Západočeská univerzita
FAKULTA APLIKOVANÝCH VĚD

Okruhy otázek ke státní závěrečné zkoušce z předmětu
Systémové programování (SP)

Operační systémy (OS)
Paralelní programování (PPR)
Formální jazyky a překladače (FJP)
Výkonnost a spolehlivost čísl. systémů (VSP)

Studijní program:	3902	Inženýrská informatika
Obor:	2612T025	Informatika a výpočetní technika – Softwarové inženýrství
	3902T031	Softwarové inženýrství

Akademický rok: 2005/2006

Obsah:

1 Pravděpodobnostní modelování počítačových systémů – generování a využití náhodných čísel (Monte Carlo metody), matematické (markovské) modely

Experimentální pravděpodobnostní modelování (v nejjednodušší podobě) spočívá v programové realizaci většího množství pokusů s náhodně generovanými parametry pokusu a v následném statistickém vyhodnocení výsledků množiny pokusů. Numerické techniky pro generování a zpracování náhodných čísel se obecně nazývají metody Monte Carlo. Simulační model založený na experimentálním modelování provádí (v rámci jednoho simulačního experimentu) pouze transformaci množiny číselných parametrů modelu na množinu číselných výsledků. Narozdíl od matematického modelu je třeba pro získání nějakých závislostí výsledků na parametrech realizovat větší množství (časově náročných) pokusů.

Generování náhodných čísel

Používají se programově realizované generátory náhodných čísel, které z posledního (nebo několika posledních) prvku(ů) posloupnosti náhodných čísel počítají podle vhodně zkonstruovaného vzorce další prvek. Nejde tedy přímo o čísla náhodná, ale pseudonáhodná, což ovšem nevadí, pokud splňují požadované pravděpodobnostní rozdělení a sousední prvky posloupnosti (dvojice, trojice) jsou statisticky nezávislé. Naopak má tento způsob generování výhodu v reprodukovatelnosti.

Programově realizované (knihovní) generátory náhodných čísel zpravidla generují čísla s rovnoměrným pravděpodobnostním rozdělením na intervalech např. 0…1 či 0…max. Pokud jsou zapotřebí náhodná čísla s jiným než rovnoměrným rozdělením, vytváří se programově sekundární generátor, který vyvolává základní generátor s rovnoměrným rozdělením a výsledky transformuje na požadované rozdělení.

Generování čísel s rovnoměrným rozdělením

Nejčastěji využívá tzv. kongruentní metody dle vzorce $y_{j+1} = (C_1 + C_2 y_j) \bmod M$, kde y_{j+1} je generovaný prvek náhodné posloupnosti, y_j je poslední prvek posloupnosti, C_1, C_2, M jsou číselné parametry generátoru a operátor mod realizuje výpočet zbytku po celočíselném dělení obou operandů. Generovaná posloupnost je periodická.

Pro generování náhodných čísel v jazyce C použijeme funkce z knihovny *stdlib.h* :
int rand (void) – vrací nezáporná celá čísla <0, RAND_MAX>
void srand (unsigned seed) – umožňuje nastavit výchozí hodnotu
int random (int num) – vrací nezáporná čísla od 0 do num-1
void randomize (void) – počáteční nastavení generátoru

Metoda inverzní transformace

Jedná se o metodu umožňující transformaci náhodných čísel Y s normalizovaným rovnoměrným rozdělením na čísla X zadaná distribuční funkcí F(x) jejich pravděpodobnostního rozdělení. Generátorem normalizovaného rovnoměrného rozdělení tedy vyrobíme konkrétní hodnoty y a transformujeme je na odpovídající x podle vzorce $x = F^{-1}(y)$.

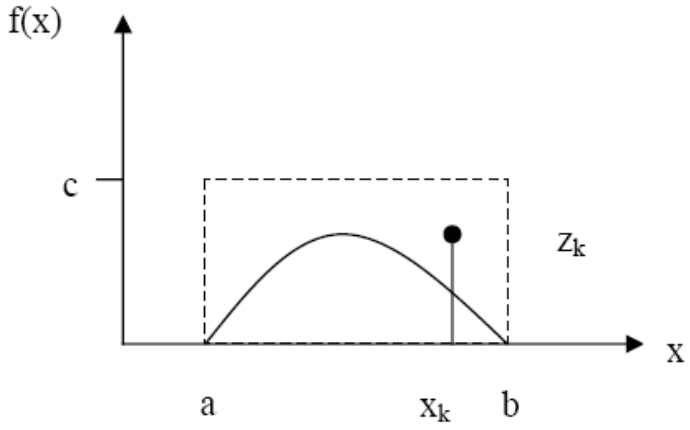
Diskrétní rozdělení

Princip metody inverzní transformace lze použít i pro generování náhodných čísel s diskrétním rozdělením. Interval <0,1) na ose reálných čísel rozdělíme na n dílčích intervalů s

velikostí p_i. Základním generátorem vytvoříme konkrétní číslo z výběru Y (v intervalu 0…1) a zjistíme číslo i dílčího intervalu, ve kterém se vyskytuje. Výstupem generátoru bude odpovídající hodnota x_i.

Vylučovací metoda

Vycházíme z hustoty pravděpodobnosti f(x) či z tabulky hodnot.



Algoritmus metody:

- Generujeme náhodné číslo x_k z výběru s rovnoměrným rozdělením na intervalu <a,b), tedy podle vzorce $x_k = a + (b-a)y$, kde y je číslo poskytnuté základním generátorem.
- Generujeme náhodné číslo z_k z výběru s rovnoměrným rozdělením na intervalu <0,c), tedy podle vzorce $z_k = cy$, kde y je (další) číslo poskytnuté základním generátorem.
- Čísla x_k,z_k interpretujeme jako souřadnice bodu v rovině.
- Pokud je náhodně generovaný bod pod křivkou funkce f(x), postup končí a výstupem generátoru je hodnota x_k, v opačném případě postup opakujeme počínaje prvním bodem. Index k v tomto případě slouží jako čítač obrátek cyklu opakování.

Kompoziční metoda

Vycházíme z hustoty pravděpodobnosti f(x), která nemusí být zadána analyticky. Používá se, pokud lze hustotu f(x) rozumně aproximovat funkce po částech konstantní či lineární.

Algoritmus metody:

- Nejprve náhodně generujeme číslo dílčího intervalu i pomocí generátoru s diskrétním rozdělením hodnot i podle pravděpodobností p_i.
- Dále generujeme náhodné číslo spadající do vybraného intervalu. Toto číslo generujeme podle charakteru funkce hustoty v příslušném intervalu. Je-li v nejjednodušším případě hustota pravděpodobnosti v intervalu konstantní (tj. rovnoměrné rozdělení), je konečným výstupem generátoru číslo s rovnoměrným rozdělením vypočítané podle vzorce $x = a_i + (b_i - a_i)y$, kde a_i a b_i jsou meze i-tého dílčího intervalu a y představuje konkrétní hodnotu poskytnutou generátorem normalizovaného rovnoměrného rozdělení.

Generování čísel s normálním rozdělením

Pro normální (gaussovské) rozdělení lze využít centrální limitní větu teorie pravděpodobnosti, která tvrdí, že součet náhodných čísel s libovolným rozdělením má asymptoticky normální

rozdělení se střední hodnotou a rozptylem danými součtem středních hodnot a rozptylů pravděpodobnostních rozdělení prvků součtu. Pomocí několika dalších úvah dostaneme vzorec: $x = a + \sigma\left(\left(\sum_{j=1}^{12} y_j\right) - 6\right)$, kde a je střední hodnota a σ je směrodatná odchylka.

Metoda Monte Carlo

Je to celá třída algoritmů pro simulaci systémů. Jde o stochastické metody používající náhodná čísla. Typicky využívány pro výpočet integrálů, zejména vícerozměrných, kde běžné metody nejsou efektivní. Výhodou je jednoduchá implementace, nevýhodou relativně malá přesnost. Metoda Monte Carlo je založena na provádění náhodných experimentů s modelem systému a jejich vyhodnocení. Je třeba mít kvalitní generátory pseudonáhodných čísel (netřeba skutečně náhodná čísla). Výsledkem provedení velkého množství experimentů je obvykle pravděpodobnost určitého jevu. Na základě získané pravděpodobnosti a známých vztahů pak spočítáme potřebné výsledky.

Jednoduše řečeno, náhodně generujeme posloupnost vstupních dat a sbíráme výstupní data po provedení modelu. Tyto data posléze analyzujeme a vyhodnocujeme z nich patřičné závěry.

Matematické (markovské) modely

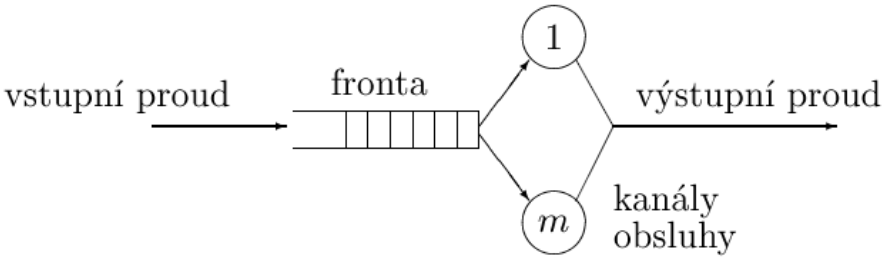
Jsou založeny na exaktním matematickém popisu vlastností a vazeb prvků originálu (např. soustavou diferenciálních rovnic). Matematické modely se obecně vyznačují vysokým stupněm zjednodušení a tedy větším odstupem od reality. Na druhé straně lze ale exaktně ověřit správnost modelu a jednoduše analyzovat závislost výsledku na parametrech modelu.

2 Základy teorie systémů hromadné obsluhy, elementární obslužný systém, Kendallova klasifikace, systémy M/M/1 a M/G/1

Systém hromadné obsluhy (SHO) – model reálného systému, jehož funkce spočívá v realizaci obsluhy (poskytování služby) pro velké počty průběžně přicházejících požadavků (transakcí). Služby jsou poskytovány prvky označovanými jako kanály obsluhy (servery). Před kanálem obsluhy se často vytváří fronta požadavků čekajících na poskytnutí služby. Cílem SHO je určit výkonnostní charakteristiky systému. Příkladem SHO je počítačový systém, dopravní síť, … SHO lze konstruovat jako matematický či simulační model (pokud nejde dostatečně zjednodušit).

Elementární SHO

Základním předpokladem je časová stálost zdrojových parametrů modelovaného systému (modelujeme ustálený provoz nikoliv přechodový děj).



Zdroj požadavků

Zdroj požadavků může být buď omezený (konečný počet) nebo neomezený (nezávisí na stavu SHO).

Požadavky vstupují do SHO *náhodně*, doba mezi jejich příchody je nazývána *interval příchodů*. Nejčastěji využívaným způsobem matematického popisu vstupního proudu je zadání *distribuční funkce pravděpodobnostního rozdělení* $F_a(t)$ nebo odpovídající *hustoty pravděpodobnosti* $f_a(t)$. Nejčastěji využívaným typem vstupního proudu je tzv. *poissonovský vstupní proud*, ve kterém má interval příchodů *exponenciální rozdělení*.

Požadavky přicházející v poissonovském proudu jsou nahodilé v tom smyslu, že pro náhodně vybírané stejně velké (ale velmi malé ve srovnání se střední dobou intervalů příchodů) časové intervaly je stále stejná pravděpodobnost příchodu požadavků.

Dále uvedené veličiny a vztahy charakterizují vstupní proud:

$F_a(t)$	- distribuční funkce pravděpodobnostního rozdělení časového intervalu mezi příchody požadavků
$E\{\tau\} = T_a = 1/\lambda$	- střední hodnota intervalu mezi příchody (střední perioda příchodů), λ představuje *střední frekvenci příchodů požadavků*
$F_a(t) = 1 - e^{-\lambda t}$	- distribuční funkce časových intervalů mezi příchody v poissonovském proudu; Střední frekvence příchodů λ je jediným parametrem rozdělení.

Poissonovský proud je tedy možné charakterizovat právě touto hodnotou.

$C_a = \sigma\{\tau\}/T_a$ - tzv. *koeficient variace*, a $\sigma\{\tau\}$ je směrodatná odchylka intervalu mezi příchody. Koeficient variace udává *nahodilost* (resp. *pravidelnost*) příchodů; pro zcela pravidelné příchody má hodnotu 0, pro příchody v poissonovském proudu má hodnotu 1. V reálných případech se většinou jeho hodnota pohybuje mezi 0 a 1.

Fronta požadavků

Je charakterizována *maximální délkou* (velikostí) fronty a *frontovou disciplínou*, tj. pravidlem, podle kterého se z fronty vybírá. Délka fronty může být *omezená* (v některých případech i nula), nebo *neomezená* – zpravidla se používá pro zjednodušení matematického řešení. Frontová disciplína může být *bez priorit* nebo *prioritní*, prioritní disciplíny se dále mohou dělit na disciplíny *s přednostním vyloučením* nebo *bez přednostního vyloučení*.

K popisu fronty se dále zavádějí tyto veličiny:

w - okamžitý počet požadavků ve frontě

$E\{w\} = L_w$ - střední počet požadavků ve frontě, tj. *střední délka fronty*

t_w - doba čekání konkrétního požadavku ve frontě

$E\{t_w\} = T_w$ - střední doba čekání požadavků ve frontě

Kanál obsluhy

Počet kanálů obsluhy značíme *m*, nejčastějším případem je jednokanálová obsluha, tj. $m = 1$. V SHO je t_s chápáno jako doba obsluhy konkrétního požadavku – obvykle náhodná veličina s daným pravděpodobnostním rozdělením. Základní charakteristikou kanálu je opět distribuční funkce rozdělení - $F_s(t)$.

Veličiny a vztahy charakterizující jeden kanál obsluhy:

$F_s(t)$ - distribuční funkce pravděpodobnostního rozdělení doby obsluhy

$E\{\tau\} = T_s = 1/\mu$ - střední doba obsluhy, veličina μ je *střední frekvence obsluh* a je jediným parametrem kanálu obsluhy

$F_s(t) = 1 - e^{-\mu t}$ - distribuční funkce exponenciálního rozdělení doby obsluhy

$C_s = \sigma\{t_s\}/T_s$ - koeficient variace doby obsluhy, a $\sigma\{t_s\}$ je směrodatná odchylka doby obsluhy. Koeficient variace udává *nahodilost* (resp. *pravidelnost*) příchodů; pro shodné, konstantní obsluhy má hodnotu 0, pro obsluhy s exponenciálním rozdělením má hodnotu 1. V reálních aplikacích se pak hodnoty pohybují mezi 0 a 1.

Výstupní proud požadavků

Jedná se o proud požadavků po průchodu obslužným systémem. Je-li SHO ve stacionárním režimu ($\rho < 1$), pak všechny požadavky, které vstoupí do SHO jsou v konečném čase obslouženy a střední frekvence i perioda odchodů požadavků je shodná se střední periodou frekvencí příchodů.

Ve stacionárním režimu si poissonovský vstupní proud po průchodu obslužným systémem s exponenciálním rozdělením doby obsluhy zachová poissonovský charakter (pravděpodobnostní rozdělení intervalů mezi příchody i mezi odchody je exponenciální, navíc se stejným parametrem λ).

Kendallova klasifikace elementárních SHO

Klasifikace využívá značení *A/B/m*

- **A** – typ pravděpodobnostního rozdělení vstupních intervalů,
- **B** – typ pravděpodobnostního rozdělení doby obsluhy,
- **m** – symbol pro označení počtu identických kanálů obsluhy v elementárním SHO.

Symboly používané pro pravděpodobnostní rozdělení:

- **GI** – Vstupní proud s nezávislými intervaly mezi příchody a jakýmkoliv pravděpodobnostním rozdělením těchto intervalů.
- **G** – Pro obecné rozdělení doby obsluhy.
- **M** – Pro exponenciální rozdělení doby obsluhy nebo intervalu příchodů.
- **D** – Deterministické (pravidelné) intervaly příchodů nebo doby obsluhy.

Někdy se ještě doplňují údaje o maximální délce fronty a typu fronty – např. M/D/1/∞/FIFO. V tomto případě se jedná o SHO s exponenciálním rozdělením intervalu příchodů, nenáhodnou dobou obsluhy pro všechny požadavky, jedním kanálem obsluhy a frontou FIFO s neomezenou kapacitou. Běžně se ale v tomto případě uvádí jen M/D/1.

Veličiny a vztahy pro elementární SHO

Charakteristické veličiny pro SHO:

q - okamžitý celkový počet požadavků v SHO (tj. ve frontě i v kanálech obsluhy)

$E\{q\} = L_q$ - střední počet požadavků v SHO

t_q - doba průchodu SHO pro jeden požadavek, též *doba obsluhy*

$E\{t_q\} = T_q$ - střední doba průchodu požadavků, též *střední doba obsluhy*

$\rho = \frac{1}{m}\frac{T_s}{T_a} = \frac{1}{m}\frac{\lambda}{\mu}$ - *zatížení obslužných kanálů SHO* (např. $\rho = 0{,}5$ udává, že kanál obsluhy je vytížen na 50%)

$u = \lambda / \mu$ - *intenzita provozu SHO*

Nutná podmínka pro dosažení stacionárního režimu: $\rho < 1$**!**

Pro elementární SHO dále platí:

$$L_q = L_w + L_s = L_w + m\frac{\lambda}{\mu}$$

$$T_q = T_w + T_s = T_w + \frac{1}{\mu}$$

Dále platí tzv. **Littleovy vzorce** (pro SHO s neomezeným počtem míst ve frontě):

$L_q = \lambda T_q$

$L_w = \lambda T_w$

Dále si můžeme uvést vztah pro výpočet koeficientu variace výstupního proudu C_0, který spočteme jako $C_0 \doteq \sqrt{1+\rho^2(C_s^2-1)+(1-\rho^2)(C_a^2-1)}$, kde C_a je koeficient variace vstupního proudu, ρ je zatížení a C_s je koeficient variace doby obsluhy.

M/M/1

- základní elementární SHO, který slouží jako porovnávací případ pro jiné
- poissonovský vstupní proud – parametr λ (střední frekvence proudu a zároveň parametr exponenciálního rozdělení)
- doba obsluhy má exponenciální rozdělení – parametr μ
- zatížení $\rho = \frac{\lambda}{\mu}$
- střední počet požadavků v SHO $L_q = \frac{\rho}{1-\rho}; \rho < 1$
- Příklad výpočtu – skripta strana 43

M/G/1

- poissonovský vstupní proud – parametr λ (dostatečně realistický vstupní proud pro velké množství aplikací)
- obecné rozdělení doby obsluhy, dané funkcí $F_s(t)$ nebo hustotou $f_s(t)$
- zatížení určíme přímo jako $\rho = \lambda T_s$, kde T_s je střední hodnota rozdělení $F_s(t)$
- $L_w = \frac{\rho^2}{2(1-\rho)}(1+C_s^2)$
- Příklad výpočtu – skripta strana 45

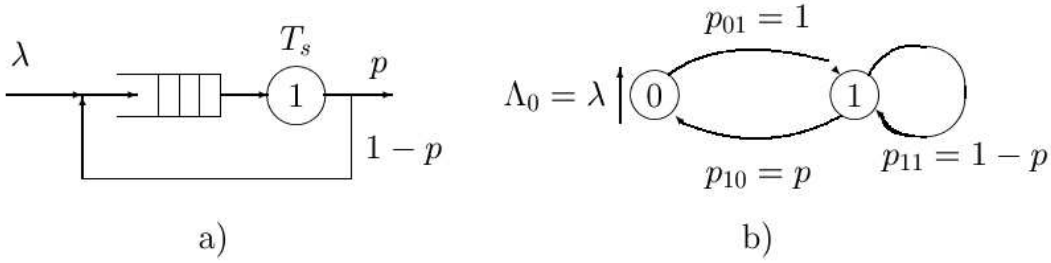
3 Matematické modely sítí front, určení výkonnostních parametrů (doba odezvy, délka front, průchodnost)

Navazuje na otázku 2

Komplikovanější reálné systémy mohou obsahovat větší počet kanálů obsluhy, každý s vlastní frontou požadavků. Požadavky vstupující do systému přecházejí mezi jednotlivými obsluhami a po ukončení všech dílčích obsluh vystupují ven ze systému. Takovýto systém lze modelovat jako síť skládající se z elementárních SHO. Sítě se vstupy z okolí se nazývají *otevřené sítě*. Dalším typem jsou sítě uzavřené.

Otevřené sítě front

Otevřené sítě front se znázorňují pomocí orientovaného grafu, jehož uzly odpovídají elementárním SHO – jejich počet označujeme jako *n* a číslujeme je tedy od *1* do *n*. Jako váhy hran pak používáme obvykle tzv. *pravděpodobnosti větvení*, tj. p_{ij} udává pravděpodobnost, že požadavek půjde po ukončení obsluhy v uzlu *i* do uzlu *j*. Okolí obslužného systému (zdroj požadavků) lze modelovat zvláštním uzlem obvykle s indexem 0, hrany s váhami p_{0j} pak modelují vstupy požadavků z okolí do j-tého serveru a hrany s váhami p_{i0} představují odchod požadavků ze sítě po dokončení i-té obsluhy.



Střední frekvence a zatížení uzlů

Dále rozlišujeme *vnitřní tok požadavků uzlem* a *tok požadavků mezi uzly*. Střední frekvenci vnitřního toku uzlem *i* značíme Λ_i, Λ_0 značíme celkový tok požadavků vstupujících z okolí do sítě. Rozdělují-li se požadavky v *i*-tém podle pravděpodobností p_{ij}, bude pak tok požadavků z uzlu *i* do uzlu *j* dán jako $\Lambda_{ij} = \Lambda_i p_{ij}$. Ve stacionárním režimu tedy musí platit $\sum_k \Lambda_k p_{ki} = \Lambda_i = \sum_j \Lambda_i p_{ij}$, kde index *k* nabývá hodnot odpovídajících vstupním hranám, index *j* výstupním hranám uzlu *i*.

Hodnoty frekvencí Λ_i získáme jako řešení soustavy *n* rovnic, pro jejichž sestavení potřebujeme znát střední frekvenci toku vstupujícího do sítě Λ_0 a matici pravděpodobností větvení $\{p_{ij}\}$. Hodnoty Λ_i je možné získat řešením lineárních algebraických rovnic.

Kontrolu stacionárního režimu činnosti jednotlivých uzlů pak můžeme provést výpočtem zatížení. Pro tento výpočet potřebujeme ještě znát počet kanálů obsluhy m_i a střední dobu obsluhy T_{si}.

Zatížení *i*-tého uzlu je pak dáno jako $\rho_i = \frac{1}{m_i}\Lambda_i T_{si}$ a musí být samozřejmě vždy menší než jedna.

Příklad viz skripta strana 48.

Délka fronty a doba odezvy

Jednoduché matematické řešení sítě front je možné pouze za následujících předpokladů (tzv. Jacksonova teorému):

- Všechny toky z okolí do sítě mají poissonovský charakter.
- Všechny obslužné uzly mají exponenciální rozdělení doby obsluhy se střední hodnotou T_{si}.
- Po ukončení obsluhy v uzlu i požadavek bez zpoždění přejde náhodně (s pravděpodobností p_{ij} do uzlu j.

Za těchto předpokladů lze považovat každý uzel za M/M/1 s frekvencí vstupního toku Λ_i a střední dobou obsluhy T_{si}. Jednotlivé uzly pak řešíme samostatně, čímž získáme hodnoty veličin L_{wi}, L_{qi}, T_{wi} a T_{qi}.

Pro celou síť pak lze určit průměrný počet požadavků L_q akumulovaných v síti a střední dobu T_q průchodu požadavku sítí jako $L_q = \sum_{i=1}^{n} L_{qi}$ a $T_q = \frac{1}{\Lambda_0} L_q$.

Pokud nejsou uvedené předpoklady splněny a některé příchody či obsluhy jsou pravidelnější než exponenciální (koeficient variace je menší než 1), slouží výše uvedený postup jako jednostranný odhad, který dává nejhorší možné hodnoty.

Uzavřené sítě front

Uzavřené sítě front nemají žádné vstupy požadavků z okolí. V síti „koluje“ pevná populace požadavků, které modelují nějakou aktivitu s omezenou dobou trvání (u otevřených sítí není aktivita časově omezená). V tomto případě nás zajímají typicky střední frekvence průchodů požadavků určitým místem sítě. Tyto frekvence pak vedou k určení výkonnostních ukazatelů propustnosti systému (tj. střední počet operací vykonaných za jednotku času). Analytické řešení lze provádět, pokud doby mají exponenciální pravděpodobnostní rozdělení.

4 Simulační modely počítačových systémů, pseudo-paralelní výpočetní procesy v modelovaném čase, objektová dekompozice modelu, programové nástroje (Simula, C-Sim, J-Sim – principy použití)

4.1 Simulační modely počítačových systémů

Experimentální pravděpodobnostní modelování, v nejjednodušší podobě, spočívá v programové realizaci většího množství pokusů s náhodně generovanými parametry pokusu a v následném statistickém vyhodnocení výsledků množiny pokusů. Pro numerické techniky založené na generování a zpracování náhodných čísel se často používá název metody Monte Carlo. Pro tyto metody obecně platí, že nejsou příliš přesné (v řádu jednotek procent) a jsou náročné na čas výpočtu (je třeba velké množství pokusů). Náhodné pokusy jsou realizovány programovými generátory náhodných čísel. Metody Monte Carlo jsou dobře paralelizovatelné, neboť pokusy není třeba synchronizovat a lze je provádět paralelně. Metody Monte Carlo se používají hlavně pro modelování úloh se stochastickým[1] charakterem. Problém s modelování nastává, pokud nelze jednoduše oddělit jednotlivé pokusy. Například pokud je v modelu SHO více požadavků. Proto se používá diskrétního modelového času, který slouží jako prostředek pro uspořádané provádění fází pokusů. Nejběžnější způsoby dekompozice modelu reálného diskrétního systému využívající techniku experimentálního pravděpodobnostního modelování se nazývají:

- metoda interpretace událostí
- metoda pseudo-paralelních procesů

4.2 Metoda interpretace událostí

Všechny události, které mají nastat v budoucím vývoji modelu (vztaženo k aktuální hodnotě modelového času), jsou vedeny v seznamu nazývaném kalendář událostí. Seznam je setříděn vzestupně podle hodnoty modelového času vzniku události. Řídící algoritmus simulačního výpočtu postupně interpretuje události nacházející se v kalendáři. Interpretace probíhá v diskrétním bodě modelového času vzniku události a je realizována vyvoláním interpretačního podprogramu příslušného typu události. Interpretační podprogram provede změny v modelu voláním metod objektů, kterých se událost týká. Pokud je interpretovaná událost přímou příčinou další události v budoucím modelovém čase, je součástí činnosti některé volané metody objektu naplánování této nové události. Interpretovaná událost je vymazána z kalendáře.
Pro realizaci tohoto způsobu výpočtu vystačíme s filosofií objektů poskytovanou konvenčními prostředky pro OOP.

4.3 Metoda pseudo-paralelních procesů

Metoda pseudo-paralelních procesů je založena na objektové dekompozici řídícího algoritmu simulačního výpočtu. Jednotlivé části jsou zapouzdřeny ve vybraných třídách objektů, které tím získávají vlastní program a charakter samostatných výpočetních procesů vykonávaných pseudoparalelně.
Některé objekty modelu mají pasivní charakter, tj. nevyvíjí v modelovém čase žádnou vlastní aktivitu a pouze poskytují služby pro jiné objekty. Jiné modely objektu, dále označované jako procesy, mají aktivní charakter, tedy vyvíjí aktivitu podle svého programu.

1 Stochastický - Náhodný

Aktivita procesu je členěna do sekvence tzv. fází aktivity, probíhajících vždy v jednom konkrétním bodě diskrétního modelového času. Výsledkem činnosti – fáze aktivity – je změna atributů objektu – procesu, případná změna atributů jiných objektů a případná změna stavu jiných procesů (aktivace). Intervaly modelového času mezi fázemi aktivity určitého procesu se nazývají úseky nečinnosti.
Řízení pseudo-paralelního výpočtu všech procesů v modelu opět vyžaduje existenci datové struktury s charakterem kalendáře. V kalendáři jsou vzestupně podle hodnoty modelového času setříděny události, jejichž záznam obsahuje ještě odkaz do programu procesu. Řídící smyčka výpočtu postupně spouští procesy v pořadí daném záznamy v kalendáři.

4.4 Objektová dekompozice modelu

Objekty reálného světa jsou často aktivní tj. vykonávají nějakou činnost, která probíhá souběžně s činností jiných objektů a modelovat je programovými objekty s charakterem výpočetních procesů je přímočaré a přirozené. Hlavní výhodou dekompozice modelu na aktivních objektech (procesech) je tedy větší schopnost modelovat realitu. Navíc je dekompozice čistě objektová, protože program vykonávaný objektem (procesem) lze jednoznačně přiřadit k jeho třídě. Čistě objektová dekompozice přináší také výhody objektového přístupu v programování – zejména pak výhodu opakované použitelnosti (reusability) programového kódu. Na druhé straně vyžaduje metody paralelních procesů obecnější model objektu. Konkrétně se jedná o doplnění programu procesu do deklarace třídy objektu (procesu).

4.5 Programové nástroje

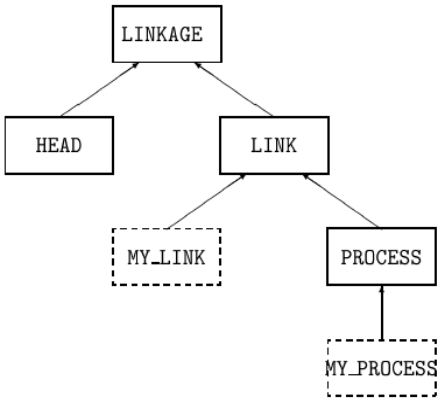
Jedná se o nástroje, které poskytují prostředky pro pseudo-paralelní procesy.

Simula

První jazyk, které toto poskytoval. Odvozena z Angolu 60. Využívá se jen jednoduchá dědičnost, objekty se vytváří pouze dynamicky, odkazuje se na ně výhradně pomocí referenčních proměnných. Uvolňování paměti nepoužívaných objektů provádí automaticky garbage collector. Nevýhodou Simuly je, že vyšla z jazyka, který je dnes překonán a nepoužívá se.

Základní objekty

- `SIMSET` – poskytuje prostředky pro obousměrně cyklické seznamy, systémové třída
- `SIMULATION` – systémová třída, je zde definován `PROCESS`
- `LINKAGE` – seznamy, nepoužívá se přímo, sdružuje společné vlastnosti `HEAD` a `LINK`
- `LINK` – prvek seznamu
- `HEAD` – hlava seznamu, operace nad celým seznamem
- `PROCESS` – potomek `LINK`, vlastní chování objektu (procesu)



Základní typy objektů pro konstrukci modelu sítě front

Významné vlatnosti (metody) třídy LINK

- `into(seznam)` – zařazení prvku na konec seznamu
- `follow(prvek)` – zařadíme náš prvek za prvek určený jako parametr metody
- `precede(prvek)` – zařazení před
- `out()` – odebrání prvku ze seznamu, není třeba uvádět seznam

Významné vlastnosti (metody) třídy HEAD

- `empty()` – test na prázdný seznam
- `cardinal()` – vrácí délku seznamu
- `first()` – vrací odkaz na první prvek seznamu
- `last()` – vrací odkaz na poslední prvek seznamu
- `clear()` – vyprázdní seznam

Významné vlastnosti (metody) třídy PROCESS

- `life()` – vlastní tělo (chování) procesu

C-Sim

Externí knihovna pro jazyk C, která využívá filosofické základy Simuly. Pro implementaci pseudoparalelních procesů využívá tzv. daleké skoky.

J-Sim

Knihovna (balík) v jazyce Java, která stejně jako C-Sim využívá filosofické základy Simuly, ale tentokrát použitelné v Javě.
Pro implementaci pseudoparalelních procesů využívá vlákna (Thread).

Principy použití

--- Doufám, že stačí příklad v J-Simu. Sem se nevěděl co napsat. Když bude mít někdo nějaký nápad doplním (pozn. Michal Jašpr)

```
class SimModel {
      JSimSimulation    simulation = null;
      JSimHead    fronta = null;
      JSimLink    pozadavek = null;
      Proces    proces = null;
      ...
```

```
      //vytvorim simulaci
      simulation = new JsimSimulation("...");
      //vytvorim frontu
      fronta = new JsimHead("...", simulation);
      //vytvorim pozadavek
      pozadavek = new JSimLink();
      ...
      //vytvorim proces
      proces = new Proces(...);
      ...
      //aktivuji proces
      proces.activateNow();
      while (true) {
           ...
      }
      ...
}

class Proces extends JSimProcess
{
    ...
    //inicializace
    public Proces(...) ...
    {
        ...
    }

    //telo procesu
    protected void life()
    {
         ...
    }
}
```

5 Spolehlivost číslicových systémů, obnovované a neobnovované systémy, ukazatele spolehlivosti pro prvky systému, spolehlivostní modely (určení spolehlivostních ukazatelů systému ze známých ukazatelů jeho prvků)

Základní úlohy, kterými se teorie spolehlivosti zabývá:

- měření spolehlivosti
- předvídáním spolehlivosti
- zlepšováním spolehlivosti

Spolehlivost – obecná vlastnost objektu spočívající ve schopnosti plnit požadované funkce při zachování hodnot stanovených provozních ukazatelů v daných mezích a čase podle stanovených technických podmínek.

Rozlišujeme dva stavy objektu:

- poruchový (tj. stav kdy porucha nastala)
- bezporuchový (tj. kdy porucha nenastala)

Za *stálou* je porucha považována, pokud po výskytu poruchy setrvá v poruchovém stavu až do okamžiku opravení nebo vyřazení systému z provozu. Dochází-li k poruše nahodile (neočekávaně se objevuje a znovu mizí), označujeme ji za *nestálou* nebo *občasnou*.

Dalším možným dělením je podle toho, zda provádíme obnovu do bezporuchového stavu nebo nikoliv. Rozlišujeme tedy systémy:

- *obnovované* – obnova je chápána jako přechod z poruchového do bezporuchového stavu. Činnost k tomu vedoucí je označována jako oprava.
- *neobnovované* – Jako neobnovovaný může být označen objekt, který je nepřístupný (např. sondy ve vesmíru) či u kterého je závada neopravitelná nebo je jeho opravení nerentabilní.

5.1 Ukazatele spolehlivosti

Veličiny používané pro hodnocení spolehlivosti mají náhodný charakter, proto se při práci s nimi využívá počet pravděpodobnosti. Při určování hodnot ukazatelů spolehlivosti se pak využívají metody matematické statistiky.

5.1.1 Ukazatele spolehlivosti neobnovovaných objektů

V teorii spolehlivosti je základní sledovanou náhodnou veličinou velikost časového intervalu od uvedení do provozu do poruchy objektu. Je-li t čas měřený od uvedení do provozu, má distribuční funkce této náhodné veličiny význam *pravděpodobnosti poruchy* objektu do času t a značí se Q(t). Opakem k výše zmíněné pravděpodobnosti poruchy Q(t) je *pravděpodobnost bezporuchového stavu*. Značí se R(t) a určí se jako $R(t) = 1 - Q(t)$.

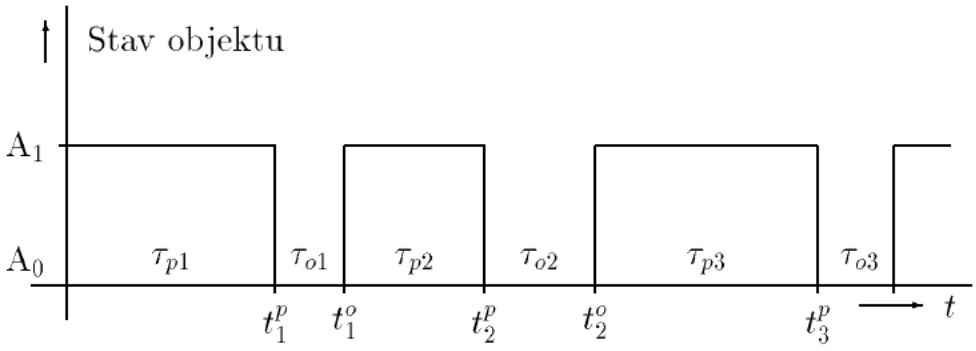
Další charakteristikou je *intenzita pravděpodobnosti náhodné veličiny* označovaná jako λ(t), která se určí ze vztahu $\lambda(t)=\frac{f(t)}{R(t)}=\frac{f(t)}{1-Q(t)}$, kde f(t) je hustota pravděpodobnosti náhodné veličiny t definované jako $f(t)=\frac{dQ(t)}{dt}$.

λ(t) je označována také jako *intenzita poruch* a udává podmíněnou hustotu poruch v čase t za předpokladu, že k poruše dosud nedošlo.

Střední doba bezporuchového provozu T_S, je střední hodnota provozní doby objektu, během níž nenastala žádná porucha. Určí se jako $T_S=\int_0^\infty R(t)dt$. Často se označuje jako MTTF tedy jako střední doba do první poruchy.

5.1.2 Ukazatele spolehlivosti obnovovaných objektů

Obnovovaný objekt prochází během života několika stavy:



- t_i^p - okamžiky, ve kterých nastala porucha
- t_i^o - okamžiky, kdy byla uskutečněna obnova bezporuchového stavu
- A_0 – poruchový stav
- A_1 – bezporuchový stav
- τ_{pi} - délka trvání jednoho bezporuchového úseku
- τ_{oi} - doba trvání jedné opravy

Pro obnovované objekty se místo střední doby do poruchy používá *střední doba mezi poruchami*. Stanoví se jako aritmetický průměr všech naměřených dob bezporuchového provozu od skončení opravy do výskytu následující poruchy. Tuto hodnotu získáme tak, že kumulativní dobu provozu t_p (součet dob provozu) vydělíme počtem výpadků díky poruchám.

Platí: $T_S=\frac{t_p}{n}=\frac{1}{n}\sum_{i=1}^{n}\tau_{pi}$

MTBF (mean time between failures) se počítá jako střední doba od jednoho výskytu poruchy do dalšího výskytu poruchy. Je tedy do něj zahrnuta i doba opravy. Někdy se též tato doba značí jako střední doba cyklu tedy T_c.

Okamžitý součinitel pohotovosti $K_p(t)$ udává pravděpodobnost, že v čase t bude systém v provozuschopném stavu. Zpravidla existuje limita $K_p = \lim_{t\to\infty} K_p(t)$ označovaná jako stacionární součinitel pohotovosti. Ten udává pravděpodobnost, že systém, který je v ustáleném provozním režimu, bude provozuschopný v libovolném zvoleném okamžiku. Je možné to interpretovat jako poměrnou část provozuschopné doby (t_p) z celkové sledované doby - $K_p = \frac{t_p}{t_p + t_0}$.

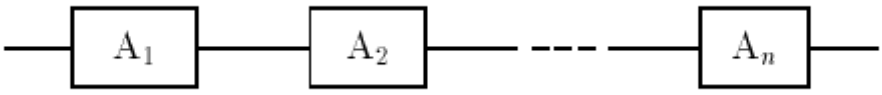
Střední doba opravy se určuje jako $T_o = \frac{t_o}{n}$ a označuje se též jako MTTR.

5.2 Modely systémů z nezávislých prvků

U mnoha systémů lze předpokládat nezávislost poruch případně i jednotlivých prvků. V takovém případě jsou doby do poruchy u jednotlivých prvků nezávislé náhodné veličiny. Spolehlivostní modely systémů s nezávislými prvky jsou relativně jednoduché, a proto v případě kdy máme možnost volby, jim dáváme přednost. Po matematické stránce jsou tyto modely založeny na vztazích pro násobení pravděpodobností nezávislých náhodných jevů (pravděpodobností bezporuchového provozu R(t) a poruch Q(t) jednotlivých prvků) a pro sčítání pravděpodobností vzájemně se vylučujících jevů (tj. možných stavů systému).

5.2.1 Sériový model

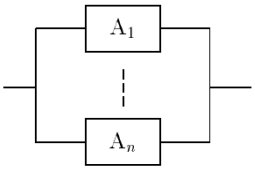
Použijeme jej, jestliže porucha libovolného systému způsobí poruchu celku a časové intervaly do poruchy jednotlivých prvků jsou navzájem nezávislé náhodné veličiny.



Výsledná intenzita bezporuchového provozu R(t) je dána součinem pravděpodobností bezporuchového provozu $R_i(t)$ pro jednotlivé prvky - $R(t) = \prod_{t=1}^{n} R_i(t)$. Pro konstantní intenzity poruch λ_i pak dostaneme $R(t) = \prod_{t=1}^{n} e^{\lambda_i t} = e^{\lambda t}$ díky tomu, že výsledná intenzita poruch systému je dána jako součet jednotlivých intenzit.

5.2.2 Paralelní model

Používá se, jestliže porucha systému nastane pouze při poruše všech jeho prvků.

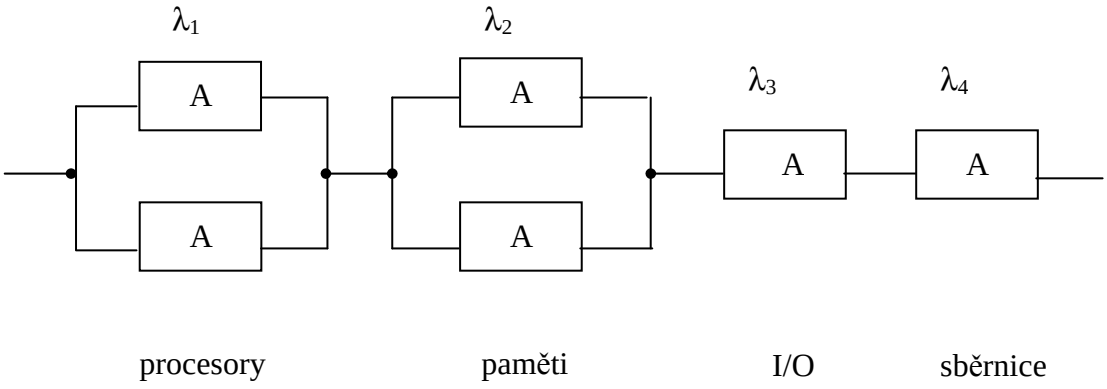


Jestliže známe pravděpodobnost poruchy $Q_i(t)$ pro každý prvek A_i a jsou-li poruchy prvků nezávislé, můžeme výslednou pravděpodobnost poruchy Q(t) vyjádřit vztahem

$Q(t) = \prod_{i=1}^{n} Q_i(t)$. Pravděpodobnost bezporuchového provozu je pak vyjádřena jako $R(t) = 1 - \prod_{i=1}^{n} (1 - R_i(t))$.

5.2.3 Sérioparalelní model

Jen ukázka



procesory paměti I/O sběrnice

Paralelní zapojení procesorů

$$Q_{par1} = Q_1 \cdot Q_1 = (1 - R_1) \cdot (1 - R_1) = 1 - 2R_1 + R_1^2$$

$$R_{par1} = 1 - Q_{par1} = 2R_1 - R_1^2$$

Paralelní zapojení pamětí

$$Q_{par2} = Q_2 \cdot Q_2 = (1 - R_2) \cdot (1 - R_2) = 1 - 2R_2 + R_2^2$$

$$R_{par2} = 1 - Q_{par2} = 2R_2 - R_2^2$$

Výpočet nad celým systémem

$$R = R_{par1} \cdot R_{par2} \cdot R_3 \cdot R_4 = \left(2R_1 - R_1^2\right) \cdot \left(2R_2 - R_2^2\right) \cdot R_3 \cdot R_4$$

$$R = 4R_1R_2R_3R_4 - 2R_1^2R_2R_3R_4 - 2R_1R_2^2R_3R_4 + R_1^2R_2^2R_3R_4$$

$$R(t) = 4e^{-(\lambda_1+\lambda_2+\lambda_3+\lambda_4)t} - 2e^{-(2\lambda_1+\lambda_2+\lambda_3+\lambda_4)t} - 2e^{-(\lambda_1+2\lambda_2+\lambda_3+\lambda_4)t} + e^{-(2\lambda_1+2\lambda_2+\lambda_3+\lambda_4)t}$$

6 Principy zajištění odolnosti výpočetních a informačních systémů proti poruchám

Odolnosti proti poruchám lze dosáhnout pouze použitím nadbytečných prostředků, souhrnně nazývaných záloha, nebo též redundance, i když jako záloha se označuje pouze redundance, která byla vytvořena záměrně. Záložní (redundantní) jsou prostředky, jejichž použití by bylo zbytečné, kdyby všechny ostatní části systému pracovali správně.

Prostředky používané při realizaci zálohy zahrnují následující zdroje:
- technické vybavení (hardware)
- programové vybavení (software)
- informace
- čas

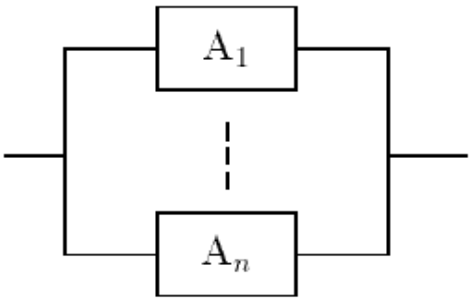
6.1 Redundantní hardware

Jedná se o nejpoužívanější formu zálohování. Do této kategorie patří např. záložní součástky, spoje, obvody a celé funkční bloky.
Pro nadbytečné technické vybavení je charakteristický růst nákladů, rozměrů systému, hmotnosti a často i spotřeby energie.
Používá se především v bezpečnostně kritických systémech (v nemocnicích, letadlech, sondách), v běžných PC může být příkladem zapojení disků do RAID.

Rozeznáváme dva druhy záloh:
- horkou
- studenou

6.1.1 Horká záloha (též statická)

Prvek systému (součástka, zařízení) je doplněn dalšími stejnými prvky, které jsou k sobě zapojena paralelně a všechny běží. Dojde-li k výpadku na jednom z prvků, automaticky jeho funkci zastoupí další. K poruše celého systému dochází až v momentě, kdy dojde k poruše všech zařízení. Pravděpodobnost poruchy takovéto zálohy lze určit pomocí paralelního spolehlivostního modelu:



Paralelní spolehlivostní model

Jestliže známe pravděpodobnost poruchy $Q_i(t)$ pro každý prvek A_i a jsou-li poruchy prvků nezávislé, můžeme výslednou pravděpodobnost poruchy Q(t) vyjádřit vztahem:

$$Q(t) = \prod_{i=1}^{n} Q_i(t)$$

Pro pravděpodobnost bezporuchového provozu lze vztah upravit do tvaru:

$$R(t) = 1 - \prod_{i=1}^{n} (1 - R_i(t))$$

Použijeme-li n shodných prvků s konstantní intenzitou poruch, je možné vyjádřit střední dobu bezporuchového provozu jako:

$$T_S = \frac{1}{\lambda_p} \sum_{i=1}^{n} \frac{1}{i}$$

Nevýhodou horké zálohy je velká spotřeba energie (všechny prvky musí být trvale připojeny na napájení) a malá hodnota střední doby bezporuchového provozu, protože všechny prvky jsou trvale zatíženy a opotřebovávají se stejně rychle.

6.1.2 Studená záloha (též dynamická)

Studenou zálohou rozumíme fakt, že záložní prvky jsou z počátku vypnuté (nezatížené – jejich intenzita poruch je nízká) a zapínají se až po poruše dosud aktivního prvku.

Dosahuje se větší střední hodnoty bezporuchového provozu a menší spotřeby energie, protože zálohu lze částečně, nebo úplně odpojit od napájení.

Hlavní nevýhodou je nebezpečí, že během přepínání ze základního na záložní prvek dojde k dočasnému výpadku signálu na výstupu systému.

6.2 Redundantní software

Do nadbytečného programového vybavení spadají především diagnostické programy, provádějící detekci a lokalizaci poruch, dále programy řídící zotavení po poruše (zařazování bezchybných datových bloků místo chybných).

Pomocí kontrolních, nebo několikanásobných výpočtů lze zjistit, nebo i opravit chyby vznikající v systému.

Tento přístup je spojen s využitím velkého množství nadbytečného času.

6.3 Redundantní informace

Využívají se především v bezpečnostních kódech k průběžné detekci chyb.