Machine Interface Technologies, Washington, DC* .

Son, H. S., S. P. H.: 2003, Development of a safety critical software requirements verification method with combined cpn and pvs: a nuclear power plant protection system application, *Reliab. Eng & System Safety* **80**, 19–32.

Stamatelatos, M., A. G. D. H.-E. C. G. S. M. P.: 2002, Probabilistic risk assessment procedures guide for nasa managers and practitioners, *NASA (Ed.)* .

Swaminathan, S., S. C.: 1999a, The event sequence diagram framework for dynamic probabilistic risk assessment, *Reliab. Eng & System Safety* **63**, 73–90.

Swaminathan, S., S. C.: 1999b, Identification of missing scenarios in esds using probabilistic dynamics, *Reliab. Eng & System Safety* **66**, 275–279.

Swaminathan, S., S. C.: 1999c, The mathematical formulation for the event sequence diagram framework, *Reliab. Eng & System Safety* **65**, 103–118.

Symons, F. J.: 1978, Modeling and analysis of communication protocols using petri nets, *Ph.D. thesis, University of Essex* .

Tombuyses, B.: 1999, Reduction of the markovian system by the influence graph method: error bound and reliability computation, *Reliab.Eng & System Safety* **63**, 1–11.

Tombuyses, B., A. T.: 1996, Dynamic psa of process control-systems via continuous cell-to-cell-mapping, *Probabilistic Safety Assessment and Management PSAM3-New York Elsevier* pp. 1541–1546.

Vernez, D., B. D. P. G.: 2003, Perspectives in the use of coloured petri nets for risk analysis and accident modelling, *Safety Science* **41**, 445–463.

Volovoi, V.: 2004, Modeling of system reliability petri nets with aging tokens, *Reliab. Eng & System Safety* **84**, 149–161.

Zhang, Y., G. M. M.: 2002, Development of a method for quantifying the reliability of nuclear safety-related software, *PSAM6: Proceedings of the 6th International Conference on Probabilistic Safety Assessment and Management* **CDROM Version**.

Publikace autora

[1] Škňouřilová, P., Briš, R.: 2008, Coloured Petri Nets and a dynamic reliability problem, *Journal of Risk and Reliability*,Vol 222,No O4, ISSN 1748-006X, DOI: 10.1243/1748006XJRR155, 635-642

[2] Gala, J., Škňouřilová, P.: 2008, Dynamický systém a jeho analýza z hlediska spolehlivosti , *časopis Automa*, Vol 12, ISSN 1210-9592

[3] Škňouřilová, P., Briš, R.: 2008, GSPN system for problem with two process variables, *Proceedings of 9th ICCC 2008*, Sinaia, Rumunsko, ISBN 978-973-746-897-0, 619-622

[4] Briš, R.,Škňouřilová, P.: 2008, Statistics I., *Scriptum*, Ostrava, Česká Republika

[5] Schreiber, J., Sojka, E., Ličev, L., Škňouřilová, P., Gaura, J., Školoudík, D.: 2008, A new method for the detection of brain stem in transcranial ultrasound images, *Proceedings of Biosignals 2008*, Funchal, Madeira-Portugalsko, 478-483

[6] Škňouřilová, P., Briš, R.: 2007, Modeling of dynamic reliability problem by Coloured Petri Nets, *Proceedings of 8th ICCC 2007*, Štrbské pleso, Slovensko, ISBN 978-80-8073-805, 701-704

[7] Škňouřilová, P., Briš, R.: 2007, CPN and reliability problem, *Proceedings of Mathematical Methods in Reliability - CDROM version*, Glasgow, UK

[8] Škňouřilová, P.: 2007, A Benchmark on Dynamic Reliability: Analysis of the modeled system, *Proceedings of WOFEX 2007 Ph.D. Workshop of Faculty of Electrical Engineering and Computer Science*, Ostrava, Česká Republika, ISBN 978-80-248-1571-8, 219-225

[9] Škňouřilová, P., Briš, R.: 2006, Using Colored Petri Nets to solve a dynamic reliability problem with two process variables, *Proceedings of ESREL 2006*, Estoril, Portugalsko, ISBN 0-415-41620-5, 173-179

[10] Škňouřilová, P.: 2006, Solving of a dynamic reliability problem by the help of Colored Petri Nets, *Proceedings of WOFEX 2006 Ph.D. Workshop of Faculty of Electrical Engineering and Computer Science*, Ostrava, Česká Republika, ISBN 80-248-1152-9, 282-287

[11] Škňouřilová, P., Briš, R.: 2006, Užití Barevných Petriho sítí k řešení benchmarkovských dynamických systémů, *Sborník konference Statistické dny v Brně*, Ostrava, Česká Republika, ISBN 80-214-3214-4

[12] Škňouřilová, P.: 2005, Relokační metoda pro řešení variačních nerovnic, *Sborník konference Olomoucké dny aplikované matematiky*, Olomouc, Česká Republika

Účast v grantových projektech

Grantový projekt Akademie věd ČR pod číslem T401940412, Kvantifikace a Modelování dynamické spolehlivosti.

Index