ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

ФАКУЛЬТЕТ ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Кафедра вычислительных систем

ОТЧЁТ

по лабораторной работе №4
по дисциплине «Моделирование»
на тему «Моделирование одноканальной СМО с очередью»

Выполнил:
студент группы ИВ-222
Терешков Р. В.

Проверил:
д.т.н., доцент
Родионов А. С.

Новосибирск – 2016

Задание

В ходе выполнения лабораторной работы была реализована программа на языке программирования C, выполняющая моделирование зависимости времени ожидания заявки в очереди от интенсивности поступления заявок и интенсивности их обработки.

Ход работы и результаты моделирования

Условно разобьём ход работы на два опыта. В первом опыте задаётся случайная величина ξ. Исходя из этой величины вычисляются следующие параметры:

$$\tau = \frac{\sqrt{\xi} \times 3.1337}{14.88};$$

$$\sigma = \frac{\tau + \xi}{1.228 + \xi \times 0.0666},$$

которые зависят от одной случайной величины. По этим параметрам смоделируем среднее значение времени нахождения заявки в очереди СМО:

ξ	τ	σ	ω
0.8401877172	0.1930379294	0.8047201326	0.0000000000
0.3943829268	0.1322554383	0.4198777659	0.6724646943
0.7830992238	0.1863643690	0.7573020773	0.9059780912
0.7984400335	0.1881809421	0.7700900530	1.4750992264
0.9116473579	0.2010795595	0.8634386198	2.0441097199
0.1975513693	0.0936040217	0.2345838677	2.8139443181
0.3352227557	0.1219329925	0.3656292905	2.9265951933
0.7682295948	0.1845865290	0.7448740396	3.1076379549
0.2777747108	0.1109943414	0.3118885807	3.7415176531
0.5539699558	0.1567464062	0.5618780212	3.8966598277
0.4773970519	0.1455105066	0.4944516645	4.3130273422
0.6288709248	0.1670072191	0.6267335392	4.6404717877
0.3647844728	0.1271957670	0.3928630066	5.1400095599
0.5134009102	0.1508977637	0.5263053564	5.3819748028
0.9522297252	0.2055064010	0.8964840805	5.7027737582
0.9161950680	0.2015804736	0.8671523816	6.3976773651
0.6357117280	0.1679131072	0.6326068979	7.0969166395
0.7172969294	0.1783626730	0.7020530468	7.5511608643
0.1416025554	0.0792483141	0.1784753393	8.1739655971
0.6069688763	0.1640732185	0.6078740370	8.1883677179
			3.75

В опыте 2 параметры рассчитываются от двух независимых случайных величин ξ_1 и ξ_2:

ξ_1	ξ_2	$\tau(\xi_1)$	$\sigma(\xi_2)$	ω
0.8401877172	0.3943829268	0.1930379294	0.4683383761	0.0000000000
0.7830992238	0.7984400335	0.1863643690	0.7686721581	0.2819740072
0.9116473579	0.1975513693	0.2010795595	0.3211768971	0.8495666057
0.3352227557	0.7682295948	0.1219329925	0.6958939776	1.0488105104
0.2777747108	0.5539699558	0.1109943414	0.5257073615	1.6337101466
0.4773970519	0.6288709248	0.1455105066	0.6098054318	2.0139070015
0.3647844728	0.5134009102	0.1271957670	0.5075269239	2.4965166663
0.9522297252	0.9161950680	0.2055064010	0.8701980533	2.7985371892
0.6357117280	0.7172969294	0.1679131072	0.6938622682	3.5008221353
0.1416025554	0.6069688763	0.0792483141	0.5409997931	4.1154360895
0.0163005716	0.2428867706	0.0268878393	0.2168298969	4.6295480432
0.1372315768	0.8041767542	0.0780156108	0.6883748116	4.7683623293
0.1566790893	0.4009443942	0.0833604414	0.3859916455	5.3733766995
0.1297904468	0.1088088020	0.0758710073	0.1495084459	5.6834973376
0.9989245180	0.2182569053	0.2104848406	0.3450538069	5.6225209429
0.5129323944	0.8391122347	0.1508288954	0.7710513220	5.8167458544
0.6126398326	0.2960316177	0.1648379108	0.3693706238	6.4229592656
0.6375522677	0.5242871901	0.1681560061	0.5482885315	6.6241738833
0.4935829870	0.9727750239	0.1479566799	0.8669114578	7.0245057349
0.2925167844	0.7713576978	0.1139016149	0.6919480966	7.7775155777
				3.45

Среднее время нахождения заявки в очереди выделена цветом в каждой из таблиц.
Формула нахождения времени ожидания заявки в очереди:

$$\omega = \omega_{i-1} - \tau_i + \sigma_{i-1}$$

Отсортируем значение σ по убыванию и возрастанию:

ξ	τ	σ_1	ω_1	σ_2	ω_1
0.84018771	0.19303792	0.03513865	0.00000000	0.93424708	0.00000000
0.39438292	0.13225543	0.14432508	0.00000000	0.89648408	0.80199164
0.78309922	0.18636436	0.16630610	0.00000000	0.86715238	1.51211135
0.79844003	0.18818094	0.17398778	0.00000000	0.86343861	2.19108279
0.91164735	0.20107955	0.17847533	0.00000000	0.80472013	2.85344185
0.19755136	0.09360402	0.19382491	0.00000000	0.77486339	3.56455796
0.33522275	0.12193299	0.23458386	0.00000000	0.77009005	4.21748837
0.76822959	0.18458652	0.25483696	0.01863790	0.75730207	4.80299189
0.27777471	0.11099434	0.27863977	0.16248052	0.74487403	5.44929963
0.55396995	0.15674640	0.31188858	0.28437388	0.70205304	6.03742726
0.47739705	0.14551050	0.36562929	0.45075196	0.63260689	6.59396980
0.62887092	0.16700721	0.39286300	0.64937403	0.62673353	7.05956948

0.36478447	0.12719576	0.41987776	0.91504127	0.60787403	7.55910725
0.51340091	0.15089776	0.42583426	1.18402127	0.56187802	8.01608353
0.95222972	0.20550640	0.49445166	1.40434913	0.52630535	8.37245515
0.91619506	0.20158047	0.52630535	1.69722032	0.49445166	8.69718003
0.63571172	0.16791310	0.56187802	2.05561257	0.42583426	9.02371859
0.71729692	0.17836267	0.60787403	2.43912792	0.41987776	9.27119017
0.14160255	0.07924831	0.62673353	2.96775364	0.39286300	9.61181963
0.60696887	0.16407321	0.63260689	3.43041396	0.36562929	9.84060941
0.01630057	0.02688783	0.70205304	4.03613302	0.31188858	10.1793508
0.24288677	0.10379021	0.74487403	4.63439585	0.27863977	10.3874492
0.13723157	0.07801561	0.75730207	5.30125428	0.25483696	10.5880733
0.80417675	0.18885576	0.77009005	5.86970059	0.23458386	10.6540545
0.15667908	0.08336044	0.77486339	6.55643020	0.19382491	10.8052780
0.40094439	0.13335108	0.80472013	7.19794251	0.17847533	10.8657518
0.12979044	0.07587100	0.86343861	7.92679164	0.17398778	10.9683561
0.10880880	0.06946827	0.86715238	8.72076199	0.16630610	11.0728757
0.99892451	0.21048484	0.89648408	9.37742953	0.14432508	11.0286969
0.21825690	0.09838717	0.93424708	10.1755264	0.03513865	11.0746348
			2.57		**7.00**

По результатам моделирования можно сделать вывод: если очередь организована так, что первыми обрабатываются заявки с наименьшим временем выполнения, то среднее время ожидания в очереди существенно уменьшается, и наоборот.

Листинг программы

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

int N = 30;

int main()
{
	int avg1 = 0, avg2 = 0;
	double ksi[N], tau[N], sig1[N], w1[N], sig2[N], w2[N];

	FILE *fp = fopen("res3", "w");

	w1[0] = w2[0] = 0.0;
	for (int i = 0; i < N; ++i) {
		ksi[i] = rand() / (double) RAND_MAX;
		tau[i] = sqrt(ksi[i]) * 3.1337 / 14.88;
		sig1[i] = (tau[i] + ksi[i]) / (1.228 + ksi[i] * 0.0666);
		sig2[i] = sig1[i];
//		if (i) {
//			w[i] = w[i - 1] - tau[i] + sig[i - 1];
//			avg += w[i];
```

```
//          }
      }

      for (int i = 0; i < N - 1; ++i) {
            for (int j = 0; j < N - i - 1; ++j) {
                  if (sig1[j] > sig1[j + 1]) {
                        double tmp = sig1[j];
                        sig1[j] = sig1[j + 1];
                        sig1[j + 1] = tmp;
                  }
                  if (sig2[j] < sig2[j + 1]) {
                        double tmp = sig2[j];
                        sig2[j] = sig2[j + 1];
                        sig2[j + 1] = tmp;
                  }
            }
      }

      for (int i = 1; i < N; ++i) {
            w1[i] = w1[i - 1] - tau[i] + sig1[i - 1];
            avg1 += w1[i];
            w2[i] = w2[i - 1] - tau[i] + sig2[i - 1];
            avg2 += w2[i];
      }

      fprintf(fp, "[KSI]\n\n");
      for (int i = 0; i < N; ++i) fprintf(fp, "%.10lf\n", ksi[i]);
      fprintf(fp, "\n\n");
      fprintf(fp, "[TAU]\n\n");
      for (int i = 0; i < N; ++i) fprintf(fp, "%.10lf\n", tau[i]);
      fprintf(fp, "\n\n");
      fprintf(fp, "[SIG1]\n\n");
      for (int i = 0; i < N; ++i) fprintf(fp, "%.10lf\n", sig1[i]);
      fprintf(fp, "\n\n");
      fprintf(fp, "[W1]\n\n");
      for (int i = 0; i < N; ++i) fprintf(fp, "%.10lf\n", w1[i]);
      fprintf(fp, "\n\n");
      fprintf(fp, "%.10lf\n\n", avg1 / (double) N);
      fprintf(fp, "[SIG2]\n\n");
      for (int i = 0; i < N; ++i) fprintf(fp, "%.10lf\n", sig2[i]);
      fprintf(fp, "\n\n");
      fprintf(fp, "[W2]\n\n");
      for (int i = 0; i < N; ++i) fprintf(fp, "%.10lf\n", w2[i]);
      fprintf(fp, "\n\n");
      fprintf(fp, "%.10lf\n\n", avg2 / (double) N);

      return 0;
}
```