KIV/VSS

Semestrální práce

Miroslav Liška – A17N0081P

topiker@students.zcu.cz

9.12.1992

21. ledna 2018

1 Zadání

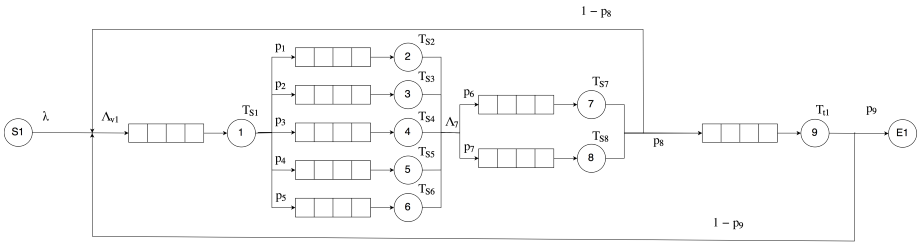
Cílem semestrální práce je vytvoření analytického výkonnostního nebo spolehlivostního modelu zadaného systému a jeho řešení. Dále vytvoření simulačního modelu pro tentýž problém a závěrem obě řešení porovnat.

2 Analýza

2.1 Popis problému

V rámci práce byl zvolen model inspirovaný jídelnou (menzou) na Borech. Navržený model je možné vidět na obrázku.1.

Schéma obsahuje jedno místo generování požadavků (S1). V reálném světě je míst generujících požadavků několik, nicméně se všichni musí dostat do budovi jednimi dveřmi. Z tohoto důvodu je vstupní proud požadavků spojen do jednoho a tedy i jejich střední doby generování. Strávnici se po vstupu hromadí na jednom místě - u turniketů. Následně si jdou vyzvednout jídlo k jednomu z pultů. Každý pult má jednu frontu a každé jídlo se vydává na jednom místě. Poté následuje placení, kde si student vybere jednu ze dvou pokladen. Po zaplacení by typicky následovalo stravování. Tato operace ale nebyla do simulace zahrnuta. Student tedy po dokončení stravování může dostat ještě hlad a vrátit se tak na začátek k turniketům, nebo se vydá odevzdat tác. Stroj s tácem zajišťuje také právě jednu frontu. V tuto chvíli strávník odchází ze zařízení, nebo má ještě hlad a vrací se zpět na začátek k turniketům. Všechny kanály obsluhy jsou tedy typu M/M/1 a všechny fronty jsou typu FIFO.



Obrázek 1: Schéma navrženého modelu

2.2 Naměřené hodnoty systému

V rámci experimentu byly naměřeny střední hodnoty jako:

- Počet strávníků, kteří přijdou za jednotku času - λ
- Počet strávníků, kteří mohou projít přes turniket za jednotku času - μ_1
- Počet strávníků, kteří mohou projít přes pokladnu za jednotku času - μ_7 a μ_8

- Počet strávníků, kteří mohou projít přes pokladnu za jednotku času - μ_7 a μ_8
- Počet strávníků, kteří mohou projít přes stroj s tácy za jednotku času - μ_9

Zbylé hodnoty, jako pravděpodobnosti výběru jídel (p_1 až p_5), doba jejich vydání (μ_1 až μ_5), pravděpodobnost volby pokladny (p_6 až p_7), pravděpodobnost návratu na začátek po zaplacení jídla (1 - p_8) a pravděpodobnost návratu na začátek po odevzdání tácu (1 - p_9), byly odhadnuty na základě předchozích zkušeností s tímto zařízením. Hodnoty jsou shrnuty v tabulkách níže.

Název	
λ_1	12

Tabulka 1: Generátory požadavků

μ_1	20
μ_2	12
μ_3	11
μ_4	13
μ_5	12
μ_6	14
μ_7	11
μ_8	12
μ_9	18

Tabulka 2: Zpracovávající zařízení

p_1	0.15
p_2	0.25
p_3	0.18
p_4	0.19
p_5	0.23
p_6	0.55
p_7	0.45
p_8	0.96
p_9	0.96

Tabulka 3: Pravděpodobnosti přechodů

3 Analytické řešení

Nejprve je nutné vypočítat střední frekvenci přichodů požadavků pro všechny uzly. Výsledná střední frekvenci přichodů se vždy skládá z těch prodů, které do uzlu přitečou mínus ty proudy, které z uzlu odtečou. Pro mnou navržený experiment jsou rovnice následující:

$$\Lambda_1 = \lambda + (1 - P_8) \cdot (\Lambda_7 + \Lambda_8) + (1 - P_9) \cdot \Lambda_9$$
$$\Lambda_2 = P_1 \cdot \Lambda_1$$
$$\Lambda_3 = P_2 \cdot \Lambda_1$$
$$\Lambda_4 = P_3 \cdot \Lambda_1$$
$$\Lambda_5 = P_4 \cdot \Lambda_1$$
$$\Lambda_6 = P_5 \cdot \Lambda_1$$
$$\Lambda_7 = \Lambda_2 + \Lambda_3 + \Lambda_4 + \Lambda_5 + \Lambda_6$$
$$\Lambda_8 = P_6 \cdot \Lambda_7$$
$$\Lambda_9 = P_7 \cdot \Lambda_7$$
$$\Lambda_{10} = P_8 \cdot (\Lambda_8 + \Lambda_9)$$

V rámci řešení této soustavy rovnic je klíčové vyjádřit Λ_1 v prvním řádku dosazením dalších neznámých z řádků ostatních. Vyjde nám tak rovnice, kde na jedné straně bude Λ_1 a na druhé vstupní proud λ, který známe ze zadání. Pak už se jen dosazuje do dalších rovnic.

Λ_1	13.02
Λ_2	1.95
Λ_3	3.20
Λ_4	2.35
Λ_5	2.47
Λ_6	3.02
Λ_7	7.16
Λ_8	5.85
Λ_9	12.5

Tabulka 4: Analiticky vypočítané frekvence příchodů požadavků jednotlivých uzlů

Jakmile máme určenou frekvenci příchodů požadavků všech uzlů a známe dobu, za jakou dobu umí jednotlivé uzly odbavit požadavek, můžeme spočítat jejich zatížení. Díky tomu dokážeme určit, zda síť není zahlcená, tedy je ve stacionárním režimu a můžeme použít Littleovy vzorce pro další výpočty.

Zatížení uzlu se spočítá jako:

$$\rho = \frac{1}{m} \cdot \frac{\lambda}{\mu}$$

tedy poměř střední hodnoty frekvence příchodu požadavků ku střední hodnotě frekvence doby obsluhy. m reprezentuje počet obslužných kanálů. V našem případě je to 1. Pokud jsou všechny hodnoty ρ menší než 1, je systém ve stacionárním režimu.

ρ_1	0.65
ρ_2	0.16
ρ_3	0.29
ρ_4	0.18
ρ_5	0.20
ρ_6	0.21
ρ_7	0.65
ρ_8	0.48
ρ_9	0.69

Tabulka 5: Zatížení jednotlivých uzlů

Z výsledků tedy vyplývá, že můžeme pro výpočet dalších statistik použít Littleovy vzorce:

$$L_q = \lambda \cdot T_q$$
$$L_w = \lambda \cdot T_w$$
$$T_w = L_w \cdot T_a$$

L_q je střední počet požadavků v celém systému hromadné obsluhy (SHO). Lze také vyjádřit jako:

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

který nám říká, že v systému je to, co je ve frontě a v kanálech obsluhy. Pokud se jedná o SHO klasifikace M/M/1, je možné spočítat L_q jako:

$$L_q = \frac{\rho}{(1-\rho)}$$

S využitím tohoto vzorce si spočítáme střední počet požadavků pro všechny uzly.

Pokud sečteme dílčí výsledky, vyjde nám střední počet požadavků pro celý systém tedy $L_q = 8,29$.

T_q, neboli střední doba odezvy systému SHO lze vyjádřit jako:

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

Lq_1	1.86
Lq_2	0.19
Lq_3	0.41
Lq_4	0.22
Lq_5	0.25
Lq_6	0.27
Lq_7	1.86
Lq_8	0.95
Lq_9	2.27
Lq	8,29

Tabulka 6: Střední počet požadavků v uzlech

V našem případě lze využít Littleova vzorce a vypočítat střední dobu odezvy jako střední počet požadavků v systému děleno střední frekvencí příchodu požadavků, tedy $T_q = 0.69$

4 Řešení simulačním programem

4.1 Použité nástroje a jazyky

Pro řešení jsem se rozhodl využít simulační knihovnu J-Sim, která je napsána v jazyce Java a je možné tak využít objektově orientované programování. Alternativou bylo použít knihvnu C-Sim, který je psán pro ANSI C.

4.2 Generování náhodných čísel

Jedním z požadavků semestrální práce bylo umožnit generovat nově příchozí požadavky s Gaussovským rozdělením. Pro vygenerování pseudonáhodných čísel s Gaussovým rozdělením byla navržena metoda Box-Mullerovi transformace [**?**]. Tato metoda je navržena tak, že během jednoho běhu vrací dvě pseudonáhodná čísla (Z_0 a (Z_1). Vstupem metody jsou dvě náhodná čísla – U_1 a U_2 – s uniformním rozdělením uvnitř intervalu $(0, 1)$. Samotný výpočet výsledných pseudonáhodných čísel je pak následující:

$$Z_0 = R\cos(\theta)\sqrt{-2\ln U_1}\cos(2\pi U_2) \tag{1}$$

$$Z_1 = R\sin(\theta)\sqrt{-2\ln U_1}\sin(2\pi U_2) \tag{2}$$

4.2.1 Výpočet statistik za běhu

Střední hodnotu Gaussova rozdělení je možné vypočítat jako:

$$\overline{x} = \frac{1}{n}(\sum_{i=1}^{n} x_i) \tag{3}$$

a směrodatnou odchylku jako:

$$s = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}} \tag{4}$$

Pro realizaci těchto výpočtů je ale nutné si všechna vygenerovaná čísla ukládat. Pro přepočet střední hodnoty po vygenerování nového pseudonáhodného čísla je v tomto případě možno použít:

$$\bar{x}_n = \frac{n\bar{x}_{n-1} + x_{new}}{n+1} \tag{5}$$

kde n je počet vygenerovaných čísel, $\bar{x}_{n-1}$ je stará střední hodnota a $\bar{x}_{new}$ je nově vygenerované číslo.

Pro přepočet rozptylu je to pak:

$$\sigma^2_{new} = \frac{n(\sigma^2_{old} + \bar{x}^2) + x^2_{new}}{n+1} - \bar{x}^2_{new} \tag{6}$$

kde n je počet vygenerovaných čísel, σ^2_{old} je hodnota staré odychlky, x_{new} je nově vygenerované číslo a $\bar{x}^2_{new}$ je střední hodnota získaná z předchozí rovnice.

Pro vygenerování pseudonáhodných čísel s Gaussovým rozdělením byla navržena metoda Box-Mullerovi transformace [**?**]. Tato metoda je navržena tak, že během jednoho běhu vrací dvě pseudonáhodná čísla (Z_0 a (Z_1). Vstupem metody jsou dvě náhodná čísla – U_1 a U_2 – s uniformním rozdělením uvnitř intervalu $(0, 1)$. Samotný výpočet výsledných pseudonáhodných čísel je pak následující:

$$Z_0 = R\cos(\theta)\sqrt{-2\ln U_1}\cos(2\pi U_2) \tag{7}$$

$$Z_1 = R\sin(\theta)\sqrt{-2\ln U_1}\sin(2\pi U_2) \tag{8}$$

Zde je ukázka histogramu z algoritmu mnou implementovaným. Vstupním parametrem byla střední hodnota 1000 a odchylka 10 a vygeneroval se celkem jeden milion čísel.

```
E_teorie=1000.0
D_teorie=100.0
E_vypocet=1000.0369173235879
D_vypocet=98.70743444378536
733,33:*
766,67:**
800,00:*****
833,33:*********
866,67:***************
900,00:**********************
933,33:*************************
966,67:****************************
```

```
1000,00:****************************
1033,33:************************
1066,67:********************
1100,00:***************
1133,33:**********
1166,67:*****
1200,00:**
1233,33:*
```

Z výsledků je vidět, že je teoretická hodnota blízská hodnotě ze simulace.

4.3 Implementace

Místem spuštění celého programu je třída Main. Ta přečtě vstup od uživatele a spustí podle toho simulaci s patřičnými parametry. Dále je implementace rozdělená do několika balíků. Nyní bude rámcově popsáno, jakou funkcionalitu jednotlivé balíky obsahují a vyzdvihnuty klíčové části implementace. Přiložený kód je pak sám o sobě podrobně okomentován.

4.3.1 Request

V tomto balíku jsou třídy, reprezentující požadavek, který běží systémem (`Request`). Požadavky v sobě uchovávají informaci, kdy byly do simulace přidány. Tento údaj pak slouží k určení délky setrvání požadavku v systému.

4.3.2 Connector

Tento balík zajištuje propojení výstupních front jedné výkonné jednotky (generátor požadavků, server atd.) do vstupní fronty jiné výkonné jednotky a tedy předávání požadavků mezi nimi. Zároveň jsou zde zahrnuty konstrukce pro rozhodnutí se, do které z možných jednotek bude požadavek předám (pravděpodobnostní rozhodování).

4.3.3 OperationUnit

V tomto balíku jsou umístěny všechny typy výkonných jednotek:

- `GeneratorServer` - generuje nové požadavky do systému.
- `WaveServer` - generuje nové požadavky do systému. Navíc umí vygenerovat několik požadavků naráz a vytvořit tak nárazovou zátěž.
- `ProcessServer` - zpracovává požadavky z fronty a předává je dál
- `EndingServer` - jedná se o koncový uzel systému, zde se požadavky ruší a evidují se patřičné statistiky
- `IServer` - rodičovská třída všem předchozím

Zpracování požadavků je simulováno metodou `hold`. Pokud nemá výkonná jednotka ve frontě žádné požadavky, je uspána. Pokud přijde požadavek, je nutné ji znovu probudit. O to se stará výše zmíněný balík `Connector`

4.3.4 NumberGenerator

V tomto balíku je zajištěno generování náhodných čísel, ať s Gaussovým rozdělením nebo exponenciálně. Generátory implementují rozhraní `ILambdaGenerator`, které je pak předáno výkonným jednotkám z balíku `OperationUnit`

4.3.5 Network

Posledním balíkem je balík `Network`. V něm se vytváří instance simulace, výkonných jednotek a jejich vzájemném propojení a generátory náhodných čísel. Následně je odstartována simulace spuštěním činnosti generátoru požadavků.

4.4 Výsledky simulace

Výsledky získané simulací můžeme vidět v tabulce níže. Pro generování požadavků byl použit exponenciální generátor. Hodnoty jsou zprůměrované z 5 měření a získané po běhu 10 000 požadavků.

Název	Λ	ρ	L_q	T_q
Server 1	13.10	0.66	1.86	0.14
Server 2	1.98	0.16	0.19	0.09
Server 3	3.25	0.29	0.41	0.12
Server 4	2.41	0.18	0.22	0.09
Server 5	2.43	0.20	0.25	0.10
Server 6	3.01	0.21	0.27	0.08
Server 7	7.33	0.66	2.04	0.27
Server 8	5.76	0.47	0.89	0.15
Server 9	12.53	0.70	2.30	0.18

Tabulka 7: Výsledky získané simulací

5 Porovnání výsledků

Můžeme pozorovat, že nasimulované výsledky nejsou rovny výsledkům z analytického řešení, nicméně se k sobě velmi blíží. Bylo tedy ověřeno, že ne vždycky musíme řešit výpočty analyticky. Někdy jsou výpočty dokonce tak složité, že je vhodné je získat simulací.

6 Uživatelská příručka

Program se spouští z příkazové řádky příkazem `run.bat [EXP|GAUSS]`. V případě, že nejsou zadány vstupní parametry, je program spuštěn nejprve s exponenciálním generátorem náhodných čísel a následně s Gaussovským generátorem s koeficienty variace 0.05, 0.2 a 0.7

7 Závěr

V rámci semestrální práce byl navržena otevřená síť front podle reálného prostředí. Následně byly naměřeny některé stěžejní hodnoty. Zbylé hodnoty byly odhadnuty. Následně byly analyticky pro tuto síť vypočítány statistiky.

Tato síť pak byla nasimulována počítačovou simulací. Z této simulace byly následně získány statistiky. Statistiky ze simulováného a analitického řešení byly porovnány a potvrdilo se, že je možné analitické řešení alternovat řešením simulačním a naopak, neboť si hodnoty byly velmi blízské.

Semestrální práci hodnotím velmi pozitivně, neboť jsem si utřídil znalosti získané ze semestru a mám pocit, že jsem probírané problematice porozuměl. Náročnost práce byla optimální.