FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Výpočty nehomogenních spolehlivostních modelů
Student:	Bc. Jan Řezníček
Vedoucí:	Ing. Martin Kohlík, Ph.D.
Studijní program:	Informatika
Studijní obor:	Návrh a programování vestavných systémů
Katedra:	Katedra číslicového návrhu
Platnost zadání:	Do konce letního semestru 2019/20

Pokyny pro vypracování

1. Seznamte se s metodami pro výpočty nehomogenních spolehlivostních modelů (modelů s proměnlivými intenzitami poruch).
2. Realizujte vybrané metody v SW Wolfram Mathematica.
3. Porovnejte realizované metody a jejich časové náročnosti a přesnosti v závislosti na volbě vstupních parametrů.
4. Vyhodnoťte vhodnost metod z hlediska přínosu pro výzkum v rámci výzkumné skupiny DDD.

Seznam odborné literatury

Dodá vedoucí práce.

doc. Ing. Hana Kubátová, CSc.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 23. listopadu 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Výpočty nehomogenních spolehlivostních modelů

Bc. Jan Řezníček

Katedra číslicového návrhu
Vedoucí práce: Ing. Martin Kohlík, Ph.D.

6. května 2019

Poděkování

Rád bych poděkoval Fakultě informačních technologií Českého vysokého učení technického v Praze za možnost studia informačních technologií, pro které mi tato škola přijde jako nejlepší volba. Také bych rád poděkoval svému vedoucímu diplomové práce, Ing. Martinovi Kohlíkovi, Ph.D, za vstřícnost a ochotu při práci a konzultacích k této práci. V neposlední řadě bych chtěl poděkovat své rodině, přítelkyni i kamarádům, kteří mi byli po celou dobu studia oporou a mohl jsem se na ně ve všem spolehnout.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, avšak pouze k nevýdělečným účelům. Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 6. května 2019 .....

Odkaz na tuto práci

Řezníček, Jan. *Výpočty nehomogenních spolehlivostních modelů.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

V této diplomové práci je popsána metoda výpočtu spolehlivosti a distribuční funkce (selhání) Markovských řetězců s nehomogenními parametry (intenzitami poruch proměnnými v čase) a její implementace v programu Wolfram Mathematica. Díky možnosti práce s nehomogenními modely umí také pracovat i s jinými rozděleními, které se ve spolehlivosti běžně používají, než jen exponenciálním. Metoda je poté otestována a porovnána s existujícími metodami s ohledem na časovou náročnost a přesnost. Toto testování probíhalo nejen na volbě vstupních parametrů, ale i na porovnání s homogenními Markovskými řetězci. V rámci práce byl také sepsán přínos této metody z pohledu spolehlivostních výpočtů a možnosti pokračování práce v rámci dalšího výzkumu.

Klíčová slova Algoritmus, distribuční funkce, Markovské řetězce, nehomogenita, spolehlivost, Wolfram Mathematica.

Abstract

This diploma thesis deals with describing the calculation method of reliability and distribution function of Markov Chains with nonhomogeneous parameters. Implementation is programmed in Wolfram Mathematica. The method can be used to perform calculations on several types of distributions (not only the exponential one) due to its ability to model nonhomogeneous systems. The method is tested and compared with existing methods with respect to time and accuracy. This testing was focused on the impact of input parameters and comparison with homogeneous Markov Chains. The work also describes the contribution of this method from the perspective of reliability calculations and the possibility of the future work.

Keywords Algorithm, Dependability, Distribution Function, Markov Chains, Non-homogenity, Reliability, Wolfram Mathematica.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V dnešní době jsme všichni obklopeni různými technickými zařízeními, které nám usnadňují život a také bychom si bez nich již život nedokázali představit a bereme je jako samozřejmost. Ať už jde o malá zařízení jako například mobilní telefon, který v dnešní době tvoří nedílnou součást společenské interakce a za kterou je spousta osob ochotná utrácet neskutečné částky. Ale samozřejmě jsou zde zařízení, která nám usnadňují život, jako jsou například dopravní prostředky městské hromadné dopravy, vnitrostátní i mezinárodní doprava a všechny typy strojů, které tyto služby zajišťují.

Všechna tato zařízení jsou většinou tvořena jednotlivými součástkami, které spolu spolupracují a vytváří funkční celek, který my vidíme navenek. Ale téměř žádné zařízení není tvořené jen jednotlivou součástkou.

Každá součástka a také celkové zařízení se vyznačuje nějakou funkčností a životností. Žádné zařízení není přímo nesmrtelné. Veškerá zařízení se vyznačují tzv. spolehlivostí, tedy dobou, po kterou jsou schopna fungovat dobře a pracovat tak, jak je popsáno v jejich specifikaci.

Tyto jednotlivé spolehlivosti zařízení se dají odhadnout, jelikož se jedná o statistické odhady na základě testování a práce s nimi. Většina používaných statistických metod však předpokládá, že se jedná o stálou intenzitu určitých poruch, tedy zanedbávají časovou proměnlivost těchto intenzit.

Cílem této diplomové práce je seznámit se s těmito metodami, které se běžně používají pro výpočet homogenních spolehlivostních modelů, tedy modelů s neměnnými intenzitami poruch v čase a vytvořit algoritmus výpočtů těchto modelů tak, aby je bylo možné použít i pro nehomogenní modely. Tento algoritmus se poté otestuje a porovná se časová náročnost a přesnost na základě zvolených vstupních parametrů.

V rámci výzkumné skupiny DDD Katedry číslicového návrhu je výzkum spolehlivosti prováděn dlouhodobě. Tato práce navazuje na obhájenou disertační práci mého vedoucího, Ing. Martina Kohlíka, Ph.D., která se zabývala hrubým odhadem spolehlivostních parametrů komplexních systémů, tj. takových, u kterých už je přímý výpočet parametrů buď dost komplikovaný nebo

v přiměřeném čase nemožný. Cílem mé diplomové práce je rozšířit a především zpřesnit metody z dizertační práce mého vedoucího. Na základě otestování těchto metod se vyhodnotí přínos pro budoucí výzkum v rámci výzkumné skupiny DDD v rámci katedry, na které je diplomová práce psána.

Diplomová práce se skládá z šesti kapitol. První kapitolu tvoří úvod. Druhá kapitola popisuje teoretický základ spolehlivostní analýzy systémů. Vymezuje základní pojmy a definice v nezbytném rozsahu pro tuto práci. Dále jsou zde představeny homogenní a nehomogenní Markovské řetězce (tzn. řetězce s konstantní, resp. nekonstantní intenzitou poruch). Nakonec je na jednoduchém příkladu ukázáno, které výpočetní rovnice se v práci používají. Třetí kapitola se zabývá návrhem a implementací mnou navrženého algoritmu sloužícího k řešení Markovských řetězců v matematickém programu Wolfram Mathematica. V první části této kapitoly je popsán návrh algoritmu a příprava vstupních dat. Ve druhé části jsou popsány jednotlivé programové funkce matematického systému nutné pro vlastní implementaci algoritmu. V závěrečné části kapitoly Implementace je uveden příklad použití algoritmu – tedy určení a vykreslení grafu distribuční funkce $F(t)$. Čtvrtá kapitola se zabývá testováním navrženého algoritmu. Pro účely testování byly použity dva Markovské řetězce převzaté z obhájené dizertační práce vedoucího této práce. Na těchto modelech byla dlouhodobě testována přesnost vypočtených parametrů včetně velikosti chyby. Pátá kapitola obsahuje rozbor výsledného hodnocení kapitoly Testování. Je zde provedeno vyhodnocení jednotlivých parametrů použitých v algoritmu. Následuje diskuse nad možným použitím algoritmu pro vědecké účely. Poslední kapitola je závěrem diplomové práce a shrnuje veškeré dosažené cíle v této diplomové práci. Součástí práce jsou dvě přílohy a CD se zdroji.

Kapitola 1

Teorie

V rámci celé diplomové práce se pracuje s tzv. spolehlivostí, Markovskými řetězci spolehlivosti, výpočty distribuční funkce (funkce selhání) a mnoha dalšími. V první řadě je vhodné si tyto pojmy vysvětlit, aby byl další postup v práci srozumitelnější.

1.1 Spolehlivost

Nejprve je potřeba si definovat, co pro nás spolehlivost značí:

Spolehlivost je obecná vlastnost objektu spočívající ve schopnosti plnit požadované funkce při zachování hodnot stanovených provozních ukazatelů v daných mezích a v čase podle stanovených technických podmínek.
(ČSN EN 62347: 2007 (01 0696)) [1]

Každé zařízení, kterých je kolem nás v dnešní době již nepřeberné množství (například PC, mobilní telefony, semafory, vlaky, ...), má v rámci svého používání určitou spolehlivost, která se mění v čase a je závislá na určitých vlivech. Nejčastější typy zhoršení spolehlivosti se dělí na tři typy [2]:

- *Zahořování* – například chyby výrobních technologií, které se projeví již na začátcích funkčnosti systému, ale zároveň je lze včas odhalit, a proto se v praxi neuvažují.
- *Stárnutí* – například degradace materiálu, elektromigrace.
- *Externí vlivy* – například kosmické záření, přirozená radiace materiálu (vyzařování pouzdra).

Proto je nutné tyto spolehlivostní vlastnosti (například bezporuchovost) počítat, respektive měřit, aby mohla být zaručena tzv. bezpečnost. Zároveň musíme být schopni navrhnout takový systém, který splňuje předepsané parametry. Z tohoto důvodu vznikl tzv. SIL (Safety Integrity Level), který rozděluje určitá zařízení do daných kategorií podle spolehlivosti [2].

Je více možností, jak přistupovat ke spolehlivosti zařízení.
Dva základní přístupy jsou:

1. Odolnost proti poruchám (fault avoidance)
   - Návrh systémů, které minimalizují možnost výskytu poruchy
   - Řešení: například robustnost – odolný HW
2. Tolerance poruch (fault tolerance)
   - Návrh systémů, které se dokáží vypořádat s poruchami
   - Řešení: tzv. redundance
     - HW – plocha navíc + informační redundance (detekční, samoopravné kódy)
     - SW – redundantní kód
     - čas – opakování výpočtů

Spolehlivost (anglicky Dependability) může být kombinovaná více aspekty:

- Spolehlivost (Reliability) – schopnost vykonávat požadovanou funkci
- Dostupnost (Availability) – schopnost fungovat, když je to potřeba
- Bezpečnost (Safety) – schopnost neohrozit životy, majetek a další
- Udržovatelnost (Maintainability) – možnost opravy v případě poruchy nebo dle potřeby
- Integrita (Integrity) – nemožnost samovolných nežádoucích změn

V praxi musíme umět predikovat poruchy a vypočítat jejich četnosti. Základním pilířem těchto výpočtů je statistika. Mimo jiné se používají statistiky z náhodných událostí a na základě jejich výskytu jsou vytvářeny statistické modely. Dále se pak vytvářejí modely poruch/chyb/selhání, které se analyzují na různých úrovních nebo se vytváří hierarchické modely. Dále je v praxi nutné řízení této spolehlivosti a případné její zlepšování (například redundancí).

1.1.1 Typy spolehlivých zařízení

Spolehlivé zařízení můžeme klasifikovat na základě dvou parametrů – možnosti opravy a kritičnosti.

1.1.1.1 Klasifikace dle možnosti opravy

Na základě této klasifikace rozlišujeme dva typy systémů – obnovované (repairable systems) a neobnovované (non-repairable systems) systémy.

Obnovované systémy se v případě poruchy mohou opravit, případně vyměnit → je vyžadován nějaký způsob údržby. Jedná se například o běžné počítače, telefonní a bankovní systémy, veřejnou dopravu a mnoho dalších, v podstatě o většinu běžných zařízení.

Neobnovované systémy se v případě poruchy nedají ani opravit ani vyměnit → jedná se o jednorázová zařízení bez vnějšího přístupu. Mezi ně například patří zařízení pro vesmírné mise nebo zařízení v nedostupných prostorách.

1.1.1.2 Klasifikace dle kritičnosti

V rámci této klasifikace dělíme systémy na tři typy – nekritické, kritické na bezpečnost a kritické na spolehlivost.

Nekritické systémy jsou takové, u kterých selhání nezpůsobí velké škody (např. běžné PC, spotřebiče, ...).

Systémy kritické na bezpečnost případnou poruchu buďto tolerují nebo se daný systém uvede do bezpečného stavu (například automobil, letadlo, vlakové návěští – v případě poruchy se trvale rozsvítí červená z důvodu bezpečnosti).

Systémy kritické na spolehlivost jsou taková zařízení, která nesmí selhat. Typicky se jedná o neobnovované systémy (např. medicínské aplikace – kardiostimulátor, vesmírné mise).

1.1.2 Spolehlivostní parametry

Dle publikace [3] je distribuční funkce (selhání) definována jako pravděpodobnost, že náhodné ohodnocení proměnné nepřesáhne hodnotu t, neboli:

$$F(t) = \int_{-\infty}^{t} f(t)\,dt \tag{1.1}$$

kde $f(t)$ je pravděpodobnostní funkce náhodné proměnné času do poruchy. $F(t)$ pak v našem případě značí pravděpodobnost selhání do času t. Pokud je náhodná proměnná diskrétní, lze integrál v rovnici nahradit sumou. Jelikož předpoklad pro funkci $F(t)$ je takový, že se funkce rovná 0 dokud není $t = 0$, integrál lze použít až od 0 do t.

Spolehlivostní funkce $R(t)$ nebo také pravděpodobnost, že zařízení se neporouchá do času t lze vypočítat jako:

$$R(t) = 1 - F(t) = \int_{t}^{\infty} f(t)\,dt \tag{1.2}$$

Intenzita, se kterou blok selže v intervalu Δt, tzv. intenzita selhání, $\lambda(t)$, je definována následovně:

$$\lambda(t) = \frac{R(t) - R(t + \Delta t)}{\Delta t\, R(t)} \tag{1.3}$$

Veličina $h(t)$, neboli okamžitá intenzita poruchy, je definována jako limita intenzity poruchy, kdy se tento časový interval blíží k nule, neboli:

$$h(t) = \frac{f(t)}{R(t)} \tag{1.4}$$

Pokud jsou v modelu intenzity událostí (selhání, poruchy, apod.) konstantní, pak je model časově homogenní, v ostatních případech je označován jako nehomogenní.

1.1.3 Statistická rozdělení používaná ve výpočtech spolehlivosti

Publikace Electronic Reliability Design Handbook MIL-HDBK-338B [3] popisuje některá běžně používaná spojitá rozdělení:

- *Exponenciální* – Toto je pravděpodobně nejdůležitější rozdělení v oblasti spolehlivostních výpočtů a používá se téměř výhradně pro predikci spolehlivosti u elektronických zařízení [4]. Popisuje situace, kdy je intenzita poruchy konstantní. Hlavními výhodami jsou:
  - má jediný, jednoduše odhadnutelný parametr λ
  - má poměrně širokou použitelnost
  - je sčitatelné – to značí, že suma více nezávislých exponenciálních rozdělení je opět exponenciální rozdělení
- *Gamma* – Gamma rozdělení se používá pro spolehlivost analýzy případů, kdy se mohou objevit částečné poruchy, například když musí před samotným selháním systému proběhnout několik částečných poruch (většinou se jedná o redundantní systémy) nebo také čas do druhé poruchy, když doba do poruchy je exponenciálně rozdělená.
- *Weibullovo* – Weibullovo rozdělení je obecná varianta rozdělení, které může při vhodném nastavení parametrů modelovat široký rozsah rozdělení v různých typech technických zařízení.
- *Normální (Gaussovo)* – Existují dvě základní použití normálního rozdělení pro spolehlivost. První se zabývá analýzou věcí, které vykazují poruchy kvůli opotřebení, například mechanická zařízení. Druhé použití je analýza výroby zařízení s ohledem na plnění specifikací. Žádná dvě zařízení vyrobená podle stejné specifikace nejsou přesně stejná. Variabilita

součástek vede k variabilitě systému a jeho složení z těchto součástek. Návrh pak musí brát ohled na tuto variabilitu, jinak by výsledné zařízení nemuselo splňovat jeho požadovanou specifikaci díky efektům kombinací těchto součástek. Jiným aspektem použití tohoto rozdělení je kontrola kvality postupů.

- *Logaritmicko-normální* – Logaritmicko-normální rozdělení je takové rozdělení, kde je logaritmus náhodné veličiny roven normálnímu rozdělení – jinými slovy normální rozdělení s $\ln(t)$ jako proměnnou. Toto rozdělení se nejběžněji používá v analýze udržovatelnosti. Aplikuje se na většinu opravovaných a údržbových událostí, zahrnující několik dceřiných událostí s nerovnoměrnou frekvencí a dobou trvání.

Tabulka na obrázku 1.1 [3] nám ukazuje křivky hustot poruch, spolehlivostních funkcí, distribučních funkcí a intenzity poruch pro všechna tato rozdělení.

Více detailů týkajících se spolehlivostních rozdělení (ať už diskrétních nebo spojitých), přidružených spolehlivostních funkcí a spolehlivostních teorií, lze nalézt v [5].

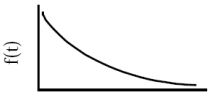
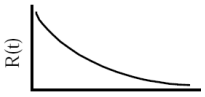
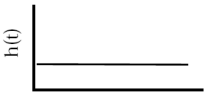
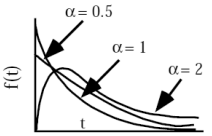
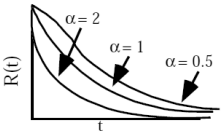
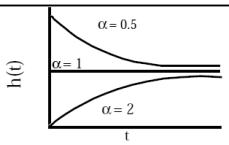
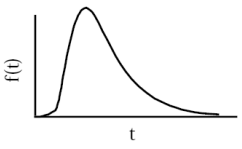
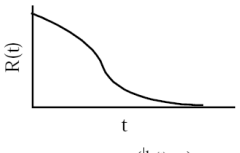
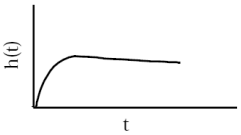
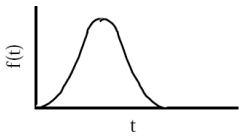
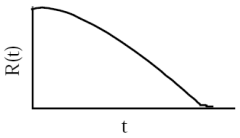
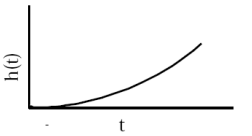
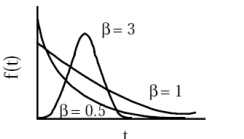
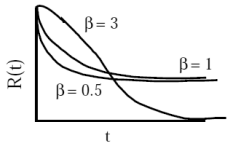
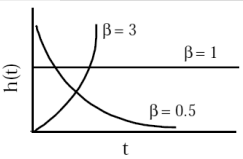
1.2 Markovské řetězce

Pro definici jakéhokoliv Markovskýého řetězce je důležité nejprve definovat všechny pojmy s ním související [6]:

- *Náhodný proces* je každá funkce $X(t)$, jejíž hodnota při každé hodnotě argumentu t je náhodná veličina. Definiční obor argumentu je množina D, tedy $t \in D$. Realizací náhodného procesu $X(t)$ rozumíme funkci $x(t)$ získanou konkrétním pokusem. Náhodný proces $X(t)$ můžeme rovněž chápat jako množinu všech možných realizací $x(t)$.

- *Náhodná posloupnost.* Pokud definiční obor argumentu D obsahuje konečný nebo spočetný počet hodnot, tedy $D = \{t_k\}$, k = 0, 1, 2,..., přechází náhodná hodnota funkce $X(t)$ v náhodnou posloupnost $X(t_k)$. Zkráceně ji zapisujeme $X(t_k) = X_k$.

- *Markovský řetězec* je speciální náhodná posloupnost. Pravděpodobnost, že člen X_k posloupnosti nabude určitou hodnotu, je ovlivněna pouze hodnotou předchozího členu posloupnosti X_{k-1}. Bez ztráty na obecnosti můžeme dále uvažovat diskrétní čas t_k jako posloupnost nezáporných celých čísel, tedy $t_k = k$. V uvažovaných aplikacích představují hodnoty členů X_k čísla stavů. Jejich definičním oborem je například množina přirozených čísel $E = \{1, 2, \ldots, n\}$.

TYPE OF DISTRIBUTION	PROBABILITY DENSITY FUNCTION, f(t)	RELIABILITY FUNCTION $R(t) = \int_t^{\infty} f(t)\,dt = 1 - F(t)$	HAZARD FUNCTION $h(t) = \frac{f(t)}{R(t)}$
EXPONENTIAL	 $f(t) = \lambda e^{-\lambda t}$	 $R(t) = e^{-\lambda t}$	 $h(t) = \lambda = \theta^{-1}$
GAMMA	 $f(t) = \frac{\lambda}{\Gamma(\alpha)} (\lambda t)^{\alpha-1} e^{-\lambda t}$	 $R(t) = \frac{\lambda^{\alpha}}{\Gamma(\alpha)} \int_t^{\infty} \lvert t^{\alpha-1} e^{-\lambda t}\,dt$	 $h(t) = \frac{t^{\alpha-1}e^{-\lambda t}}{\int_t^{\infty} t^{\alpha-1}e^{-\lambda t}dt}$
LOGNORMAL	 $f(t) = \frac{1}{\sigma t (2\pi)} e^{-\frac{1}{2}\left(\frac{\lvert \ln(t) - \mu}{\sigma}\right)^2}$	 $R(t) = 1 - \phi\left(\frac{\lvert \ln(t) - \mu}{\sigma}\right)$ See Note	 $h(t) = \frac{f(t)}{1 - \phi\left(\frac{\lvert \ln(t) - \mu}{\sigma}\right)}$
NORMAL	 $f(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{t-\mu}{\sigma}\right)^2}$	 $R(t) = 1 - \phi\left(\frac{t-\mu}{\sigma}\right)$ See Note	 $h(t) = \frac{f(t)}{1 - \phi\left(\frac{t-\mu}{\sigma}\right)}$
WEIBULL	 $f(t) = \frac{\beta}{\eta}\left(\frac{t-\gamma}{\eta}\right)^{\beta-1} e^{-\left[\left(\frac{t-\gamma}{\eta}\right)^{\beta}\right]}$	 $R(t) = e^{-\left[\frac{(t-\gamma)^{\beta}}{\eta}\right]}$	 $h(t) = \frac{\beta}{\eta}\left(\frac{t-\gamma}{\eta}\right)^{\beta-1}$

Note: $\phi\left(\frac{\ln t - \mu}{\sigma}\right)$ (lognormal) and $\phi\left(\frac{t-\mu}{\sigma}\right)$ (normal) is the standardized form of these distributions and is equal to the integral of the pdfs for those distributions (i.e., the cumulative distribution function).

Obrázek 1.1: Křivky hustot poruch, spolehlivostní funkce a funkce selhání, intenzity poruch pro nejběžněji používané rozdělení.

1.2.1 Homogenní Markovské řetězce

Rozlišujeme dva typy homogenních Markovských řetězců, které se používají:

- *Markovský řetězec s diskrétním časem* lze definovat pomocí tzv. matice přechodu $\mathbf{P}$, kde každý prvek $\mathbf{P}_{ij}$ definuje pravděpodobnost přechodu ze stavu i do stavu j. V případě homogenních Markovských řetězců s diskrétním časem není pravděpodobnost závislá na čase. Pravděpodobnost setrvání v současném stavu (tedy prvek $\mathbf{P}_{ii}$) je dopočítána tak, aby součet všech hodnot v řádku matice byl roven 1.

- *Markovský řetězec se spojitým časem* lze definovat pomocí tzv. matice intenzit $\mathbf{Q}$, kde každý prvek $\mathbf{Q}_{ij}$ definuje intenzitu přechodu ze stavu i do stavu j. V případě homogenních Markovských řetězců se spojitým časem jsou všechny prvky konstantní, pravděpodobnost přechodu je tedy $\mathbf{P}_{ij}(t) = 1 - e^{-\mathbf{Q}_{ij}t}$. Intenzita setrvání v současném stavu (tedy prvek $\mathbf{Q}_{ii}$) je dopočítána tak, aby součet všech hodnot v řádku matice byl roven 0.

1.2.2 Nehomogenní Markovské řetězce

Oba typy homogenních Markovských řetězců lze také definovat s použitím proměnných prvků v matici. Takovéto Markovské řetězce se nazývají nehomogenní.

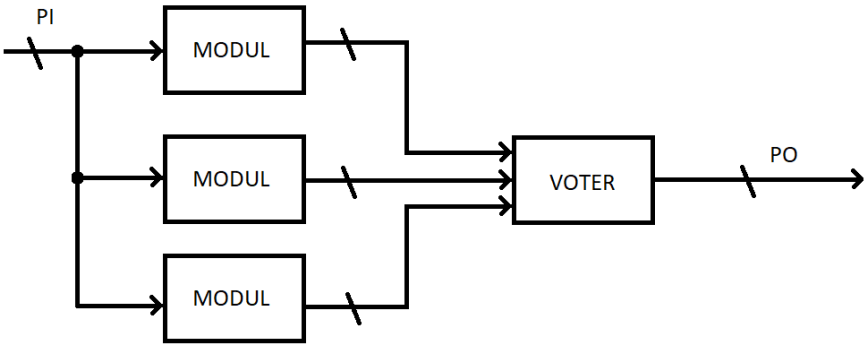
Většina metod pro výpočet pravděpodobností stavů a jejich vývoj v čase nelze pro nehomogenní případy použít. Avšak existují metody, které jsou schopné odhadnout nehomogenní Markovský řetězec pomocí homogenního [7].

Metody sice vedou k nepřesným výsledkům, avšak lze je upravit tak, aby tato chyba byla velmi malá. Toho lze dosáhnout zavedením malých časových intervalů. V práci byla použita metoda, kdy se rozdělil celkový časový interval na velké množství „dílků“, kde každý z nich reprezentuje konstantní matici přechodů, resp. intenzit, tedy v každém dílku se jedná o samostatný homogenní Markovský řetězec.

1.3 Ilustrativní příklad výpočtu Markovského řetězce

Pro jednoduchost si ukážeme všechny matematické výpočty použité v této DP na ukázkovém modelu TMR, který má odpovídající strukturu jako je na obrázku 1.2. Zkratka PI značí primární vstup, zkratka PO značí primární výstup.

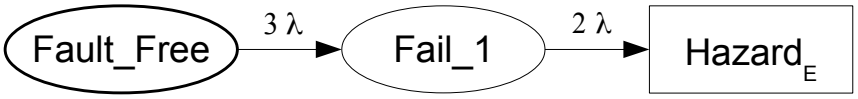
Tento model obsahuje tři identické části, které současně pracují na stejném problému a poté jsou majoritou (voterem) přivedeny na výstup. Pro ukázku v této práci budeme předpokládat intenzitu poruchy jednoho ze tří segmentů



Obrázek 1.2: Blokový model TMR.

λ a intenzitu poruchy voteru zanedbáme, jelikož v reálných podmínkách je tato intenzita řádově nižší než intenzita poruchy segmentů.

Markovský řetězec takového modelu poté vypadá jako na obrázku 1.3.



Obrázek 1.3: Markovský řetezec modelu TMR.

Jednotlivé stavy popisují stav modelu na základě jeho poruch a funkčnosti. Stav *Fault_Free* je stav, při kterém všechny tři části modelu TMR jsou funkční a vykonávají svoji práci. Stav *Fail_1* je stav, při kterém se jedna ze tří částí poškodila a již nemůže vykonávat svoji práci dle zadání. Avšak zařízení má většinu součástí funkčních, takže jako celek je funkční. Stav $Hazard_E$ je stav, ve kterém se poškodila jedna ze zbylých dvou funkčních částí a tudíž již celkový model nemůže vykonávat svoji práci správně, jelikož majorita již neplatí (pouze jedna „funkční“ část ze tří). Takovýto stav nazýváme *absorpční*. Pro větší přesnost si popíšeme, co to vlastně absorpční stav je:

- Stav Markovského řetězce nazýváme *absorpční*, pokud tvoří množinu stavů C, která je uzavřená, to jest pravděpodobnost $\mathbf{P}_{ij}$ přechodu z tohoto stavu do jiného je rovna nule a tudíž z této uzavřené množiny již nikdy „neuteče“ [8].

$$\forall i \in C, \forall j \notin C \; : \; \mathbf{P}_{ij} = 0 \tag{1.5}$$

Matice skokových intenzit $\mathbf{Q}$ řetězce na obrázku 1.3 bude tedy vypadat:

$$\mathbf{Q} = \begin{bmatrix} -3\lambda & 3\lambda & 0 \\ 0 & -2\lambda & 2\lambda \\ 0 & 0 & 0 \end{bmatrix}$$

Diferenciální rovnice tohoto řetězce mají tvar:

$$p'_{Fault\_Free}(t) = -3\lambda p_{Fault\_Free}(t) \tag{1.6}$$
$$p'_{Fail\_1}(t) = 3\lambda p_{Fault\_Free}(t) - 2\lambda p_{Fail\_1}(t) \tag{1.7}$$
$$p'_{Hazard_E}(t) = 2\lambda p_{Fail\_1}(t) \tag{1.8}$$
$$p_{Fault\_Free}(0) = 1, p_{Fail\_1}(0) = 0, p_{Hazard_E}(0) = 0 \tag{1.9}$$

V případě, že je potřeba převést matici skokových intenzit $\mathbf{Q}$ na matici přechodů $\mathbf{D}$, je použit vzorec [8]:

$$\mathbf{D} = \mathbf{I} + \frac{1}{\alpha}\mathbf{Q} \tag{1.10}$$

kde α je vypočítána jako:

$$\alpha := \sup_{i \in S}(-\mathbf{Q}_{ii}) \tag{1.11}$$

za předpokladu, že:

$$\sup_{i \in S}(-\mathbf{Q}_{ii}) \leq +\infty \tag{1.12}$$

a $\mathbf{I}$ značí tzv. jednotkovou matici.

Výsledná matice přechodu $\mathbf{D}$ s parametrem $\alpha = \sup_{i \in S}(-\mathbf{Q}_{ii}) = 3\lambda$ tedy vypadá:

$$\mathbf{D} = \mathbf{I} + \frac{1}{\alpha}\mathbf{Q} = \begin{bmatrix} 1 - \frac{3\lambda}{3\lambda} & \frac{3\lambda}{3\lambda} & 0 \\ 0 & 1 - \frac{\lambda}{3\lambda} & \frac{2\lambda}{3\lambda} \\ 0 & 0 & 1 - 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & \frac{1}{3} & \frac{2}{3} \\ 0 & 0 & 1 \end{bmatrix} \tag{1.13}$$

Z této přechodové matice je vidět, že jediná možnost, do které se můžeme z počátečního stavu dostat, je selhání jednoho ze tří segmentů. Tato situace ale nastane až za určitou dobu t. V případě, že chceme vypočítávat jednotlivé limitně nulové úseky, je potřeba nejprve matici $\mathbf{Q}$ celou vydělit hodnotou limitně se blížící k $+\infty$. Matice pak bude vypadat:

$$\mathbf{Q}(t \to 0) = \begin{bmatrix} \frac{-3\lambda}{x} & \frac{3\lambda}{x} & 0 \\ 0 & \frac{-2\lambda}{x} & \frac{2\lambda}{x} \\ 0 & 0 & 0 \end{bmatrix}, x \to +\infty \tag{1.14}$$

a výsledná přechodová matice bude:

$$\mathbf{P}(t \to 0) = \begin{bmatrix} 1 - \frac{1}{x} & \frac{1}{x} & 0 \\ 0 & 1 - \frac{2}{3x} & \frac{2}{3x} \\ 0 & 0 & 1 \end{bmatrix}, x \to +\infty \tag{1.15}$$

Takováto matice $\mathbf{P}$ se již dá použít pro přechod mezi časovými úseky pomocí maticového násobení, tzv. Chapman-Kolmogorovovy rovnice [9]:

$$\mathbf{P}(2t) = \mathbf{P}(t) * \mathbf{P}(t) \tag{1.16}$$

Podobným způsobem jsme schopni spočítat jakoukoliv matici časového posunu Δt, kterou pro posun v čase můžeme pronásobovat s maticí aktuálního času:

$$\mathbf{P}(t + \Delta t) = \mathbf{P}(t) * \mathbf{P}(\Delta t) \tag{1.17}$$

Díky těmto časovým posuvům lze ovlivňovat v každém násobení i jednotlivé hodnoty intenzit poruch, které mají za následek celkovou nehomogenitu řetězců. To je výhodné vzhledem k cílům této diplomové práce. V rámci následující kapitoly se tyto vzorce objeví v implementovaném algoritmu.

Kapitola 2

Návrh a tvorba výpočetního algoritmu

V rámci celého zadání diplomové práce byl použit software Wolfram Mathematica [10], ve kterém byl již dříve vytvořen původní návrh výpočtů.

Wolfram Mathematica je matematické prostředí, které bylo vyvinuto pro provádění numerických i symbolických výpočtů s nastavitelnou přesností. Síla tohoto prostředí spočívá hlavně v inteligentní automatizaci, propojení symbolické a numerické metodologie nebo využití rychlých pracovních nástrojů založených na spojení zpracování a virtualizace dat v jedné platformě. Této výhody se dá využít při řešení této diplomové práce.

2.1 Příprava výpočetního algoritmu na vstupní data

Jako vstupní data byly použity soubory Markovských řetězců tvořené seznamem přechodů, které jsou definované jako:

$$\{x, y\} \to \alpha \tag{2.1}$$

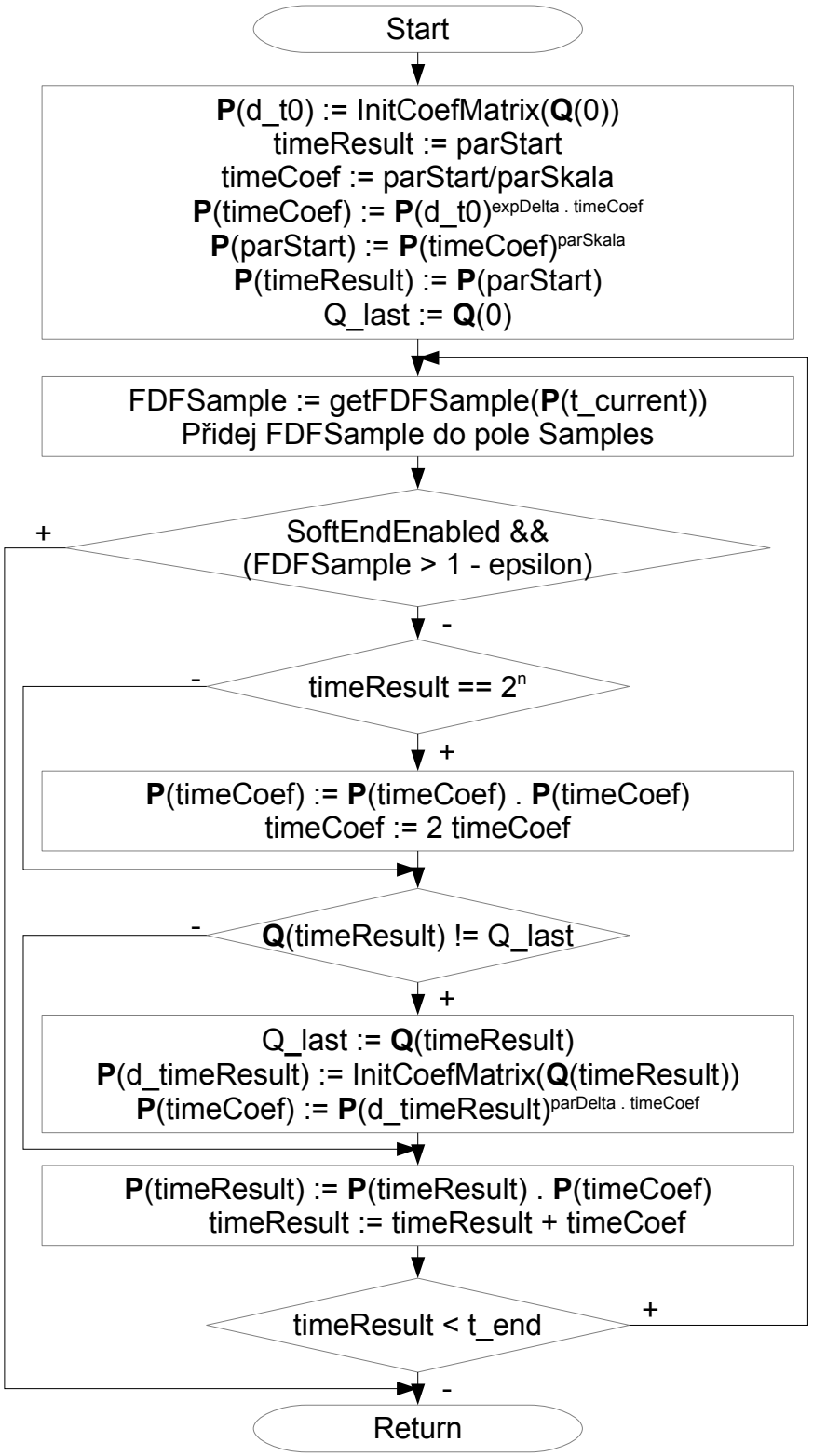
kde:

x : index výchozího stavu, $x \in \mathbb{N}$,

y : index koncového stavu, $y \in \mathbb{N} \wedge y \neq x$,

α : hodnota intenzity přechodu, $\alpha \in \mathbb{R}$,

Na každém řádku seznamu jsou vypsány všechny přechody, které vedou do stejného koncového stavu. V tomto seznamu také nejsou zahrnuty přechody $\{x, x\} \to \alpha$, které se předpokládají automaticky.



Obrázek 2.1: Postup výpočtu distribuční funkce pro nehomogenní model.

2.2 Výpočetní algoritmus

Na obrázku 2.1 je vidět postup algoritmu pro výpočet distribuční funkce. Tento algoritmus je převzat a upraven z článku, resp. posteru z konference DTIS [11].

V této části je rozebrán podrobně celý algoritmus včetně všech použitých procedur.

2.2.1 Načtení matice a získání absorpčních stavů

Na základě pravidel popsaných výše byla vytvořena funkce `LoadMatrix` pro získání zápisu regulární matice všech stavů Markovského modelu. Tento seznam je dále udržován pro potřeby dalších výpočtů.

Dále byla vytvořena funkce `FindAbsorbStates`, která v této regulární matici nalezne všechny absorpční stavy. Pro přehlednost se v tuto chvíli absorpční stavy naleznou tak, že jejich celý řádek v matici obsahuje nuly – nevede z nich žádný přechod.

2.2.2 Funkce `InitCoefMatrix`

Funkce `InitCoefMatrix` zpracovává načtenou matici skokových intenzit $\mathbf{Q}(0)$, převádí jí na pravděpodobnostní matici pro násobení $\mathbf{P}(d_{t0})$ a provádí výpočty časů pomocí parametrů `parSkala_`, `parDelta_` a `destmista_`. Tyto parametry jsou definovány takto:

- `parSkala_`: parametr škálovatelnosti, který stanovuje kolik bodů se bude vypočítávat a potom případně i vykreslovat v jednotlivých časových úsecích $2^x - 2^{x+1}$, $x \in \mathbb{N}_0$ (jedná se o mocniny 2, tedy $2^3 = 8$ bodů v jednom časovém úseku),
- `parDelta_`: nastavuje hloubku podílového koeficientu x z rovnic 1.14 a 1.15,
- `destmista_`: určuje počet desetinných míst, se kterými se v matici bude pracovat a na kolik míst nám program zajistí přesnost.

Další parametry této funkce jsou již ovlivňovány za běhu:

- `matrixDef_`: nejdůležitější parametr této funkce je samozřejmě naše matice načtená pomocí dříve zmíněné funkce `LoadMatrix`, do které jsou přidána pravidla – seznam všech parametrů v matici a jejich hodnot (mohou být závislé na čase),
- `parActual_`: určuje jednotku času, kde se ve výpočtu právě nacházíme (v prvním zavolání této funkce v algoritmu je nastavena na parametr `parStart_`),

Jelikož je potřeba při násobení matic a posunu v čase používat matice pravděpodobnosti, byla tedy matice skokových intenzit **Q** převedena na matici přechodů **P**. Tento převod probíhá na základě vzorců 1.13, 1.14 a 1.15.

Takto vytvořená matice je potom pomocí násobení regulárních matic převedena na dvě matice:

1. Matice časového rozdílu **P**(`timeCoef_`), která je násobením dopočítána do hodnoty $2^{\texttt{parDelta\_}-\texttt{parSkala\_}}$, kde oba parametry jsou popsány výše.

2. Matice původní časové jednotky **P**(`parStart_`), která je násobením dopočítána do hodnoty $2^{\texttt{parDelta\_}}$, jejíž parametr je popsán již dříve.

Obě výše popsané matice se používají také v dalších částech algoritmu.

2.2.3 Hlavní část algoritmu

V této části je popsáno, jak pracuje hlavní část algoritmu.

I. Algoritmus si načte obě matice vytvořené při inicializaci – tedy matici časového rozdílu **P**(`timeCoef_`) a matici původní časové jednotky **P**(`parStart_`) z funkce `InitCoefMatrix`.

II. Proběhne prvotní nastavení všech časových koeficientů, ve kterých se pohybujeme – parametr startu, časových úseků měření (tedy podíl hodnoty $2^{\texttt{parStart\_}}$ a $2^{\texttt{parSkala\_}}$, kde `parStart_` značí parametr začátku měření a `parSkala_` byl popsán výše), parametru `parFinish_`, který nám udává hodnotu časového úseku $2^{\texttt{parFinish\_}}$, kdy je algoritmus ukončen, parametru `timeResult_`, který je na začátku nastaven na hodnotu $2^{\texttt{parStart\_}}$ a parametru `timeCoef`, který je před spuštěním algoritmu nastaven na hodnotu $2^{\texttt{parStart\_}-\texttt{parSkala\_}}$.

III. V případě nastavení parametru `toEnd_` a `epsilon_` se algoritmus v každém kroku podívá, zda-li vypočítaná hodnota distribuční funkce $F(t)$ již přesáhla hodnotu $1 - \texttt{epsilon\_}$ a v případě úspěchu tento algoritmus výpočtu ukončí.

Další část algoritmu již probíhá ve smyčce, která je ukončena v případě výše zmíněného dosažení hodnoty $1 - \texttt{epsilon\_}$ nebo v případě dosažení výše zmíněné hodnoty $2^{\texttt{parFinish\_}}$.

a) Na začátku smyčky se vypočítá pravděpodobnost uvíznutí v absorpčním stavu/absorpčních stavech za předpokladu, že počáteční stav byl v matici přechodu stav na prvním řádku (pravděpodobnost je spočtena jako suma pravděpodobností přechodu v dané časové hodnotě z počátečního stavu do absorpčního, tedy takové indexy na řádku, které jsou uloženy v proměnné po volání funkce `FindAbsorbStates`).

b) Do pole výsledků *Samples* je uložena dvojice hodnot: časová jednotka, která se vyskytuje ve výše zmíněné proměnné `timeResult_`, a v předchozím kroku vypočítané hodnoty pravděpodobnosti uvíznutí v absorpčním stavu.

c) V případě, že se dokončil jeden časový úsek (tedy úsek 2^x – 2^{x+1}, $x \in \mathbb{N}_0$), algoritmus zdvojnásobí časový koeficient ve výše zmíněné proměnné `timeCoef_`, zvedne se hodnota současného exponentu a koeficientová matice $\mathbf{P}$(`timeCoef_`) se pronásobí sama se sebou, abychom v algoritmu dosáhli zvětšování časových jednotek.

d) V případě pokud se změnila jakákoliv intenzita v závislosti na čase definována v pravidlech, musí proběhnout nová inicializace koeficientové matice $\mathbf{P}$(`timeCoef_`) s novými parametry (funkce `InitCoefMatrix`).

e) Výsledná matice se pronásobí s koeficientovou maticí $\mathbf{P}$(`timeCoef_`), aby byl zajištěn posun v čase, dále se k výslednému času přičte napočítaná hodnota `timeCoef_` a proběhne tak posun v aktuálním čase.

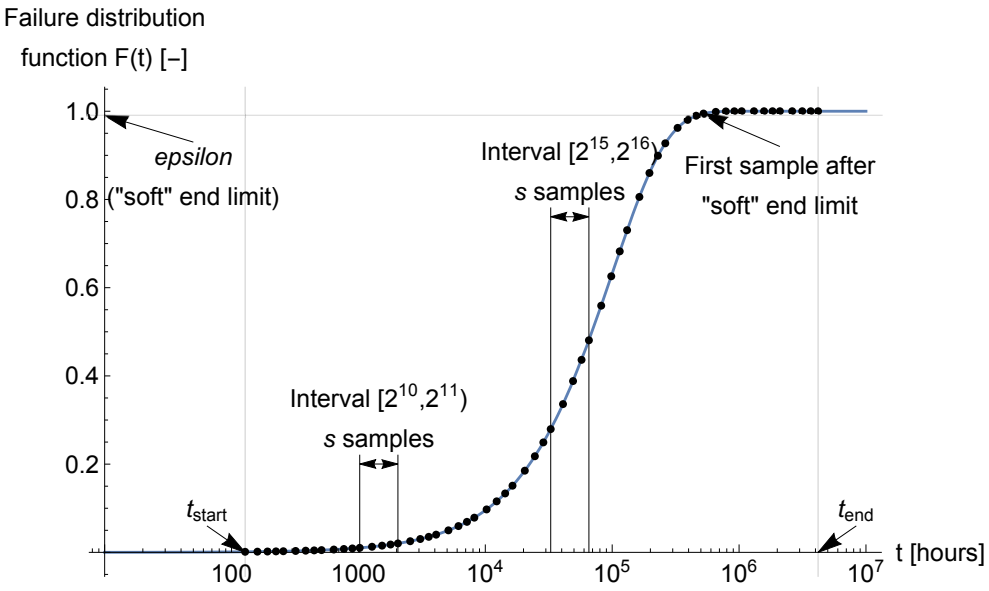
Pro účely testování je porovnávána jak doba výpočtu inicializační matice, tak výpočtu celé distribuční funkce $F(t)$, výsledné časy jsou poté ukládány do samostatného souboru, aby k nim byl v dalších fázích přístup. V případě, že se některé části algoritmu provedly velmi rychle, byly tyto části pomocí vestavěné funkce Mathematicy `RepeatedTiming` [10] spuštěny několikrát po sobě, výsledné časy se poté zprůměrovaly.

Výsledný seznam dvojic časových hodnot a hodnot pravděpodobností uvíznutí je dále graficky znázorněn pro vizuální kontrolu, zda-li algoritmus provedl vše přibližně podle požadavků a tento seznam je uložen v samostatném souboru pro účely dalších prací, které jsou ukázány v kapitole Testování.

2.3 Shrnutí významu parametrů algoritmu

Celý tento algoritmus má za úkol pokusit se výpočet distribuční funkce $F(t)$ rozdělit na velmi malé intervaly (limitně nulové intervaly v ideálním případě – v praxi tento ideální případ nelze praktikovat), díky čemuž se může v jednotlivých krocích kontrolovat, zda-li se v závislosti na čase nezměnily intenzity jednotlivých poruch. Na základě intenzit lze přepočítávat matici přechodů a výpočet pravděpodobnosti uvíznutí v absorpčním stavu, jak je vidět na obrázku 2.2. Horizontální osa označuje čas t v hodinách, vertikální osa pak hodnotu distribuční funkce $F(t)$, tedy pravděpodobnost uvíznutí v absorpčním stavu.

Na obrázku 2.2 jsou použity ilustrativní hodnoty těchto parametrů (obrázek byl převzat z článku [12]):



Obrázek 2.2: Ukázka jednotlivých parametrů, které ovlivňují výpočet distribuční funkce $F(t)$.

- *epsilon* – parametr určení limitu výpočtu (v algoritmu se nachází jako parametr `epsilon_`, hodnota vybrána v rozmezí 10^{-10} až 10^{-3}, tedy taková hodnota, která již z pohledu spolehlivosti značí téměř jisté selhání),
- t_{start} – parametr počátku výpočtu (v algoritmu parametr `parStart_`, v ukázce je použita hodnota $2^7 = 128$),
- t_{end} – parametr koncové hranice výpočtu (v algoritmu `parFinish_`, v ukázce je použita hodnota $2^{22} \doteq 4 \times 10^6$),
- s – parametr počtu vzorků v časové jednotce (v algoritmu parametr `parSkala_`, v ukázce je použita hodnota 4).

Kapitola 3

Testování

Vypracovávání algoritmu a jeho následné testování probíhalo během celého navazujícího magisterského studia. Veškeré výsledky byly zaznamenávány a použity pro články na konference DTIS 2019 [11] a DSD 2019 [12].

Je důležité podotknout, že časová náročnost všech testování je ovlivněna hlavně výkonem samotného zařízení, na kterém dané výpočty probíhaly. V našem případě byl pro testování použit notebook s těmito parametry:

- Procesor: Intel Core i5-7300HQ @2.5 GHz
- Operační systém: Windows 10 64-bit
- Program: Wolfram Mathematica 11.2

3.1 Definice testovacích modelů

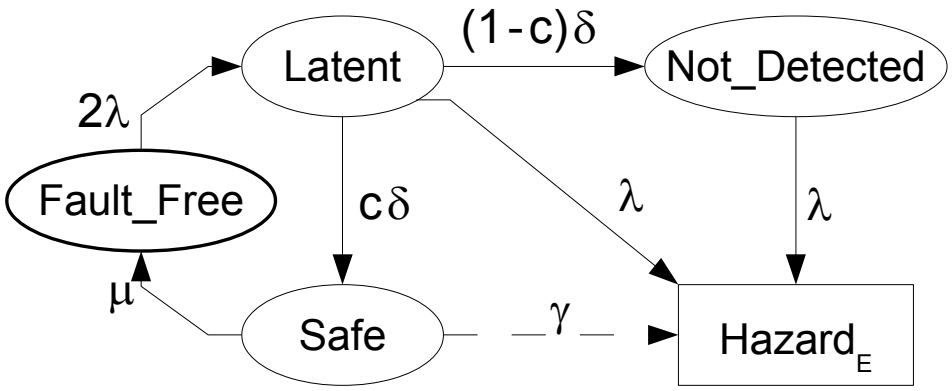
Pro účely testování byly použity dva Markovské řetězce, které jsou převzaty z dizertační práce pana Ing. Martina Kohlíka, Ph.D. [13].

3.1.1 Blok 2oo2

Model vyobrazený na obrázku 3.1 je použit pro výpočet distribuční funkce $F(t)$ 2oo2 bloku.

$Fault\_Free$ je stav, kdy je daný blok funkční a bez poruchy. Intenzita první poruchy je 2λ, protože první porucha může nastat v jakémkoliv ze dvou funkčních modulů bloku.

Stav *Latent* je aktivní, když tento blok obsahuje poruchu, která ještě nebyla detekována. Intenzita tzv. on-line testu (převrácená hodnota průměrného zpoždění mezi poruchou a její detekcí) je značena jako δ. Pokud je daná porucha detekována, blok se uzamkne a přejde do stavu $Safe$. Pravděpodobnost úspěšné detekce poruchy v testu je definována jako c.



Obrázek 3.1: Markovský řetězec modelu dva-ze-dvou (2oo2).

Pokud tento on-line test selže (porucha nebyla detekována), blok se bude nacházet ve stavu *Not_Detected*. Bezpečnost bloku v tuto chvíli ještě není narušena, ale případná další porucha (s intenzitou λ) má za následek selhání bloku a ten se pak přesune do nebezpečného, tzv. $Hazard_E$ stavu. Další chyba uvnitř stejného modulu, který již obsahuje poruchu, nemá na poruchu žádný vliv, jelikož druhý modul je stále funkční.

Přechod vedoucí mezi stavy *Latent* a $Hazard_E$ vyjadřuje pravděpodobnost, že porucha na zbylém funkčním modulu nastane dříve, než bude dokončen on-line test.

Blok je uzamčen ve stavu $Safe$ do té doby, dokud není dokončena jeho oprava (intenzita opravy μ). Blok v této situaci není funkční, avšak bezpečnost není ohrožena.

V případě, že blok se nachází ve stavu $Safe$, bude funkčnost bloku udržována záložní/nouzovou metodou (např. manuální obsluhou). Intenzita γ udává pravděpodobnost chyby manuální obsluhy.

Pravděpodobnost detekce poruchy, intenzita poruchy a intenzita on-line testu bloku jsou definovány následujícími parametry, které byly vybrány z [14].

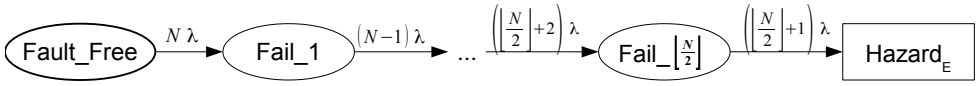
$\mu = 24^{-1}\,[h^{-1}]$ – intenzita opravy

$\lambda = 10^{-5}\,[h^{-1}]$ – intenzita poruchy

$\delta = 10^{-1}\,[h^{-1}]$ – intenzita on-line testu

$c = 0.6$ – pravděpodobnost odhalení chyby během testu

$\gamma = 10^{-3}\,[h^{-1}]$ – intenzita chyby manuální obsluhy



Obrázek 3.2: Markovský řetězec generického N-modulárního redundantního systému.

3.1.2 Blok *N*-MR

Tento blok je tvořen N stejnými moduly. Na začátku blok začíná ve stavu $Fault\_Free$, kdy jsou všechny moduly funkční a není zde žádná porucha.

Jednotlivé stavy $Fail\_1$ až $Fail\_\lceil \frac{N}{2} \rceil$ popisují stavy, ve kterých daný počet modulů již selhal, avšak celý blok je stále funkční, protože je stále funkční více jak polovina modulů.

V případě, že se porouchá nadpoloviční část modulů, je tento blok považován za nefunkční a přechází do stavu $Hazard_E$.

Intenzity přechodů mezi jednotlivými stavy jsou definovány jako součin intenzity jednoho modulu λ a celkového počtu funkčních modulů.

Model vyobrazený na obrázku 3.2 je použit pro výpočet distributivní funkce $F(t)$ pro generický N-MR blok. V této diplomové práci byly použity bloky s $N = 3$ až 9 moduly.

3.2 Testování modelů v závislosti na parametrech

Během testování výpočtů spolehlivosti pomocí výše popsaného algoritmu násobení pravděpodobnostních matic se testovala závislost časové náročnosti a přesnosti výpočtů na jednotlivých parametrech. Z těchto testů se vybraly takové hodnoty parametrů, které odpovídají kompromisu mezi časovou náročností a přesností.

3.2.1 Parametr `parDelta_`

Parametr `parDelta_` byl definován jako hloubka počátečního dělení matice pro převod inicializační matice přechodu na limitně nulovou časovou matici. Výsledná data, včetně použité výchozí hodnoty pro testování, jsou vidět v tabulce 3.1.

Ostatní parametry pro toto testování byly zvoleny takto:

- `parSkala_`: 4, `destmista_`: 100

V prvním sloupci jsou zobrazeny hodnoty parametru `parDelta_`, se kterými se algoritmus testoval. V dalších dvou sloupcích jsou vypsány hodnoty časů výpočtu první inicializace matice (funkce `InitCoefMatrix`), resp. délka

výpočtu hlavní části algoritmu. Poslední dva sloupce označují relativní odchylky prvního vzorku a pak celkových hodnot výpočtu. Průměrné hodnoty byly vypočítány z absolutních hodnot odchylek.

`parDelta_` [-]	**Čas inic. části [s]**	**Čas hlavní části [s]**	**Rel. odchylka 1. vzorku [-]**	**Průměrná rel. ochylka [-]**
2^{10}	0.047	0.959	-1.11×10^{-2}	8.17×10^{-4}
2^{15}	0.076	0.976	-3.48×10^{-4}	2.56×10^{-5}
2^{20}	0.100	0.963	-1.09×10^{-5}	8.00×10^{-7}
2^{25} [1]	**0.131**	**0.976**	$\mathbf{-3.40 \times 10^{-7}}$	$\mathbf{2.50 \times 10^{-8}}$
2^{30}	0.160	0.962	-1.06×10^{-8}	7.81×10^{-10}
2^{35}	0.187	0.950	-3.32×10^{-10}	2.44×10^{-11}
2^{40}	0.220	0.979	-1.04×10^{-11}	7.63×10^{-13}
2^{45}	0.240	0.960	-3.25×10^{-13}	2.38×10^{-14}
2^{50}	0.268	0.966	-1.01×10^{-14}	7.45×10^{-16}
2^{55}	0.290	0.980	-3.17×10^{-16}	2.33×10^{-17}
2^{60}	0.310	0.945	-9.60×10^{-18}	7.05×10^{-19}
2^{65} [2]	0.340	0.954	—	—

[1] Výsledná hodnota, která byla pro další testy použita jako výchozí.
[2] Hodnota, která byla považována za „nejpřesnější“ a odchylky se porovnávaly s těmito naměřenými daty.

Tabulka 3.1: Porovnání CPU-časů a relativních odchylek metody na základě úpravy parametru `parDelta_`.

3.2.2 Parametr `destmista_`

Parametr `destmista_` byl definován jako počet desetinných míst, na které se zaokrouhluje výpočet inicializační matice po dělení pomocí předem stanoveného parametru `parDelta_`.

Ostatní parametry pro toto testování byly zvoleny takto:

- `parDelta_`: 25, `parSkala_`: 4

Výsledná data vidíte v tabulce 3.2. Vzhledem k minimálním časovým rozdílům ve výpočtu byla jako základní hodnota použita nejvyšší hodnota, která dává zároveň největší přesnost. Výsledky jednotlivých testů jsou stejné, jediné rozdíly jsou v počtu platných desetinných míst ve výsledku.

`destmista_` [-]	**Čas inic. části [s]**	**Čas hlavní části [s]**
15	0.120	0.900
20	0.120	0.911
25	0.120	0.940
30	0.120	0.946
35	0.121	0.936
40	0.121	0.924
	...	
60	0.120	0.933
70	0.124	0.970
	...	
100 [1)]	0.131	0.976

[1)] Výsledná hodnota, která byla použita jako výchozí.

Tabulka 3.2: Porovnání CPU-časů a relativních odchylek metody na základě úpravy parametru `destmista_`.

3.2.3 Parametr `parSkala_`

Parametr `parSkala_` byl definován jako počet vzorků mezi dvěma časovými jednotkami (tedy mezi hodnotami 2^x a 2^{x+1}, kde $x \geq 0$). Tento parametr koriguje výsledný počet dat při testování, aby křivka mezi parametry `parStart_` a `parFinish_` byla vykreslena co nejpřesněji v závislosti na době výpočtu.

Ostatní parametry pro toto testování byly zvoleny takto:

- `parDelta_`: 25, `destmista_`: 100

U tohoto parametru se však nemění přesnost, ale pouze časová náročnost výpočtů. Výsledná data, včetně použité výchozí hodnoty pro testování, jsou vidět v tabulce 3.3.

V prvním sloupci je zobrazen počet vzorků na jeden časový úsek (jako výchozí hodnota při dalších testech byla zvolena hodnota $2^4 = 16$ vzorků). Zbylé dva sloupce vyznačují časovou náročnost výpočtu inicializační části, resp. hlavní části algoritmu.

`parSkala_` [-]	**Čas inic. části** [s]	**Čas hlavní části** [s]
2^1	0.130	0.210
2^2	0.130	0.330
2^3	0.130	0.560
2^4 [1)]	**0.130**	**0.962**
2^5	0.125	1.840
	...	
2^{10}	0.126	57.40

[1)] Výsledná hodnota, která poté byla použita jako výchozí.

Tabulka 3.3: Porovnání CPU-časů metody na základě parametru `parSkala_`.

3.3 Testování modelu v závislosti na nehomogenitě intenzit poruch

Všechny předchozí výpočty byly založené na variantě homogenního Markovského řetězce (všechny intenzity byly konstantní v čase). Tato sekce popisuje časovou náročnost a relativní odchylku mezi homogenním a nehomogenním řešením.

Při testování je použito stejné nastavení, které je popsáno výše, ale namísto abychom testovali, zda intenzita $\mathbf{Q}(t_{current})$ a $\mathbf{Q}_{last}$ je stejná, tak tuto nerovnost předpokládáme vždy, když ji algoritmus kontroluje. Tato modifikace simuluje, že každý vzorek (postup mezi dvěma časovými body) je ovlivněn jakoukoliv změnou intenzity jakékoliv poruchy. To má za následek výpočet inicializační matice $\mathbf{P}(dt_{current})$ a matice časového posunu $\mathbf{P}(\Delta t)$ v každém kroku algoritmu. Hodnoty výpočtů by měly být totožné s homogenním modelem, avšak časová složitost bude podobná jako v případě nehomogenního modelu.

Tabulka 3.4 ukazuje vliv této modifikace. První sloupec značí danou metodu, druhý sloupec dobu výpočtu první inicializační matice a třetí sloupec nám vyobrazuje rozdíl časové náročnosti samotného výpočetního algoritmu. Průměrná relativní odchylka všech vzorků (porovnaných mezi homogenní a nehomogenní variantou) je zobrazena v posledním sloupci.

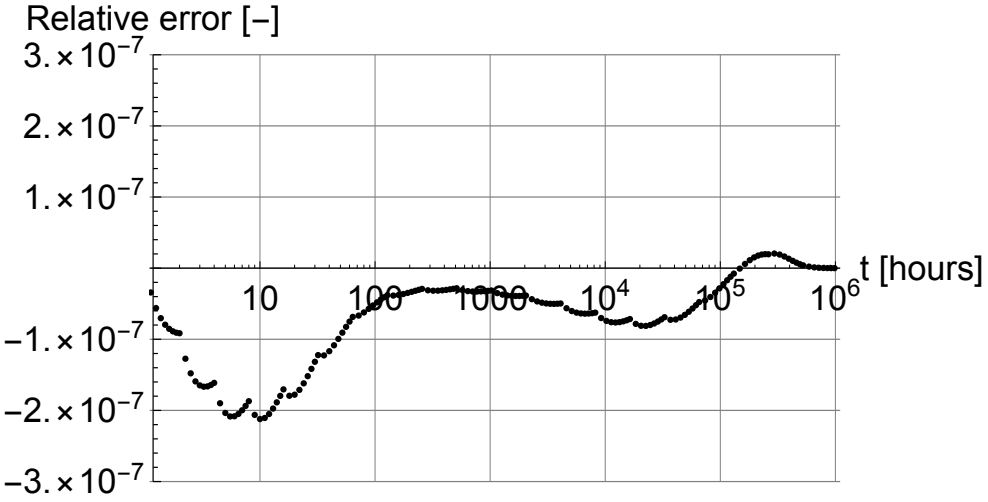
Na grafu, který je vyobrazen na obrázku 3.3, je vidět relativní odchylka všech vzorků mezi homogenní a nehomogenní metodou. Horizontální osa reprezentuje časovou osu v hodinách, vertikální osa pak představuje velikost relativní odchylky.

Jak je vidět z grafu, maximální relativní odchylka nehomogenní metody je

přibližně 10^{-7}. Inicializační doba výpočtu je téměř stejná v obou případech, ale doba výpočtu hlavní částí algoritmu se zvětšila téměř 20x z důvodu opakovaného výpočtu nové matice $\mathbf{P}(dt_{current})$ a matice $\mathbf{P}(\Delta t)$ v každém kroku.

Typ metody	**Čas inic. části** [s]	**Čas hlavní části** [s]	**Průměrná rel. ochylka** [-]
Homogenní	0.130	1.030	—
Nehomogenní	0.141	17.83	7.20×10^{-8}

Tabulka 3.4: Porovnání CPU-časů homogenní a nehomogenní metody.

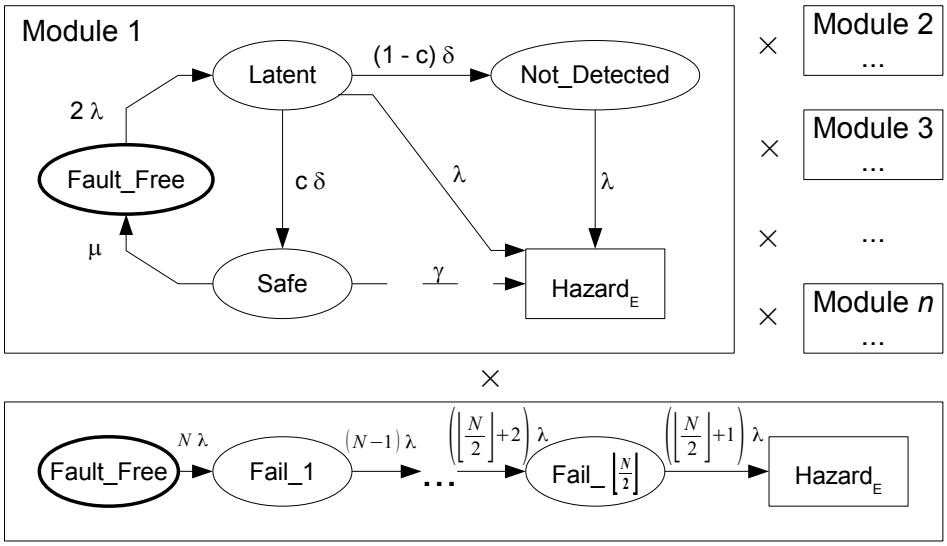


Obrázek 3.3: Relativní odchylka všech vzorků mezi homogenní a nehomogenní metodou.

3.4 Porovnání standardního modelu s hierarchickým modelem

V rámci testování probíhala aplikace výpočetního algoritmu jak na standardní Markovský řetězec vytvořený kartézským součinem bloku N-MR a bloku 2oo2, tak také na hierarchickém spolehlivostním modelu těchto dvou bloků.

Exaktní model, který je vyobrazen na obrázku 3.4, má příliš mnoho stavů, takže i výsledné násobení matic probíhá na velkém řetězci, což má za následek výrazné zpomalení výpočtu. Z tohoto důvodu byl spuštěn výpočet i na tzv. hierarchický model.



Obrázek 3.4: Vyobrazení postupu při výpočtu exaktního modelu.

Hierarchický model byl otestován tak, že se nejprve výpočetní algoritmus spustil na blok 2oo2, jehož vzorky vytvořily distribuční funkci $F(t)$. Z funkce $F(t)$ byla vypočtena intenzita selhání 2oo2 bloku a ta byla dosazena jako intenzita poruchy λ do bloku N-MR. Tato varianta byla možná pouze za předpokladu, že je použita nehomogenní varianta algoritmu.

Následně byly výsledky hierarchického modelu porovnány s výsledky exaktního modelu, který vznikl již zmíněným kartézským součinem bloků.

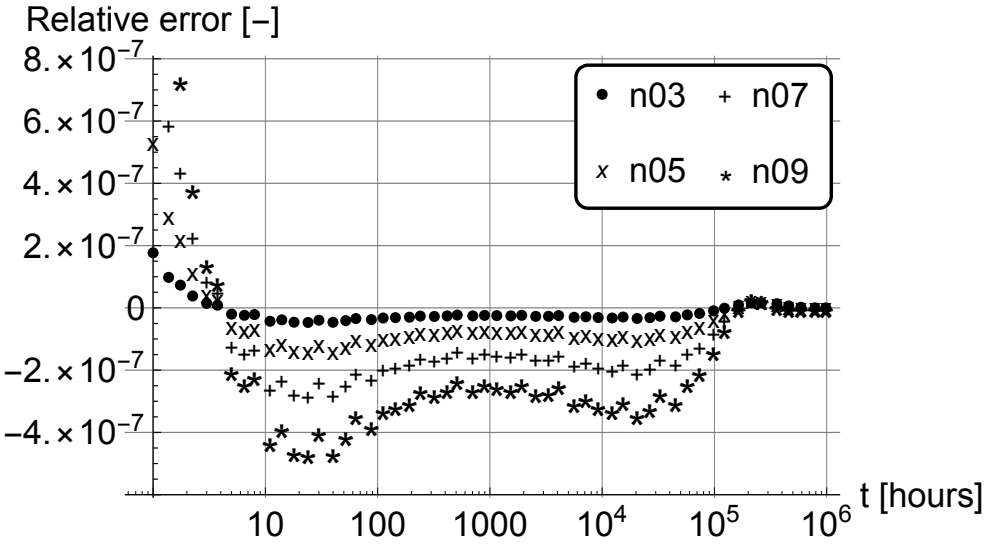
Tabulka 3.5 nám ukazuje porovnání CPU-časů a relativních odchylek mezi hierarchickým a exaktním řešením. V prvním sloupci je definován počet N, tedy 2oo2 bloků pro model N-MR, další dva sloupce pak zobrazují CPU-čas výpočtů exaktního řešení, resp. hierarchického řešení. Tyto hodnoty jsou dány součtem výpočtu inicializační části programu a pak samotné hlavní části programu. Předposlední sloupec vyznačuje porovnání prvního vypočítaného vzorku, tedy odchylku vzorku v čase $t = 1$h hierarchického modelu proti exaktnímu řešení. Průměrná relativní odchylka všech vzorků hierarchického modelu proti exaktnímu je vidět v posledním sloupci.

Graf na obrázku 3.5 nám zobrazuje relativní odchylky jednotlivých vzorků mezi exaktním a hierarchickým modelem. Horizontální osa reprezentuje časovou jednotku měřenou v hodinách, vertikální osa znázorňuje velikost relativní odchylky. V tomto grafu je však zobrazen pouze každý třetí vzorek, jelikož naměřených hodnot je opravdu hodně a způsobily by nepřehlednost výsledného grafu.

CPU-čas, který stráví algoritmus výpočtem exaktního modelu roste velmi strmě v závislosti na počtu 2oo2 bloků, jelikož vzniká kartézským součinem

***N*-MR bloků**	**Čas exak. řešení [s]**	**Čas hier. řešení [s]**	**Rel. odchylka 1. vzorku [-]**	**Průměrná rel. ochylka [-]**
n03	1.160	0.091	1.77×10^{-7}	2.94×10^{-8}
n05	37.23	0.130	5.31×10^{-7}	8.67×10^{-8}
n07	627.5	0.169	1.06×10^{-6}	1.73×10^{-7}
n09	5,938	0.225	1.77×10^{-8}	2.85×10^{-7}

Tabulka 3.5: Porovnání CPU-časů a relativních odchylek mezi exaktním a hierarchickým řešením.



Obrázek 3.5: Relativní odchylky vzorků mezi exaktním a hierarchickým řešením.

i více stavů Markovského řetězce. Naproti tomu výpočet hierarchického modelu pro všechny naměřené počty bloků nepřesáhl 1 vteřinu.

Maximální hodnoty relativní odchylky se s narůstajícím počtem bloků v N-MR mírně zvyšují, avšak tyto hodnoty se v celé oblasti pohybují velmi nízko, řádově v hodnotách maximálně do 10^{-7}.

3.5 Testování metody maticového násobení a porovnání s analytickým řešením bloku *N*-MR

U vybraných modelů lze použít analytické řešení, tedy řešení pomocí vzorce. Jeden z těchto příkladů je výše zmíněný blok N-MR. V této části se porovná výše zmíněná metoda násobení matic s analytickým řešením a tedy přesnými výsledky.

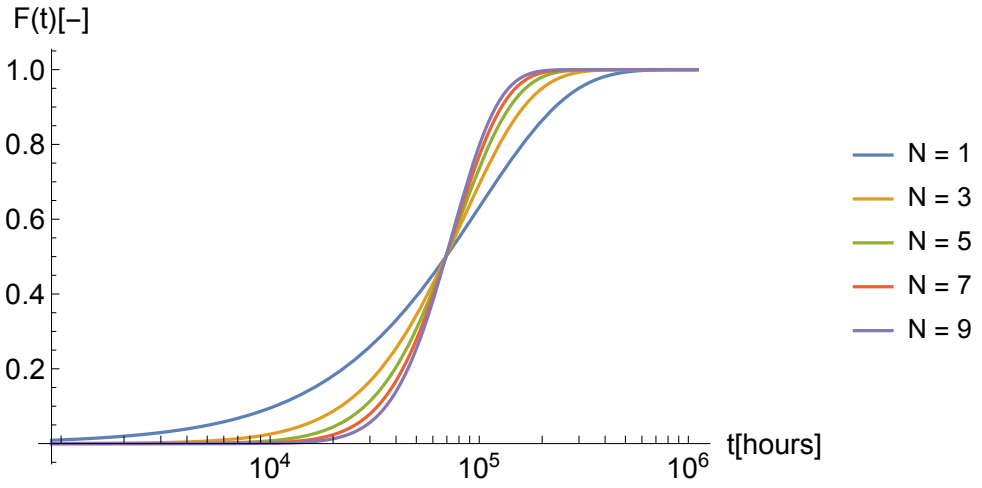
V této situaci nám velice pomůže systém N-MR, se kterým se během celé diplomové práce pracovalo. Tento model má vzorec pro výpočet distribuční funkce $F(t)$:

$$F_{N\text{-MR}}(t) = 1 - \sum_{i=M}^{N} \binom{N}{i} [F(t)]^{N-i} [1 - F(t)]^{i} \tag{3.1}$$

kde:

$$F(t) = 1 - e^{-\lambda t} \tag{3.2}$$

Z rovnice 3.1 vyplývají grafy jednotlivých distribučních funkcí vykreslené na obrázku 3.6. Jako parametr λ jsme zvolili konstantní hodnotu ($\lambda = 10^{-5}$). Horizontální osa znázorňuje čas t v hodinách. Vertikální osa značí hodnotu distribuční funkce $F(t)$.



Obrázek 3.6: Distribuční funkce N-MR.

Tato distribuční funkce se dá použít jako vhodný příklad pro porovnání s výše popsanou metodou násobení matic, jelikož se jedná o exaktní výsledek.

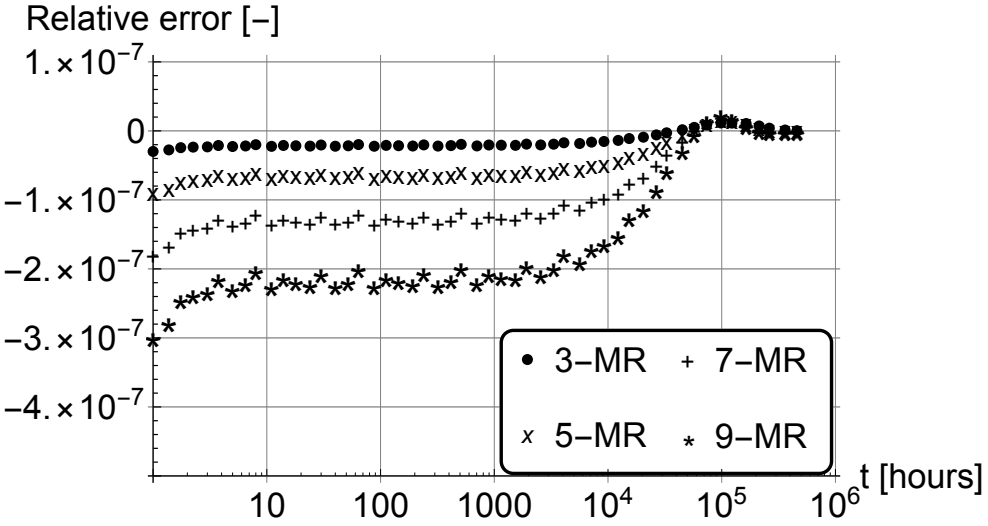
Tabulka 3.6 nám ukazuje porovnání relativních odchylek mezi metodou násobení matic a přesným výpočtem daným rovnicí. V prvním sloupci máme

definován počet bloků v rámci N-MR, zbylé dva sloupce značí hodnotu relativní odchylky prvního vzorku, respektive průměrné relativní odchylky všech vzorků modelu vypočítaného pomocí násobení matic oproti výpočtu rovnice.

***N*-MR bloků**	**Rel. odchylka prvního vzorku [-]**	**Průměrná rel. ochylka [-]**
n03	-2.98×10^{-8}	1.69×10^{-8}
n05	-8.94×10^{-8}	4.92×10^{-8}
n07	-1.79×10^{-7}	9.76×10^{-8}
n09	-2.98×10^{-7}	1.62×10^{-7}

Tabulka 3.6: Porovnání relativních odchylek mezi výpočtem pomocí násobení matic a výpočtem pomocí rovnice.

Graf, který je na obrázku 3.7, nám zobrazuje relativní odchylky jednotlivých vzorků mezi výpočtem pomocí maticového násobení a analytickým řešením pomocí rovnice. Horizontální osa reprezentuje časovou osu, která byla napočítána v hodinách, vertikální osa značí velikost relativní odchylky jednotlivých vzorků. V grafu je znovu vyznačen pouze každý třetí vzorek, jelikož naměřených dat je příliš a došlo by k znepřehlednění grafu.



Obrázek 3.7: Spolehlivost N-MR.

Maximální hodnota odchylky se s přibývajícím počtem bloků mírně zvyšuje, avšak zůstává stále velmi nízká (maximálně okolo 10^{-7}).

Kapitola 4

Hodnocení

Tato kapitola je zaměřena na celkové hodnocení předcházející kapitoly a budoucí přínos pro studium i vědeckou tvorbu.

Veškerá testování probíhala v několika opakováních, jelikož byla snaha veškeré algoritmy a výpočty co nejlépe implementovat a dosáhnout tak co nejmenší výpočetní složitosti. Dále pak samozřejmě docházelo k nepřesnostem při implementaci, které ovšem byly viditelné až na základě vypočtených dat. Z tohoto důvodu zabralo testování největší práci. Dále byl tento čas testování podmíněn i zhodnocením výsledků, které se potom projevily jak v této diplomové práci, tak i ve článcích na konferencích DTIS 2019 a DSD 2019 [11, 12].

4.1 Vyhodnocení parametrů použitých v algoritmu

Během testování jednotlivých parametrů (parametry `parDelta_`,`destmista_` a `parSkala_`) probíhaly další úpravy algoritmu, jelikož bylo potřeba daný výpočet usměrňovat. Například pro zrychlení výpočtů byly všechny tyto parametry napočítány jako mocniny dvou, protože program Wolfram Mathematica s těmito mocninami umí lépe pracovat. Například maticové násobení jiných hodnot než mocnin dvou by mělo za následek opakované násobení stejnou maticí pro získání postupu z matice $\mathbf{P}^2$ na matici $\mathbf{P}^3$. Docházelo by tedy k mnohem většímu počtu těchto operací a tedy i k zpomalení celého výpočtu.

4.1.1 Vyhodnocení parametru `parDelta_`

Při testování parametru `parDelta_` je vidět, že změna parametru má největší dopad na časovou náročnost právě při výpočtu matice časového posunu a v hlavním těle algoritmu již takový vliv nemá. Je zřejmé, že podělení původní časové matice hodnotou $2^{\texttt{parDelta\_}}$ má největší vliv na počet operací zpětného násobení matic inicializační části algoritmu (funkce `InitCoefMatrix`). Relativní odchylky jsou s rostoucí hodnotou tohoto parametru menší, jelikož se zpřesňuje násobení pravděpodobnostních matic a výpočet pravděpodobnosti

uvíznutí v absorpčním stavu. Vzhledem k původním odhadům z disertační práce vedoucího [13], kdy odchylky nezřídka dosahovaly i desítek tisíc procent, byla díky naměřeným odchylkám v řádech 10^{-7} považována za „dostatečně“ přesnou.

4.1.2 Vyhodnocení parametru `destmista_`

Při testování parametru `destmista_` již nebyl brán příliš velký zřetel na míru odchylky, jelikož při tomto testování je odchylka přímo závislá na počátečním počtu desetinných míst, se kterými se v matici pracuje. Jde tedy jen o jistotu, kolik platných cifer nám výpočet zaručí. Porovnávaly se tedy pouze časové údaje. Z nich je patrné, že tento parametr nemá téměř vliv ani na výpočet první inicializační matice, ani pak na samotné hlavní tělo algoritmu – v časech se pohybujeme v řádu setin sekundy. Z tohoto důvodu nám tedy postačilo vzít nejvyšší testovanou hodnotu – 100 desetinných míst – která zaručuje vyšší přesnost.

4.1.3 Vyhodnocení parametru `parSkala_`

Posledním z testovaných parametrů je parametr `parSkala_`. Tento parametr nemá vliv na relativní odchylku, jelikož pouze přidává více či méně bodů na křivku mezi jednotlivé dva časové úseky (2^x – 2^{x+1}, $x \in \mathbb{N}_0$). Z tohoto důvodu se tedy znovu jednalo pouze o výpočetní dobu, za kterou daný běh skončí. Z tabulky, která je vyobrazena v kapitole Testování, je patrné, že tento parametr nemá téměř žádný vliv ani na výpočet první inicializační matice (funkci `InitCoefMatrix`). Ovšem vzhledem k potřebě více výpočtů jednotlivých časových posunů v hlavní části algoritmu vyplývá, že doba výpočtu se v závislosti na velikosti tohoto parametru lineárně zvyšuje. A to tak, že do hodnoty parametru 2^4 se stále pohybujeme pouze v řádech jednotek sekund, avšak například při hodnotě 2^{10} se již pohybujeme v řádech téměř minut. Z tohoto důvodu bylo usouzeno, že počet $2^4 = 16$ bodů mezi dvěma časovými úseky je přijatelný kompromis s ohledem na výpočetní dobu.

4.2 Vyhodnocení algoritmu pro Markovské řetězce

4.2.1 Vyhodnocení algoritmu pro nehomogenní řetězce

Vzhledem k tomu, že zadání se zmiňuje převážně o nehomogenních Markovských řetězcích, bylo tedy potřeba otestovat tento algoritmus i na nehomogenní verzi. Při ní se mezi jednotlivými časovými úseky znovu vypočítávala inicializační matice (byla opakovaně volána funkce `InitCoefMatrix`). Tato skutečnost měla za následek velký nárůst výpočtu hlavní části algoritmu (inicializační část byla vypočítávána stejně jak v nehomogenním, tak v homogenním řešení). Je to ovlivněno tím, že se mnohonásobně víckrát inicializovalo,

tudíž se musela tato časová doba promítnout i do výsledného času. To však nemělo za následek snížení přesnosti, jelikož se relativní odchylka pohybuje v rozmezí řádově 10^{-8} – 10^{-7}. Tímto výsledkem je naznačeno, že algoritmus má na tomto modelu velmi přesné výsledky jak pro homogenní, tak i pro nehomogenní řešení. Je však možné, že se jedná pouze o daný testovací model. Použití na jiných modelech bude případně součástí vědecké činnosti na doktorském studiu.

4.2.2 Vyhodnocení algoritmu pro hierarchické modely řetězců

V průběhu testování bylo patrné, že některé modely, které se testovaly, byly časově velmi zdlouhavé, a proto se mohlo také otestovat, zda-li řešení hierarchického modelu, tedy rozdělení výpočtu na část modelu 2oo2 a část N-MR, bude mít lepší rychlost a stejně přesné výsledky. Z těchto testů je patrné, že relativní odchylky jsou stále v rozumných mezích (řádově 10^{-8} – 10^{-6}), ale čas výpočtu hierarchického řešení enormním způsobem snížil dobu oproti exaktnímu řešení. Je to způsobeno tím, že při výpočtu exaktního řešení, se násobí matice kartézského součinu všech stavů, zatím co u hierarchického modelu se jedná o postupné násobení matic $5 * 5$ a následné spuštění dat na matice $n * n$, kde n je počet N-MR bloků. Tato možnost by však nešla využít, pokud bychom daný algoritmus neměli vytvořen pro nehomogenní řetězce, jelikož výsledek z bloku 2oo2 byl použit jako intenzita poruchy λ do modelu N-MR. Relativní odchylky jsou vyobrazeny v grafu 3.5 v kapitole Testování.

4.2.3 Vyhodnocení algoritmu pro analytické řešení řetězců

Poslední a téměř nejzásadnější testování proběhlo proti analytickému řešení modelu N-MR. Nejdůležitější pro tyto testy byla relativní odchylka, a to jak prvního vzorku, tak i celkový průměr. Z testování je patrné, že výsledné výpočty jsou poměrně přesné (řádově 10^{-8} až 10^{-7}).

4.3 Hodnocení s ohledem na budoucí tvorbu

Výzkumná skupina DDD na Katedře číslicového návrhu na Fakultě informačních technologií Českého vysokého učení technického v Praze se problematikou spolehlivosti zabývá již od jejího vzniku. Vedoucí diplomové práce je právě jedním z těch, kteří se problematikou intenzivně zabývají, a jejichž výsledky jsou sepsány v mnoha článcích z různých konferencí, v disertační práci i v dalších pracích. Avšak jak už bylo zmíněno, metody uvedené v [13] jsou schopné pouze velmi hrubých pesimistických odhadů nebo odhadů pro omezený časový interval (v takovém případě byly výsledky použitelné pouze do specifikovaného časového okamžiku). Proto je tato metoda v diplomové práci výrazným vylepšením původní metody, jelikož dosahuje větší přesnosti a je

použitelná po celou životnost zařízení, aniž by došlo k výraznému zpomalení výpočtu. Z těchto důvodů byly také sepsány již zmíněné články na konference DTIS 2019 a DSD 2019.

Závěr

Tato diplomová práce se zabývala spolehlivostními modely a výpočty jejich spolehlivostních parametrů v čase na základě proměnných intenzit poruch. Jedním z cílů bylo seznámení se s metodami pro tyto výpočty. Z první kapitoly je patrné, že tato metoda se velice podobá standardním výpočtům homogenních Markovských řetězců, avšak je rozšířena o přesnější výpočty jednotlivých časových úseků, které díky tomu mohou být ovlivněny změnou intenzity jednotlivých poruch. V další kapitole je popsán algoritmus vybrané metody vytvořený v programu Wolfram Mathematica, který se odráží od již připravených dat z disertační práce pana Ing. Martina Kohlíka, Ph.D. Součástí další kapitoly je detailní porovnání časové náročnosti a přesnosti v závislosti na volbě vstupních parametrů, které byly požadovány, tak i na výpočtech původních modelů z disertační práce a na modelu pro homogenní řetězce. Celkový přínos tohoto algoritmu je popsán v poslední kapitole Hodnocení, kde je shrnut i postup v další tvorbě pro výzkumnou skupinu DDD.

Jak již bylo v práci zmíněno, výsledky naměřené touto metodou přináší výrazné vylepšení oproti předchozím metodám ať už co se týče časové náročnosti, ale hlavně přesnosti. Dále je z měření vidět, že v případě potřeby má metoda možnosti pro další zpřesnění. To bude mít v některých ohledech vliv na časovou náročnost, což také záleží na daném výpočetním zařízení, na kterém je výpočet spuštěn. V budoucích pracích je tedy prostor pro tento typ testování.

Předpokládaná metoda výpočtu spolehlivosti by mohla mít v budoucnu velký potenciál v tomto oboru, proto již během přípravy této diplomové práce byly sepsány dva vědecké články na mezinárodní konference DTIS 2019 a DSD 2019. V současné situaci během vypracování práce byl článek na konferenci přijat a obhájen jako poster a jeho kratší verze poté vydána ve sborníku. Druhý článek je zatím pouze v recenzním řízení, avšak při obhajobě této diplomové práce bude již znám výsledek. Díky těmto výsledkům je celkem zřejmé, že tato metoda je správným krokem ve směru, kam se ubírat při řešení spolehlivosti nejen Markovských řetězců, ale i jiných modelů.

Na samotném algoritmu jsem pracoval již od začátku navazujícího magisterského studia, kdy byly veškeré postupy konzultovány a přesnost výpočtů se postupně zlepšovala. Testování probíhalo opakovaně, a to z důvodů postupného přidávání parametrů, ověření jejich správnosti a kontroly, zda po přidání parametru nedošlo k znepřesnění výpočtu. Například při testu exaktního řešení pro 9-MR blok, kdy jeden výpočet zabral i několik desítek minut, byla jakákoliv změna parametru nebo algoritmu výrazným prodloužením doby práce na výpočtech.

Již během práce vznikly debaty ohledně toho, že se nejedná o jediný způsob, jakým se dají tyto modely a jejich spolehlivosti počítat. Veškeré poznatky, které vznikly během práce byly konzultovány a zapsány pro budoucí využití – ať už jako postup pro dizertační práci, tak i případné zadání jiných bakalářských, případně diplomových prací. Dále se v rámci budoucího doktorského studia předpokládá širší konzultace i s jinými obory (například katedrou aplikované matematiky) nebo úprava výpočtu pro další modely, převážně z průmyslového prostředí.

Jak již bylo zmíněno dříve, předpokládá se, že tato metoda nemusí mít pouze uplatnění na poli Markovských řetězců a jejich výpočtů spolehlivosti, resp. jejich distribuční funkce (selhání) $F(t)$. Tato metoda by do budoucna mohla spolupracovat s jakýmikoliv modely, které se zaměřují na spolehlivost a pracují s touto distribuční funkcí. To by mohlo mít velice pozitivní vliv i na další spolupráci, ať už v podmínkách vědecké tvorby, tak i v průmyslovém odvětví. Na tyto vlivy je samozřejmě brán zřetel a předpokládá se, že během budoucích studií se bude k této metodě vracet a na jejím základě bude stavěn další postup prací.

Literatura

[1] Norma ČSN EN 62347:2007(01 0696) [online]. 2007. Dostupné z: http://csnonlinefirmy.unmz.cz/html_nahledy/01/79665/79665_nahled.htm

[2] Fišer, P.: Spolehlivost systémů, zvyšování spolehlivosti [online]. 2018, přednáška z předmětu MI-TSP na FIT ČVUT v Praze. Dostupné z: https://moodle.fit.cvut.cz/mod/page/view.php?id=57411

[3] Electronic Reliability Design Handbook – MIL-HDBK-338B [online]. 1998, US Department of Defense. Dostupné z: http://www.weibull.com/mil_std/mil_hdbk_338b.pdf

[4] Reliability Prediction of Electronic Equipment – MIL-HDBK-217F [online]. 1995, Notice 2., US Department of Defense. Dostupné z: http://www.weibull.com/mil_std/mil_hdbk_217f_2.pdf

[5] Shooman, M. L.: Reliability of Computer Systems and Networks: Fault Tolerance, Analysis and Design (Appendix A: Summary of Probability Theory). 2002, John Wiley & Sons, Inc., New York, NY, USA.

[6] Hlavička, J.; Racek, S.; Golan, P.; aj.: Číslicové systémy odolné proti poruchám. 1992.

[7] Ocaña Rilola, R.: Two Methods To Estimate Homogenous Markov Processes. 2002, Journal of Modern Applied Statistical Methods: Vol. 1, Iss. 1, Article 17.

[8] Hrabák, P.; Vašata, D.: Klasifikace stavů Markovského řetězce [online]. 2019, přednáška z předmětu MI-SPI.16 na FIT ČVUT v Praze. Dostupné z: https://courses.fit.cvut.cz/MI-SPI/lectures/files/mi-spi-lec-15-slides.pdf

[9] Weisstein, E. W.: Chapman-Kolmogorov Equation [online]. From MathWorld–A Wolfram Web Resource. Dostupné z: http://mathworld.wolfram.com/Chapman-KolmogorovEquation.html

[10] Wolfram: Wolfram Mathematica [online]. Květen 2019, [Citováno 2019-05-06]. Dostupné z: http://www.wolfram.com/mathematica/

[11] Řezníček, J.; Kohlík, M.; Kubátová, H.: Hierarchical Dependability Models based on Non-Homogeneous Continuous Time Markov Chains. 2019, Design & Technology of Integrated Systems, DTIS 2019.

[12] Řezníček, J.; Kohlík, M.; Kubátová, H.: Accurate Inexact Calculations of Non-Homogeneous Markov Chains. 2019, Digital System Design, DSD 2019, Euromicro Symposium.

[13] Kohlík, M.: Hierarchical Dependability Models Based on Markov Chains, A dissertation thesis. 2015.

[14] Dobiáš, R.; Kubátová, H.: FPGA based design of the railway's interlocking equipments. 2004, Digital System Design, DSD 2004, Euromicro Symposium.

Příloha **A**

Seznam použitých zkratek

2oo2 – 2-out-of-2 blok, blok „dva-ze-dvou“

CPU – Central processing unit, procesorová jednotka počítače

DDD – Digital Design & Dependability, výzkumná skupina

DSD – Digital System Design, konference

DTIS – Design & Technology of Integrated Systems, konference

HW – Hardware

***N*-MR** – *N*-Modular Redundancy, *N*-Modulární redundance

PI – Primary Input, primární vstup

PO – Primary Output, primární výstup

SW – Software

TMR – Triple-Modular Redundancy, Tří-Modulární redundance

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `data` ..... adresář s testovacími daty
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `DP_Reznicek_Jan_2019.pdf` ..... text práce ve formátu PDF