



SEMESTRÁLNÍ PRÁCE Z KIV/VSP

VÝKONNOST A SPOLEHLIVOST
ČÍSLICOVÝCH SYSTÉMŮ

Standartní zadání – otevřená síť front

Michel Soběhart – A14N0144P

sobeharm@students.zcu.cz

Datum narození: 4.8.1992

9. prosince 2015

Obsah

1	Zadání	1
1.1	Standartní zadání semestrální práce	1
1.2	Formát odevzdaného programu	2
2	Otevřená síť front	3
2.1	Popis modelu sítě front	4
2.2	Zvolené hodnoty	4
3	Teoretický úvod do problematiky	4
3.1	Systémy hromadné obsluhy	4
3.2	Elementární SHO	5
3.2.1	Zdroj požadavků	5
3.2.2	Vstupní proud požadavků	6
3.2.3	Fronta požadavků	6
3.2.4	Kanál obsluhy	6
3.3	Otevřené sítě front	7
3.4	Generátory náhodných hodnot	7
3.4.1	Exponenciální rozdělení	7
3.4.2	Normální (Gaussovo) rozdělení	8
4	Teoretické výpočty	10
4.1	Statistiky uzlů systému	10
4.1.1	Střední doba frekvence uzlů	10
4.1.2	Podmínka stacionarity systému a zatížení uzlů	11
4.1.3	Střední počet požadavků v uzlech	12
4.1.4	Střední doba odezvy uzlu	13
4.2	Střední délka front	14
4.2.1	Střední doba mezi požadavky	14
4.3	Statistika celého systému	15
4.3.1	Střední počet požadavků celého systému	15
4.3.2	Střední doba odezvy celého systému	15
4.4	Statistika měřeného místa	16
5	Implementace	17
5.1	Balík distribution	18
5.1.1	Rozhraní IDistribution	18
5.1.2	Třída ExponentiaFromJSIM	18
5.1.3	Třída GaussianByExponential	18

5.1.4	Třída ExponentialDistribution	18
5.1.5	Třída GaussianDistribution	19
5.1.6	Třída UniformDistribution	19
5.2	Balík generator	19
5.2.1	Třída NetworkGenerator	19
5.2.2	Třída PhotoRequestGenerator	19
5.3	Balík main	19
5.3.1	Třída Main	20
5.4	Balík process	20
5.4.1	Třída Crossing	20
5.4.2	Třída Follower	20
5.4.3	Rozhraní IProcessInput	20
5.4.4	Třída Output	20
5.4.5	Třída PhotoProcess	21
5.4.6	Třída PhotoRequest	21
5.4.7	Třída PhotoCounter	21
5.5	Balík statistics	21
5.5.1	Rozhraní IStatistics	21
5.5.2	Třída Statistics	22
5.5.3	Třída FlowStatistics	22
5.6	Spuštění aplikace	22
6	Výsledky a porovnání	23
6.1	Výsledky pro uzly sítě	23
6.1.1	Uzel Podání žádosti na internetu	23
6.1.2	Uzel Podání žádosti na pobočce	24
6.1.3	Uzel Zpracování objednávek	24
6.1.4	Uzel Výroba fotografie	25
6.2	Celkové výsledky v síti	26
6.3	Statistiky měřeného místa - tok před uzlem Zpracování fo- tografií	26
7	Závěr	30

1 Zadání

1.1 Standartní zadání semestrální práce

1. Vymyslete si otevřenou síť front (tj. propojení, vstupní proudy) obsahující alespoň 4 obslužné uzly (jeden kanál, fronta FIFO, neomez. délka), alespoň 2 vstupní proudy požadavků, alespoň 2 vnitřní zpětné vazby.
2. Parametry sítě (tj. střední frekvence vstupních proudů, střední doby obsluhy v jednotlivých kanálech a p-ti větvení) zvolte tak, aby síť pracovala ve stacionárním režimu. Doporučená hodnota zatížení pro všechny uzly: $\rho > 0.5$.
3. Určete výpočtem střední frekvence toků v uzlech. Dále určete veličiny L_q a T_q pro jednotlivé uzly a L_q a T_q pro celou síť pro případ, že všechny vstupní toky jsou Poissonovské a doby obsluhy ve všech uzlech mají exponenciální rozdělení.
4. Vypočtené hodnoty ověřte vlastnoručně vytvořeným simulačním programem. Použijte simulační knihovnu C-Sim nebo J-Sim.
5. Dále uvažujte případ, kdy všechny náhodné časové intervaly v modelu (příchody, obsluhy) mají Gaussovské pravděpodobnostní rozdělení $N(a, \sigma)$ s (různou) střední hodnotou zvolenou v bodě 2. Vytvořte generátor tohoto rozdělení jako funkci v jazyce C nebo Java (s parametry např. a, σ) a testováním ověřte správnou funkci generátoru - chce se tedy po Vás vytvoření a prokázání správné funkce generátoru, který napíšete VY - využít můžete pouze knihovní funkce pro generování rovnoměrného rozdělení (jako v průběžném příkladu č. 1). Použití knihovní funkce pro Gaussovo rozdělení z J-SIMu se nepočítá!
6. Simulací ověřte chování sítě (tj. určete stejné veličiny jako v bodech 3) a 4) pro případ, že všechna rozdělení (příchody, obsluhy) budou mít hustotu $N(a, \sigma)$ se stejnou střední hodnotou jako pro exponenciální rozdělení alespoň pro 3 různé hodnoty koeficientu variace $C = \sigma/a$. Pokuste se o poměrně odlišné koeficienty, at' je vidět rozdíl v chování systému (např. 0.05 - tj. skoro konstantní generátor, 0.2 a 0.7 - ale to je pouze příklad). Poznámka: Simulační program je stejný jako v bodě 4, ale volá se jiný generátor podle bodu 5.

7. Simulační program upravte pro sledování dalších individuálně zadaných výkonnostních charakteristik sítě.
8. Řešení zpracujte formou písemného referátu (cca 10 stran, grafy, tabulky, barevné obrázky, hudební vložky, multimédia ap. - berte to jako přípravu na diplomku, navíc vlastní referát lze využít při zkoušce).

Dokumentace bude mimo jiné obsahovat všechny body, které měla obsahovat průběžná semestrální práce číslo 4 - tedy detailní analýzu, dále pak popis použitého algoritmu generování náh. čísel, výsledky simulací pro různé vstupní parametry a jejich srovnání s teoretickými výpočty, závěr, atd..

1.2 Formát odevzdaného programu

Formát a podmínky na odevzdaný program jsou podobné jako u průběžné semestrální práce číslo 4:

- Hlavní program bude možné spustit s následujícím formátem parametrů (argumentů programu): `program (počet požadavků) (EXP / GAUSS)`
Počet požadavků udává, kolik požadavků se musí nechat simulací projít, než bude ukončena a vypočítány výsledky.
Druhým parametrem bude jeden z řetězců "EXP" nebo "GAUSS" (na velikosti písmen necht' nezáleží). V případě EXP bude spuštěna simulace pro exponenciální rozdělení, v případě GAUSS budou postupně spuštěny alespoň tři simulace pro normální rozdělení s různými koeficienty variace - tyto koeficienty budou uvedeny ve výstupu programu.
- Při spuštění bez parametrů se program bude chovat, jako by byl postupně spuštěn nejdříve s parametrem EXP a poté GAUSS, počet požadavků použije nějaký defaultní.
- Program (respektive rozbalený ZIP archiv) musí být dodán ve stavu schopném okamžitého spuštění na jednom z těchto systémů: WindowsXP, Linux na i586. Dle charakteru operačního systému bude připraven script "run.bat" nebo "run.sh", který zajistí spuštění programu a předání parametrů (já budu tedy při testování spouštět tento script). V případě

Javovských programů předpokládejte, že jsou nastavené cesty na nainstalovanou Javu 1.6 a že nejsou nastavené cesty na žádné případné knihovny, které nejsou součástí distribuce J2SE!! Pokud budete nějaké potřebovat, přidejte si je např. do vašeho .jar souboru - to se týká i knihovny jsim.jar!

- Pro odstranění logování se inspiřujte souborem logging.properties, který najdete na stránce Studijní materiály.
- Zkušenost z minulého roku: Mnoho lidí nechávalo simulací "projít" málo požadavků, rekordman jich měl snad jen 100 a divil se, proč to oproti teorii nevychází. V dnešní době je bez problémů program ladit na řádově desítkách tisíc požadavků a nakonec to spustit třeba pro stovky tisíc - pro srovnání výstupu s teorií.
- Výstup programu: Držte se zadání a v první řadě musí program vypočítat základní charakteristiky obslužných uzlů (tj. střední frekvence toků v uzlech, zatížení uzlů, T_q , L_q pro každý uzel) a T_q a L_q pro celou síť. Dále individuálně zadané charakteristiky. Tyto výstupy musí být ve výstupu programu jasně k nalezení.
- Pro získání lepšího hodnocení můžete každou (či jen nějakou) veličinu počítat ještě "lépe", tedy určovat její rozptyl či dokonce histogram (jak to dělá ukázkový program QNAnalyzer).

Poznámka: Napíše-li přednášející k Vašemu zadání práce, že chce, abyste sledovali STATISTIKY té a té veličiny, znamená to, že budete ze simulace zjišťovat střední hodnotu, rozptyl a charakter rozdělení (histogram) této veličiny! Jaké jsou možnosti a způsoby sledování se dozvíte na přednášce.

2 Otevřená síť front

Zadáním mé semestrální práce je otevřená síť front, která představuje zjednodušeně celý proces výroby fotografií od podání objednávky až po výrobu.

2.1 Popis modelu sítě front

Můj model se skládá ze čtyř uzlů (procesů) tvorby fotografií. Prvním uzlem je podání fotek přes internet, druhým uzlem podání fotek na pobočce, třetím je zpracování fotografií a čtvrtým samotná výroba. Model je popsán v oficiálním zadání, které je dodané jako příloha k tomuto dokumentu.

2.2 Zvolené hodnoty

Hodnoty jsem volil podle toho, aby zatížení bylo vyšší než 0.5 a mělo tedy smysl počítat ostatní veličiny. Zvolené hodnoty parametrů pro exponenciální rozdělení generátorů jsou tedy (1).

$$\lambda_1 = 10, \lambda_2 = 5 \quad (1)$$

Parametry exponenciálního rozdělení pro doby zpracování jednotlivých uzlů systému jsou (2).

$$\mu_1 = 15, \mu_2 = 10, \mu_3 = 20, \mu_4 = 20 \quad (2)$$

Pravděpodobnosti, kdy se požadavky rozdělují jsou (3). Pravděpodobnost vždy určuje průchod požadavku do dalšího uzlu a k ní inverzní pravděpodobnost určuje návrat požadavku do jednoho z předchozích uzlů.

$$p_1 = 0.8, p_2 = 0.9, p_3 = 0.85, p_4 = 0.95 \quad (3)$$

3 Teoretický úvod do problematiky

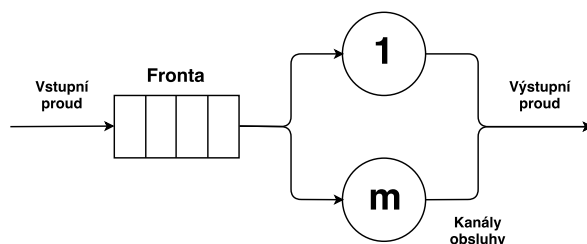
3.1 Systémy hromadné obsluhy

Jako *systémy hromadné obsluhy* (dále jen SHO) jsou označovány více či méně formální modely, jejichž funkce spočívá v poskytnutí nějaké *služby* (realizace

obsluhy) pro velké počty souběžně přicházejících *požadavků* (také *transakcí*). Služby poskytují prvky zvané *kanály obsluhy* či *servery*, před kanálem obsluhy se obvykle vytváří fronta požadavků. Významnou vlastností je abstrakce od sémantiky služby, modeluje se pouze časová posloupnost přicházejících požadavků a jejich časová náročnost na služby. Vzhledem k velkým počtům požadavků je použit přiměřený pravděpodobnostní popis časových charakteristik proudů požadavků.

3.2 Elementární SHO

Konceptuální model elementárního obslužného systému je naznačena na následujícím obrázku 1. Jako stav systému zde figuruje celkový počet požadavků, které jsou akumulované v systému (ve frontě i v obslužných kanálech).



Obrázek 1: Elementární SHO

Dále popíšeme jednotlivé prvky oslužného systému. Základním předpokladem pro další popis a výpočty je *stacionární režim činnosti systému*. To lze jinak popsat jako časová stálost zdrojových statistických charakteristik modelovaného systému.

3.2.1 Zdroj požadavků

Zdroj požadavků může být *omezený* nebo *neomezený*. My se bude zabývat omezeným zdrojem, protože simulací projde určitý počet požadavků. Vstupní proud požadavků je tedy závislý na stavu obslužného systému.

3.2.2 Vstupní proud požadavků

Předpokládejme, že požadavky vstupují do SHO náhodně, časově za sebou. Náhodná veličina $\tau_k = t_k - t_{k-1}$, kde t_k a t_{k-1} představují dva časové údaje o příchodu prvků do systému a hodnota indexu $k \geq 1$. τ_k se nazývá *interval příchodů*.

Nejčastěji užívaným způsobem matematického popisu vstupního proudu je zadání distribuční funkce pravděpodobnostního rozdělení $F_a(t) = P(\tau \leq t)$ nebo odpovídající hustota pravděpodobnosti $f_a(t)$. Typem proudu, které se nejčastěji využívá je tzv. *poissonovský vstupní proud*, ve kterém má interval příchodů *exponenciální pravděpodobnostní rozdělení*. Tato informace se dále bude využívat jako podmínka při výpočtu u některých veličin.

3.2.3 Fronta požadavků

Fronta požadavků je charakterizována maximálním počtem přijatých požadavků ve frontě, tedy *maximální délkou* a *frontovou disciplínou*. Frontová disciplína je pravidlo, jakým se vybírá požadavek z fronty. Já využívám v mé síti front FIFO (First-In-First-Out), ale existují i jiné, jako například LIFO, prioritní FIFO. Fronta FIFO bere požadavky z fronty podle času, kdy do dané fronty přišly.

3.2.4 Kanál obsluhy

Nejčastějším případem je jednokanálová obsluha, kdy m je počet kanálů obsluhy, tedy 1. Doba obsluhy t_s konkrétního požadavku je obvykle chápána jako náhodná veličina, která má dané pravděpodobnostní rozdělení.

Pro **zatížení kanálu** je nutná *podmínka stacionarity*. Podmínka je dána vztahem $\rho < 1$ a vypočítá se pomocí vzorce (4).

$$\rho = \frac{1}{m} \cdot \frac{\lambda}{\mu} \quad (4)$$

3.3 Otevřené sítě front

Reálné obslužné systémy, které jsou komplikovanější (nemají pouze jeden kanál obsluhy), kdy každý kanál obsluhy má vlastní frontu požadavků, které bude obsluhovat. Požadavky vstupují do systému, zde přechází mezi kanály obsluhy a po ukončení všech dílčích obsluh vystupuje ze systému. Takový systém lze modelovat jako síť, kdy prvky jsou lementární SHO. *Otevřené* sítě znamená, že mají explicitní vstupy z okolí do systému.

Otevřenou síť front tedy můžeme znázornit jako orientovaný graf, kde uzly odpovídají elementárním systémům hromadné obsluhy. Hrany popisují možnosti přechodu požadavků mezi jednotlivými uzly. Váhy hran se obvykle využívají tzv. *pravděpodobnostní větvení*. Hrana mezi uzly i a uzlem j je dána pravděpodobností p_{ij} přechodu do uzlu j po ukončení obsluhy uzlu i . Součet všech hran, které vycházejí z uzlu, je roven 1. Okolí obslužného systému lze také vyjádřit jako zvláštní uzel, kde hrana modeluje vstup požadavků z okolí do uzlu v systému.

Teoretický úvod byl psán s použitím podkladů [?].

3.4 Generátory náhodných hodnot

Generátory náhodných hodnot máme v semestrální práci dva typy, kterými se budeme zabývat. Prvním typem je exponenciální rozdělení a druhým normální (Gaussovo) rozdělení. K výpočtu potřebujeme také rozdělení uniformní.

3.4.1 Exponenciální rozdělení

Exponenciální rozdělení s parametrem $\lambda > 0$ je spojité rozdělení na množině kladných čísel s hustotou (5)

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & \text{pro } x > 0 \\ 0, & \text{pro } x \leq 0 \end{cases} . \quad (5)$$

Výpočtem integrálu hustoty pravděpodobnosti získáme pro kladné x distri-

buční funkci (6)

$$F(x) = \int_0^x \lambda e^{-\lambda x} dx = 1 - e^{-\lambda x}, \quad (6)$$

z toho plyne, že distribuční funkce exponenciálního rozdělení má tvar (7)

$$F(x) = \begin{cases} 1 - e^{-\lambda x}, & \text{pro } x > 0 \\ 0, & \text{pro } x \leq 0 \end{cases}. \quad (7)$$

Abychom získali vzorec, které při generování náhodných čísel bude vykazovat exponenciální rozložení, musíme distribuční funkci invertovat. Inverzní funkce bude tedy mít tvar (8)

$$F^{-1}(x) = -\frac{\ln(1-y)}{\lambda}, \quad (8)$$

kde y je vygenerované číslo.

Nakonec ještě přidám výpočet vzorců pro střední hodnotu (9) a rozptyl (10), které vycházejí z integrace hustoty, které platí tedy pouze na intervalu $(0, +\infty)$

$$E(X) = \frac{1}{\lambda}, \quad (9)$$

$$D(X) = \frac{1}{\lambda^2}, \quad (10)$$

kde $X \sim Exp(\lambda)$.

3.4.2 Normální (Gaussovo) rozdělení

Normální rozdělení je dáno dvěma parametry, parametr a , který značí střední hodnota a parametr σ je rozptyl veličiny. Při odvozování vzorce na výpočet

hodnot generátoru se využívá tzv. *centrální limitní věty teorie pravděpodobnosti*.

Tato věta nám říká, že součet náhodných čísel s libovolným rozdělením má asymptoticky (blíží se narůstajícím počtem prvků součtu) normální rozdělení se střední hodnotou a rozptylem danými součtem středních hodnot a rozptylů pravděpodobnostních rozdělení prvků součtu.

Pokud tedy chceme vypočítat vzorec s využitím normalizovaného rovnoměrného rozdělení, je střední hodnota $E[y] = 0.5$ a rozptyl roven $D[y] = \frac{1}{12}$. Součtem dvanácti konkrétních hodnot je tedy střední hodnota rovna 6 a rozptyl roven 1. Vzorec, který lze využít pro generování náhodných čísel tedy realizujeme pouze posunem o zadaná hodnoty běžného normálního rovnoměrného rozdělení s hodnotami $N(0, 1)$ na rozdělení $N(0, 6)$. Vzorec tedy bude (11).

$$x = \sigma\left(\left(\sum_{j=1}^1 2y_j\right) - 6\right) + a \quad (11)$$

Pokud bych chtěl generovat hodnoty pomocí exponenciálního rozdělení s parametrem λ , lze využít obdobného vzorce pro výpočet (12). Zvolil jsem si zde 15 hodnot, které budu sčítat a parametr $\lambda = 3$.

$$\begin{aligned} x &= \sigma\left(\left(\sum_{j=1}^1 5y_j\right) - \left(15 \cdot \left(\frac{1}{\lambda} + \frac{1}{\lambda^2}\right)\right)\right) + a \\ x &= \sigma\left(\left(\sum_{j=1}^1 5y_j\right) - \left(15 \cdot \left(\frac{1}{3} + \frac{1}{9}\right)\right)\right) + a \end{aligned} \quad (12)$$

4 Teoretické výpočty

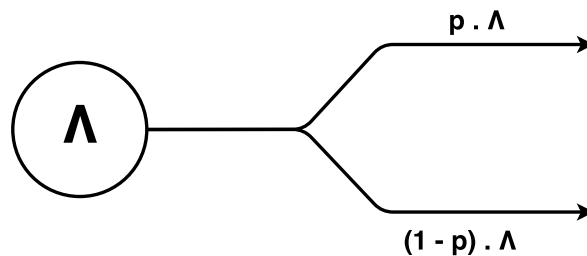
4.1 Statistiky uzlů systému

4.1.1 Střední doba frekvence uzlů

Střední doba frekvence vnitřního toku si označím Λ_{UZEL} . Nejdříve si určím frekvenci celkového toku, vystupujícího z okolí do sítě front (frekvence generátorů náhodných veličin). To je určeno středními hodnotami generátorů, hodnoty se tedy rovnají (13).

$$\begin{aligned}\Lambda_A &= \lambda_1 = 10 \\ \Lambda_B &= \lambda_2 = 5\end{aligned}\tag{13}$$

Pokud chci tedy vypočítat střední dobu frekvence v jednom uzlu, musím sečíst všechny frekvence toku hran, které do uzlu vstupují. Frekvenci toku hrany se vypočítá jako násobek pravděpodobnosti toku požadavku po dané hraně se střední frekvencí toku v daném uzlu. Máme-li tedy větvení požadavku podle pravděpodobnosti p z uzlu, kde střední doba frekvence toku je Λ , toky hran budou vypadat dle obrázku 2.



Obrázek 2: Výpočet frekvence toku hrany

Vzorec pro výpočet střední doby frekvence toku uzlu bude (14).

$$\Lambda_i = \sum_j p_{ji} \cdot \Lambda_j\tag{14}$$

kde index j nabývá hodnot, které odpovídají vstupním hranám do zadaného uzlu i .

Pro můj model tedy vytvořím rovnice pomocí vzorce (14) a dostanu jednotlivé rovnice pro sledovanou statistiku (15).

$$\begin{aligned}
 \Lambda_1 &= \Lambda_A + (1 - p_1) \cdot \Lambda_1 + (1 - p_4) \cdot \Lambda_4 \\
 \Lambda_2 &= \Lambda_B + (1 - p_2) \cdot \Lambda_2 + (1 - p_4) \cdot \Lambda_4 \\
 \Lambda_3 &= p_1 \cdot \Lambda_1 + p_2 \cdot \Lambda_2 \\
 \Lambda_4 &= p_3 \cdot \Lambda_3
 \end{aligned} \tag{15}$$

Do rovnice doplním hodnoty známých proměnných a počítám pomocí řešení soustav rovnic (16).

$$\begin{aligned}
 \Lambda_1 &= \Lambda_A + 0.2 \cdot \Lambda_1 + 0.05 \cdot \Lambda_4 \\
 \Lambda_2 &= \Lambda_B + 0.1 \cdot \Lambda_2 + 0.15 \cdot \Lambda_3 \\
 \Lambda_3 &= 0.8 \cdot \Lambda_1 + 0.9 \cdot \Lambda_2 \\
 \Lambda_4 &= 0.85 \cdot \Lambda_3
 \end{aligned} \tag{16}$$

Po vypočítání soustavy rovnic dostávám hodnoty pro střední frekvenci všech uzlů (17).

$$\begin{aligned}
 \Lambda_1 &\doteq 13,4868421 \\
 \Lambda_2 &\doteq 8,65153079 \\
 \Lambda_3 &\doteq 18,57585139 \\
 \Lambda_4 &\doteq 15,78947368
 \end{aligned} \tag{17}$$

4.1.2 Podmínka stacionarity systému a zatížení uzlů

Podmínka stacionarity systému je taková, že zatížení uzlů musí být nižší než 1 ($\rho < 1$). Podmínka úzce souvisí se zatížením jednotlivých uzlů. Pokud by v jakémkoliv uzlu byla vyšší hodnota střední frekvence toků než střední doba obsluhy požadavku, nejspíše by tošlo k zahlcení systému na daném uzlu. Pro výpočet zatížení uzlu musíme určit střední dobu obsluhy požadavku pomocí vzorce (18).

$$T_{s_i} = \frac{1}{\mu_i} \quad (18)$$

Zatížení uzlu systému tedy vypočítáme pomocí vzorce (19).

$$\rho_i = \Lambda_i \cdot T_{s_i} \Rightarrow \rho_i = \frac{\Lambda_i}{\mu_i} \quad (19)$$

kde Λ je střední hodnota frekvence toku v uzlu a μ je střední doba obsluhy požadavku. Nyní tedy můžeme vypočítat zatížení všech uzlů v systému a výsledné hodnoty jsou (20).

$$\begin{aligned} \rho_1 &= \frac{\Lambda_1}{\mu_1} = \frac{13,4868421}{15} \doteq 0,8991228 \\ \rho_2 &= \frac{\Lambda_2}{\mu_2} = \frac{8,65153079}{10} \doteq 0,865153079 \\ \rho_3 &= \frac{\Lambda_3}{\mu_3} = \frac{18,57585139}{20} \doteq 0,92879257 \\ \rho_4 &= \frac{\Lambda_4}{\mu_4} = \frac{15,78947368}{20} \doteq 0,78947368 \end{aligned} \quad (20)$$

Z předchozích vzorců (20) můžeme říci, že náš systém je stacionární, protože všechny hodnoty ρ_i pro každý uzel jsou menší než 1, tudíž je splněna podmínka stacionarity.

4.1.3 Střední počet požadavků v uzlech

Střední počet požadavků v uzlu je roven střední délce fronty uzlu. Abych mohl určit jednoduchým způsobem tyto hodnoty, potřebuji si představit uzel jako elementární systém hromadné obsluhy. K tomu využiji tzv. Jacksonův teorém, který určuje podmínky, při kterém si dokážu uzel představit jako elementární SHO. Podmínky tedy jsou:

1. Všechny vstupní toky systému mají poissonovský charakter rozdělení
2. Všechny obslužné uzly mají exponenciální rozdělení doby obsluhy a střední hodnotu doby obsluhy T_s

3. Pokud se ukončí obsluha požadavku v uzlu, přejde požadavek s určitou pravděpodobností p_{ij} do uzlu j bez jakéhokoliv zpoždění

V našem systému je splnění podmínek jasné. Podmínka poissonovského charakteru rozdělení vstupních toků je zřejmá ze zadání v kapitole 1.1. Podmínku exponenciálního rozdělení doby obsluhy uzlů a střední hodnotu obsluhy systém splňuje, důkazem mohou být informace z kapitoly 4.1.2. Poslední podmínka je také splněna, požadavek při obslužení uzlem přechází do dalšího uzlu s určitými pravděpodobnostmi. To je vidět na obrázku ??.

Po splnění Jacksonova teorému můžeme tedy určit vzorec pro výpočet středního počtu požadavků v uzlu (21).

$$L_{q_i} = \frac{\rho_i}{1 - \rho_i} \quad (21)$$

kde ρ_i je zatížení daného uzlu, pro který počítáme střední počet požadavků.

Podle daného vzorce vypočítáme všechny hodnoty (22).

$$\begin{aligned} L_{q_1} &= \frac{\rho_1}{1 - \rho_1} = \frac{0,8991228}{1 - 0,8991228} \doteq 8,913043447r \\ L_{q_2} &= \frac{\rho_2}{1 - \rho_2} = \frac{0,865153079}{1 - 0,865153079} \doteq 6,415816328 \\ L_{q_3} &= \frac{\rho_3}{1 - \rho_3} = \frac{0,92879257}{1 - 0,92879257} \doteq 13,04347823 \\ L_{q_4} &= \frac{\rho_4}{1 - \rho_4} = \frac{0,78947368}{1 - 0,78947368} \doteq 3,74\bar{9} \end{aligned} \quad (22)$$

4.1.4 Střední doba odezvy uzlu

Střední doba odezvy uzlu lze vypočítat podle vzorce (23).

$$T_{g_i} = \frac{L_{q_i}}{\Lambda_i} \quad (23)$$

kde L_{g_i} je střední počet požadavků v uzlu a Λ_i je střední doba frekvence toku v uzlu. Můžeme tedy vypočítat všechny hodnoty (24).

$$\begin{aligned}
T_{q_1} &= \frac{L_{q_1}}{\Lambda_1} = \frac{8,913043447}{13,4868421} \doteq 0,66086956 \\
T_{q_2} &= \frac{L_{q_2}}{\Lambda_2} = \frac{6,415816328}{8,65153079} \doteq 0,74158163 \\
T_{q_3} &= \frac{L_{q_3}}{\Lambda_3} = \frac{13,04347823}{18,57585139} \doteq 0,70217391 \\
T_{q_4} &= \frac{L_{q_4}}{\Lambda_4} = \frac{3,74\bar{9}}{15,78947368} \doteq 0,2375
\end{aligned} \tag{24}$$

4.2 Střední délka front

Střední délka front závisí na středním počtu požadavků v uzlech a jeho zatížení. Vzorcem pro tento výpočet tedy je (25).

$$L_{w_i} = L_{q_i} - \rho_i \tag{25}$$

Výsledkem pro naši síť tedy bude (26).

$$\begin{aligned}
L_{w_1} &= L_{q_1} - \rho_1 = 8,913043447 - 0,8991228 = 8,013920647 \\
L_{w_2} &= L_{q_2} - \rho_2 = 6,415816328 - 0,865153079 = 5,550663249 \\
L_{w_3} &= L_{q_3} - \rho_3 = 13,04347823 - 0,92879257 = 12,11468566 \\
L_{w_4} &= L_{q_4} - \rho_4 = 3,74\bar{9} - 0,78947368 = 2,96052632
\end{aligned} \tag{26}$$

4.2.1 Střední doba mezi požadavky

Střední doba mezi požadavky závisí na střední frekvenci toku v uzlu. Vzorec pro výpočet tedy bude (27)

$$T_{a_i} = \frac{1}{\Lambda_i} \tag{27}$$

Výsledkem pro naši síť tedy bude (28).

$$\begin{aligned}
T_{a_1} &= \frac{1}{\Lambda_1} = \frac{1}{13,4868421} \doteq 0,0741463415 \\
T_{a_2} &= \frac{1}{\Lambda_2} = \frac{1}{8,65153079} \doteq 0,1155864811 \\
T_{a_3} &= \frac{1}{\Lambda_3} = \frac{1}{18,57585139} \doteq 0,0538\bar{3} \\
T_{a_4} &= \frac{1}{\Lambda_4} = \frac{1}{15,78947368} \doteq 0,06\bar{3}
\end{aligned} \tag{28}$$

4.3 Statistika celého systému

4.3.1 Střední počet požadavků celého systému

Střední počet požadavků v celé síti lze jednoduše vyjádřit jako součet středního počtu požadavků jednotlivých uzlů (29).

$$L_q = \sum_i L_{q_i} \tag{29}$$

V našem případě tedy výsledná hodnota středního počtu požadavků v síti bude dána vzorcem (30).

$$\begin{aligned}
L_q &= L_{q_1} + L_{q_2} + L_{q_3} + L_{q_4} \\
L_q &= 8,913043447 + 6,415816328 + 13,04347823 + 3,74\bar{9} \\
L_q &= 32,122338
\end{aligned} \tag{30}$$

4.3.2 Střední doba odezvy celého systému

Střední doba odezvy celého systému je závislá na středním počtu zpracovaných požadavků a také na středních hodnotách vstupních generátorů. Jelikož máme dva vstupní generátory, musí se obě stření hodnoty zohlednit, takže výsledný vzorec bude (31).

$$T_q = \frac{L_q}{\Lambda} = \frac{L_q}{\Lambda_A + \Lambda_B} \quad (31)$$

Po dosazení známých hodnot dostaneme výsledek (32).

$$T_q = \frac{32,122338}{10 + 5} \doteq 2,1414892 \quad (32)$$

4.4 Statistika měřeného místa

Jako měřené místo jsem dostal statistiky toku, které se nachází před kanálem obsluhy pro zpracování fotografií. Měřím zde propustnost toku, celkový počet požadavků, střední dobu mezi dvěma požadavky, rozptyl a odchylku.

Propustnost toku je počet požadavků za jednotku času. Tu jsem již vypočítal v předchozích výpočtech a to v kapitole 4.1.1. Propustnost se totiž rovná střední době frekvence uzlu. Teoretickým výsledkem propustnosti tedy je (33).

$$\Lambda_3 \doteq 18,57585139 \quad (33)$$

Teoretická hodnota počtu požadavků, které projdou daným uzlem se vypočítá pomocí vzorce, který je stejný jako výpočet střední frekvence uzlů v systému v kapitole 4.1.1. Avšak místo hodnot Λ dosadím počet vygenerovaných požadavků jednotlivým generátorem. Jelikož generátory mají střední hodnoty rovnou 10 a 5, tak vyplývá, že by měly teoreticky generovat požadavky v poměru 2:1. Budu tedy provádět výpočet pro 10000 prvků. Generátor se střední hodnotou 10 vygeneruje tedy 6666,6 požadavků a generátor se střední hodnotou 5 vygeneruje 3333,3 požadavků. Následující vzorce ukazují postup výpočtu.

$$\begin{aligned} COUNT_A &= 6666,6 \\ COUNT_B &= 3333,3 \end{aligned} \quad (34)$$

$$\begin{aligned}
COUNT_1 &= COUNT_A + (1 - p_1) \cdot COUNT_1 + (1 - p_4) \cdot COUNT_4 \\
COUNT_2 &= COUNT_B + (1 - p_2) \cdot COUNT_2 + (1 - p_4) \cdot COUNT_4 \\
COUNT_3 &= p_1 \cdot COUNT_1 + p_2 \cdot COUNT_2 \\
COUNT_4 &= p_3 \cdot COUNT_3
\end{aligned} \tag{35}$$

$$\begin{aligned}
COUNT_1 &\doteq 8990 \\
COUNT_2 &\doteq 5767 \\
COUNT_3 &\doteq 12382 \\
COUNT_4 &\doteq 10525
\end{aligned} \tag{36}$$

Pro moje měření při 10000 požadavcích je tedy očekávaná hodnota 12382 požadavků.

Střední dobu mezi dvěma požadavky jsem také již vypočítal a to v kapitole 4.2.1. Výsledkem tedy je (37).

$$T_{a_3} = \frac{1}{\Lambda_3} = \frac{1}{18,57585139} \doteq 0,0538\bar{3} \tag{37}$$

Rozptyl střední doby mezi dvěma požadavky lze teoreticky vypočítat podle vzorce pro výpočet rozptylu exponenciálního rozdělení (10). Výslednou hodnotou tedy bude (38).

$$D[x] = \frac{1}{(\Lambda_3)^2} = \frac{1}{18,57585139^2} = 0,00289803 \tag{38}$$

Nakonec ještě odchylka, která se vypočítá podle vzorce (39).

$$\sigma = \sqrt{D[x]} = \sqrt{0,00289803} = 0,0538\bar{3} \tag{39}$$

5 Implementace

Program byl implementován v programovacím jazyce Java. Jako knihovna pro simulování byla použita knihovna JSim verze 0.6.0. Všechny třídy ob-

sahují dokumentační komentáře a také komentáře kódu, takže budu popisovat třídy jen krátce.

5.1 Balík `distribution`

Balík obsahuje různé generátory náhodných veličin. Dle zadání se jedná o Gaussovo rozdělení a také exponenciální rozdělení. Jako pomocné mám ještě vytvořené rozdělení uniformní. Navíc jsem pro testování vyzkoušel další dvě rozdělení.

5.1.1 Rozhraní `IDistribution`

Rozhraní obsahuje jednu jedinou metodu, `nextGeneratedValue`, kterou musí všechny ostatní generátory obsahovat. Tato metoda generuje další náhodné číslo pomocí daného rozdělení. Všechny generátory tedy implementují rozhraní `IDistribution`.

5.1.2 Třída `ExponentialFromJSIM`

Třída pro generování exponenciálního rozdělení. Využívá stream ze třídy `JSimExponentialStream`. Byla vytvořena pouze jako testovací k vlastním a ověření řešení.

5.1.3 Třída `GaussianByExponential`

Třída pro generování Gaussovo rozdělení. Využívá ke generování čísel rozdělení exponenciální a to podle vzorce (12).

5.1.4 Třída `ExponentialDistribution`

Třída pro generování exponenciálního rozdělení. Generuje další hodnotu pomocí inverzní funkce k distribuční funkci, tedy podle vzorce (7).

5.1.5 Třída GaussianDistribution

Třída pro generování Gaussovo rozdělení pomocí rovnoměrného rozdělení. Vzorec (11) pro generování je odvozen v kapitole 3.4.2.

5.1.6 Třída UniformDistribution

Třída pro generování rovnoměrného rozdělení.

5.2 Balík generator

Balík obsahující generátory sítě a požadavků.

5.2.1 Třída NetworkGenerator

Třída slouží k vytvoření sítě pomocí zadaných parametrů. Podporuje možnost spustit aplikaci s parametry pro exponenciální nebo gaussovské rozdělení, které bylo vytvořeno autorem, ale také obdobná rozdělení, při kterém bylo využito rozdělení knihovny JSim.

5.2.2 Třída PhotoRequestGenerator

Třída slouží ke generování požadavků do sítě. Podle parametrů se opět nastavují různá rozdělení, stejně jako u třídy `NetworkGenerator` (viz kapitola 5.2.1).

5.3 Balík main

Balík obsahující hlavní třídu i metodu celého programu.

5.3.1 Třída Main

Třída obsahuje hlavní metodu celého programu. Vytváří se zde celá programová reprezentace sítě, začíná a spouští se simulace běhu otevřené sítě front.

5.4 Balík process

Balík obsahuje programovou reprezentaci všech součástí systému.

5.4.1 Třída Crossing

Třída představuje rozcestí (větvení) sítě. Křižovatka je postavená na principu, že do ní vstupuje požadavek, který podle určité pravděpodobnosti směřuje dále do jednoho ze dvou uzlů. K tomu je připravena kolekce instancí třídy `Follower`, kde se vždy vzestupně porovnávají pravděpodobnosti. Pokud je pravděpodobnost nižší, tak se požadavek přesměruje na zadaný uzel.

5.4.2 Třída Follower

Třída funguje jako přepravka pro hodnoty do křižovatky. Obsahuje proměnné pro pravděpodobnost a uzel, do kterého požadavek má být přesměrován.

5.4.3 Rozhraní `IProcessInput`

Rozhraní obsahuje jednu jedinou metodu, `receiveRequest`, kterou musí obsahovat všechny uzly, které mohou přijímat požadavek, tedy rozcestí (větvení), jeden z procesů výroby fotografií nebo výstup ze sítě.

5.4.4 Třída Output

Třída představuje výstup ze sítě front. Při výstupu požadavku ze sítě front se odečte jeden požadavek z počtu očekávaných. Pokud je hodnota counteru

rovna nule, končí běh programu a vypočítají se statistiky.

5.4.5 Třída PhotoProcess

Třída představuje hlavní uzel (proces) v síti front. Každý uzel obsahuje svojí vlastní frontu požadavků, které postupně zpracovávají. Doba obsluhy jednoho požadavku je dána pomocí daného rozdělení.

5.4.6 Třída PhotoRequest

Třída představuje jeden požadavek, který prochází sítí. Systém obsahuje čas příchodu do sítě, aby se mohly vypočítat statistiky.

5.4.7 Třída PhotoCounter

Třída slouží jako přepravka, obsahuje celočíselnou hodnotu představující počet požadavků. Využívají se jako vstupní a výstupní počet požadavků a podle něj se také vytvořena ukončovací podmínka simulace.

5.5 Balík statistics

Balík obsahuje statistiky pro síť front.

5.5.1 Rozhraní IStatistics

Rozhraní obsahuje jednu jedinou metodu, `processStatistics`, kterou musí obsahovat všechny statistické třídy. Jedná se o metodu pro započtení hodnot do potřebných statistik.

5.5.2 Třída Statistics

Třída počítá všechny statistiky pro uzly, které počítáme a dále porovnáváme. Počítané hodnoty můžete vidět v kapitole 4.1.

5.5.3 Třída FlowStatistics

Třída počítá všechny statistiky pro tok. Počítané hodnoty můžete vidět v kapitole 4.4.

5.6 Spuštění aplikace

Pro spuštění aplikace je vytvořen JAR soubor s názvem `VSP_Sobehart.jar`. Dle zadání se aplikace spouští s dvěma parametry. Prvním parametrem je počet kroků simulace, které se provedou. Druhým parametrem je poté zkratka rozdělení, které bude použité při generování hodnot. Možnosti rozdělení jsou:

- **EXP** – exponenciální rozdělení vytvořené autorem
- **GAUSS** – Gaussovo rozdělení generované pomocí rovnoměrného rozdělení
- **EXPJSIM** – exponenciální rozdělení převzaté z knihovny `JSim`
- **GAUSSBYEXP** – Gaussovo rozdělení generované pomocí exponenciálního rozdělení

Spuštění je tedy pomocí příkazu:

```
java -jar VSP_Sobehart.jar 10000 EXP
```

V příkazu můžete vidět, že se program spustí se 10000 počtem kroků a exponenciálním rozdělením. Při spuštění s Gaussovo rozdělením se provede simulace pro všechny tři koeficienty 0,1 a 0,5 a 0,9.

6 Výsledky a porovnání

Popis výsledků aplikace jsem se rozhodl udělat pomocí tabulky, kde vždy porovnám vypočítané teoretické hodnoty s těmi, které jsou výsledkem simulace.

6.1 Výsledky pro uzly sítě

6.1.1 Uzel Podání žádosti na internetu

Počet kroků = 10000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,9053273093645018	0,8991228
Lq	9,562708139880867	8,913043447
Tq	0,8517068642178602	0,66086956
Lw	11,346381949046645	8,013920647

Počet kroků = 100000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,896653128076603	0,8991228
Lq	8,703191480049482	8,913043447
Tq	0,6714007726141508	0,66086956
Lw	9.023001857721276	8,013920647

Load: 0.9026668078025032 Lq: 9.273987500285825 Tq: 0.6915414220394885
Lw: 9.342632243728161

Počet kroků = 500000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,8993594439965191	0,8991228
Lq	8,936352100095839	8,913043447
Tq	0,6408583287535077	0,66086956
Lw	8,6339751818738676	8,013920647

6.1.2 Uzel Podání žádosti na pobočce

Počet kroků = 10000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,8837253916968967	0,865153079
Lq	7,600329982563445	6,415816328
Tq	0,8547740835572778	0,74158163
Lw	7,488884701876217	5,550663249

Počet kroků = 100000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,857717661798500	0,865153079
Lq	6,028279213290727	6,415816328
Tq	0,6509841384636753	0,74158163
Lw	5,596165816406911	5,550663249

Počet kroků = 500000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,8642091027865474	0,865153079
Lq	6,364263883079575	6,415816328
Tq	0,7391827758631155	0,74158163
Lw	6,389284635698185	5,550663249

Load: 0.8642091027865474 Lq: 6.364263883079575 Tq: 0.7391827758631155
Lw: 6.389284635698185

6.1.3 Uzel Zpracování objednávek

Počet kroků = 10000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,9280029785260991	0,92879257
Lq	12,889463474020284	13,04347823
Tq	0,6017439176460575	0,70217391
Lw	11,136625720739016	12,11468566

Počet kroků = 100000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,9244204482217303	0,92879257
Lq	12,231091961668866	13,04347823
Tq	0,6458978632621963	0,70217391
Lw	12,952210063927156	12,11468566

Počet kroků = 500000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,9262494416502974	0,92879257
Lq	12,559219379171426	13,04347823
Tq	0,6698752661599979	0,70217391
Lw	12,430331430225342	12,11468566

6.1.4 Uzel Výroba fotografie

Počet kroků = 10000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,7841597285630988	0,78947368
Lq	3,633055700600989	3,74999999
Tq	0,21466766765348477	0,2375
Lw	3,360852755375727	2,96052632

Počet kroků = 100000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,7847522680589178	0,78947368
Lq	3,6458096955638126	3,74999999
Tq	0,23251959888229937	0,2375
Lw	3,658793547800341	2,96052632

Počet kroků = 500000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Load	0,7882261097559854	0,78947368
Lq	3,7220174255086818	3,74999999
Tq	0,2376088058585964	0,2375
Lw	3,7463587176353554	2,96052632

6.2 Celkové výsledky v síti

Počet kroků = 10000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Tq	2,2306178720450487	2,1414892
Lq	33,685557297065586	32,122338

Počet kroků = 100000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Tq	2,091400230404032	2,1414892
Lq	31,30837235057289	32,122338

Počet kroků = 500000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Tq	2,0820627313883833	2,1414892
Lq	31,58185278785552	32,122338

6.3 Statistiky měřeného místa - tok před uzlem Zpracování fotografií

V měřeném místě se soustředím na statistiky toku. U toku měřím propustnost (počet požadavků za jednotku času), poté měřím celkový počet požadavků, které tokem prošly. Dále se soustředím na časy mezi dvěma požadavky, na kterých měřím průměrnou dobu mezi dvěma požadavky, jejich rozptyl a odchylku.

Počet kroků = 10000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Propustnost	18,50725099857102	18,57585139
Počet požadavků	12385	12382
Průměrný čas	0,054009906629519586	0,053833333
Rozptyl	0,0029339641892077045	0,00289803
Odchylka	0,05416607969207024	0,053833333

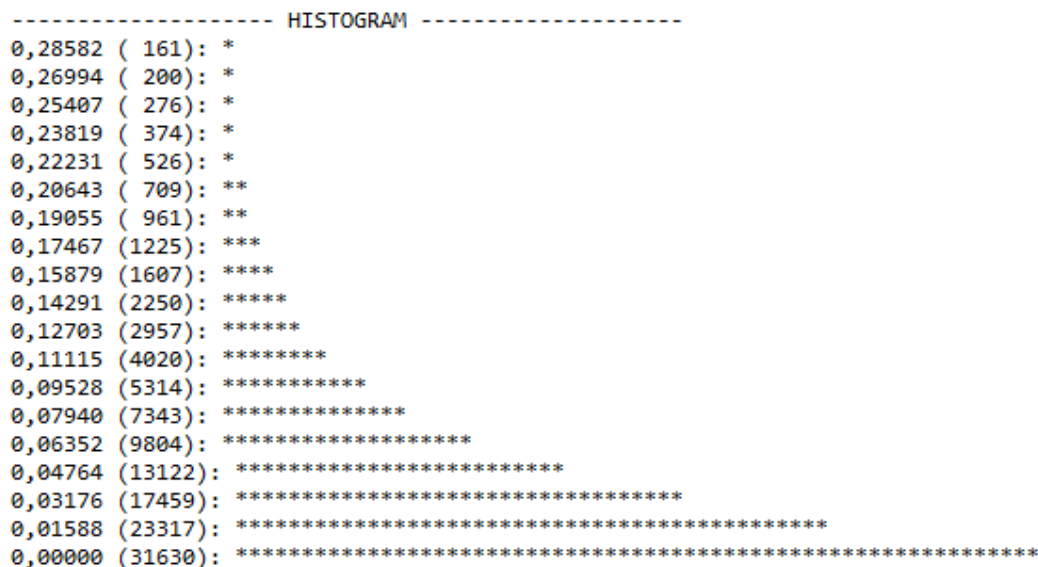
Počet kroků = 100000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Propustnost	18,504798891207233	18,57585139
Počet požadavků	123736	123820
Průměrný čas	0,05403541242730649	0,053833333
Rozptyl	0,0029458993826295206	0,00289803
Odchylka	0,05427614008594864	0,053833333

Počet kroků = 500000		
Veličina	Simulovaná hodnota	Teoretická hodnota
Propustnost	18,55618808182215	18,57585139
Počet požadavků	619154	619100
Průměrný čas	0,05389016037048494	0,053833333
Rozptyl	0,0028998202457758506	0,00289803
Odchylka	0,05384997906940959	0,053833333

V měřeném místě generuji také histogram toku. Histogram je upraven, poněvadž maximální hodnoty, kterých je málo, deformují histogram a není přesně vidět, jak náhodná veličina vypadá. Proto jsou oříznuty hodnoty, které jsou vyšší než polovina rozdílu maximální a minimální doby mezi dvěma požadavky. Histogram se tvoří dynamicky a implicitně je nastaven na vytvoření 20-ti stejně velkých intervalů. Proto se po oříznutí sníží velikost intervalu a je tak lépe vidět, jak vypadá náhodná veličina.

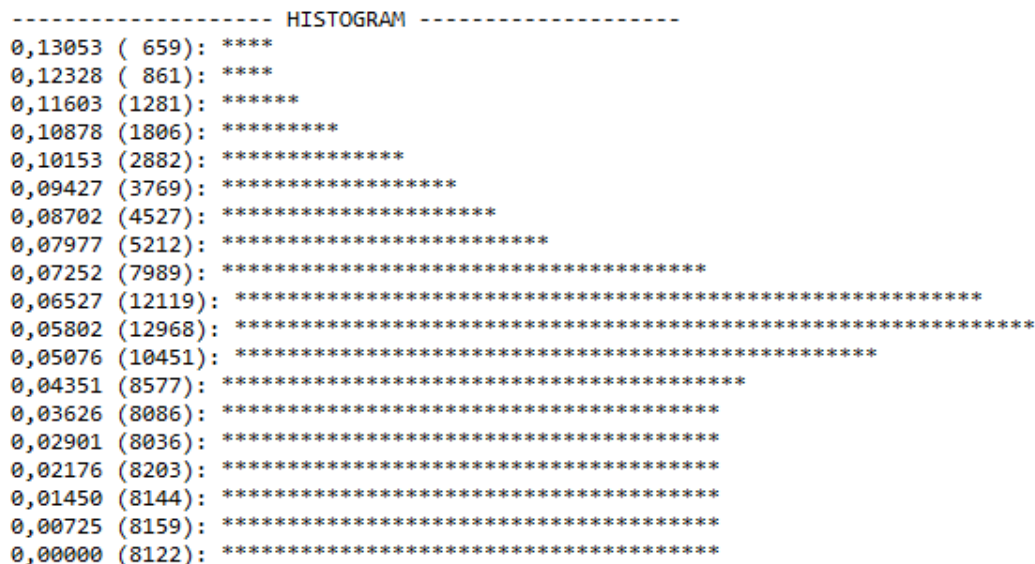
Při spuštění programu s exponenciálním rozdělením, je vidět na obrázku 3, že generované hodnoty mají exponenciální rozdělení.

Dále se snažím porovnat výsledky s normálním (Gaussovo) rozdělením, kdy vyzkouším 3 koeficienty variace. Vzhledem k faktu, jak funguje normální rozdělení, tak jsem zvolil 3 rozdílné koeficienty variace. Výsledky můžete vidět na následujících obrázcích 4, 5, 6.



Obrázek 3: Histogram exponenciálního rozdělení pro 100000 měření

Na obrázcích můžeme vidět, že při nízkém koeficientu variace má náhodná veličina tvar normálního rozdělení. Naopak při zvyšování koeficientu se začíná přeměňovat v exponenciální. Při vysokém koeficientu jsou již časy mezi



Obrázek 4: Histogram normálního rozdělení pro 100000 měření, koeficient variace je 0,1

```

----- HISTOGRAM -----
0,19068 ( 221): *
0,18008 ( 287): **
0,16949 ( 507): ***
0,15890 ( 699): ***
0,14830 ( 900): ****
0,13771 (1274): *****
0,12712 (1819): *****
0,11652 (2657): *****
0,10593 (3706): *****
0,09534 (5039): *****
0,08474 (6669): *****
0,07415 (8689): *****
0,06356 (10429): *****
0,05297 (12101): *****
0,04237 (12917): *****
0,03178 (13770): *****
0,02119 (13882): *****
0,01059 (13822): *****
0,00000 (14326): *****

```

Obrázek 5: Histogram normálního rozdělení pro 100000 měření, koeficient variace je 0,5

```

----- HISTOGRAM -----
0,22132 ( 312): **
0,20903 ( 472): **
0,19673 ( 580): **
0,18444 ( 809): ***
0,17214 (1143): ****
0,15985 (1542): *****
0,14755 (2009): *****
0,13525 (2699): *****
0,12296 (3585): *****
0,11066 (4510): *****
0,09837 (5714): *****
0,08607 (6911): *****
0,07378 (8405): *****
0,06148 (10051): *****
0,04918 (11652): *****
0,03689 (13208): *****
0,02459 (14803): *****
0,01230 (16435): *****
0,00000 (18230): *****

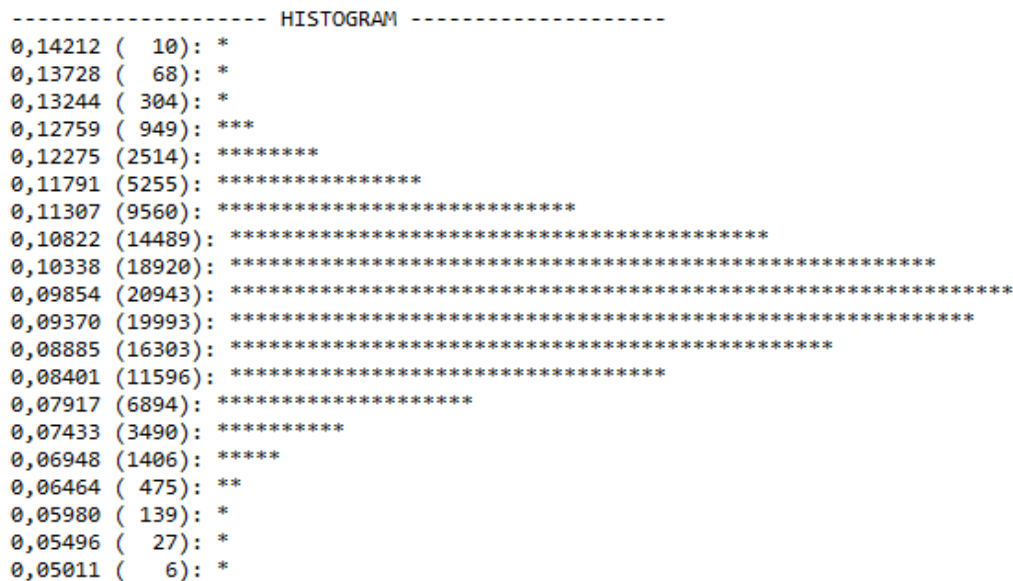
```

Obrázek 6: Histogram normálního rozdělení pro 100000 měření, koeficient variace je 0,9

dvěma požadavky exponenciální.

Při tvorbě normálního rozdělení jsem se potýkal s problémy a nemohl při-

jít na to, proč hodnoty blížící nulovému času mezi dvěma prvky nesnižují, ale zůstávají takřka konstantní. Nakonec jsem přišel na důvod, proč se histogram nesnižuje k nule. Je to fakt, že požadavky, které projdou místem ještě nejdou přímo na výstup, ale mohou se vrátit zpět (vzhledem k návrhu sítě front). Při vložení měřícího místa za generátor rozdělení a malou hodnotu korelace 0,1 jsem dostal graf na obrázku 7, který ukazuje, že generátor funguje v pořádku.



Obrázek 7: Histogram normálního rozdělení pro 100000 měření, koeficient variace je 0,1, měřeno za generátorem rozdělení

7 Závěr

Vytvořil jsem aplikaci, která funguje pro všechna zadaná rozdělení a simulované hodnoty se blíží hodnotám vypočítaným, nejvíce u exponenciálního rozdělení, které jsem sám vytvořil. Vypočítané hodnoty jsem se snažil co nejméně zaokrouhlovat, aby byl výsledek přesnější. Mohu říci, že mě překvapilo, že simulované hodnoty vycházely bez velké odchylky od vypočítaných. Hodnoty se růstem prvků, které projdou simulací, zpřesňovaly.

Generátory hodnot Gaussovo rozdělení vycházejí oba dva obdobně a při stejných koeficientech jsou histogramy hodnot skoro stejné. Z toho usuzuji, že fungují oba dobře. Generátor exponenciálního rozdělení jsem chtěl otestovat s generátorem z knihovny JSim. Ten však nikdy nedosáhl hodnot, které

by se blížili teoretickým hodnotám podobným způsobem, jako můj vlastní vytvořený generátor.

Tvorba semestrální práce byla jedna ze zábavnějších na této škole. Od začátku vývoje byl vidět postup, mohlo se různě zkoušet a experimentovat. Při implementaci mi nejvíce pomohla přednáška o simulacích a JSimu. Při teoretických výpočtech jsem naopak vycházel ze skript, které byly dostupné na stránkách předmětu. Další nástroj, který mi pomohl při testování správnosti mého systému a výpočtů byl QNAnalyzer. Rozhodně se vyplatilo složitě vytvořit svojí síť front do textového souboru, abych poté mohl své výsledky kontrolovat.

Avšak celková doba, kterou jsem strávil na semestrální práci byla až nečekaně dlouhá a zabrala mnoho času. Možná to mohlo být tím, že jsem se snažil o dokumentaci, kde budou všechny důležité informace.