



Suleyman Demirel Üniversitesi
Mühendislik Fakültesi
Bilgisayar Mühendisliği

Paralel Programlama Dersi

Sevdanur GENC - 0921012067

Paralel Programlama

Histogram Hesaplama - Seri ve Paralel Programlama Ile Analiz Sonuclari

Seri Programlama Kod Ve Analizleri

Seri Programlama kodlari ve ornek ekran ciktisi su sekildedir.

```
c:\users\sevdanur\documents\visual studio 2010\Projects\HistogramCalculation\Release\HistogramCalculation.exe

Hesaplama Islemine Basladi : 52.000000 saniye surdu.

Rastgele Uretilen 1000 Deger ;
6.6,7.8,8.9,10.10,13.15,16.16,18.18,18.19,21.21,24.24,24.25,26.28,31.32,33.36,37.38,38.38,39.40,42.43,43.45,45.45,47.50,52.52,52.52,53.53,54.54,55.55,
57.60,60.61,62.63,64.65,65.67,69.70,70.70,72.73,73.73,74.74,74.76,76.76,78.79,80.81,82.82,84.85,86.87,88.88,89.90,90.91,91.92,92.94,95.96,97.97,97.98,
99.99,100.100,101.105,106.108,109.109,111.112,113.113,115.117,117.117,119.119,120.121,122.122,123.124,124.125,125.126,126.130,132.133,133.133,134.135,
135.137,139.140,140.141,141.141,142.143,145.145,145.145,145.146,147.149,151.154,155.155,155.155,157.158,158.159,159.160,160.160,161.163,165.166,168.169,170.170,
173.177,178.178,179.179,179.180,180.184,184.184,184.185,185.186,187.190,191.192,194.195,196.197,198.198,198.199,200.200,201.203,203.204,205.206,206.
206.207,209.209,209.210,211.215,216.218,219.221,221.224,224.226,226.226,227.228,229.229,230.230,230.230,230.231,233.235,237.238,242.242,244.246,24
6.248,251.251,252.253,253.254,254.255,256.257,262.262,264.268,269.271,272.273,274.275,275.276,277.277,278.279,280.281,281.282,285.287,290.291,292.292,
295.295,296.296,299.300,303.305,307.307,309.309,309.310,310.310,312.312,314.314,314.315,318.318,318.318,318.322,322,322.323,324.327,327,327,327,329,330,330,33
1.331,333.334,335.337,337.339,341.342,342.343,345.345,345.348,349.351,353.354,355.355,355.356,357.357,357.358,359.359,360.361,364.364,365.366,368.369,
370.370,372.372,373.374,374.376,378.380,382.383,383.385,385.386,391.391,392.395,395.395,395.396,398.399,401.402,402.403,403.405,406.408,409.411,412.41
3.414,415.416,417.418,418.419,421.422,423.423,423.424,425.426,426.428,429.430,431.432,433.433,434.434,435.435,436.436,437.437,438.438,438.441,441.
441.442,442.443,444.444,446.446,447.447,447.447,451.452,453.453,455.456,456.457,459.459,459.461,463.467,467.467,468.469,469.469,471.472,473.473,47
4.478,479.479,479.480,480.481,482.485,485.485,487.489,490.491,492.493,494.495,495.496,497.499,498.500,500.500,501.501,502.503,504.504,504.505,505.
506.507,508.509,509.510,511.512,512.515,516.516,516.517,518.518,518.520,522.522,522.522,522.523,523.523,524.525,526.526,528.532,533.536,536.539,541.54
1.541,541.542,542.542,543.545,547.548,550.550,550.550,550.551,555.555,556.556,558.558,559.563,564.564,565.567,567.567,569.571,574.574,574.575,576.577,578.
578.578,579.581,581.582,583.583,583.586,586.586,586.587,588.588,590.590,591.592,595.601,603.604,605.606,606.607,608.608,609.609,610.611,611.61
1.612.613.613.614.614.616.616.616.617.617.619.621.622.622.623.623.624.624.627.629.631.631.632.634.635.636.637.637.645.646.647.647.648.648.649.649,
650.651.652.653.654.654.656.657.659.659.664.664.666.668.668.670.671.672.674.675.675.676.677.679.679.680.681.683.683.684.685.685.689.690.69
1.692.692.692.693.695.695.697.697.697.698.701.701.701.703.703.703.704.708.711.713.716.716.718.718.719.721.722.724.724.725.726.726.726.727.728.728.729.
729.730.731.732.733.733.734.734.738.740.742.744.745.746.746.747.748.749.751.751.752.752.753.753.754.755.755.756.758.758.765.765.765.765.770.771.773.773.77
4.775.776.777.779.779.782.782.783.783.785.786.787.787.788.788.790.792.792.795.795.795.795.798.799.800.801.801.802.804.806.806.807.808.808.808.809.810.813.
816.816.817.817.818.818.818.819.820.821.822.828.828.828.832.833.833.833.835.836.838.838.838.838.838.839.839.841.841.844.845.846.847.848.852.852.852.85
4.854.854.854.858.859.859.860.861.862.864.864.866.868.868.869.869.870.871.873.873.876.877.878.878.880.880.880.882.883.885.885.885.885.886.
886.888.889.892.894.895.895.896.898.901.901.901.902.903.904.904.905.905.909.909.912.913.915.916.916.917.918.919.920.922.922.923.923.924.924.92
5.925.928.930.932.932.933.934.934.935.936.937.938.938.940.942.942.942.944.945.946.946.946.946.947.948.948.950.953.953.954.955.958.958.958.959.959,
960.961.967.970.970.970.971.973.973.975.975.976.977.977.978.979.980.982.982.983.984.985.985.987.987.989.990.991.994.994.995.997.998.999,

Dizinin En Buyuk Elemani : 999
Dizinin En Kucuk Elemani : 6
Dizinin Deger Araligi (max-min) : 993
Histogram Icin Sutun Sayisi Giriniz : 20
Aralik Olcum Sonucu : 50

6 - 55 Araligi ... 52 Adet
56 - 105 Araligi ... 56 Adet
106 - 155 Araligi ... 54 Adet
156 - 205 Araligi ... 51 Adet
206 - 255 Araligi ... 49 Adet
256 - 305 Araligi ... 36 Adet
306 - 355 Araligi ... 52 Adet
356 - 405 Araligi ... 47 Adet
406 - 455 Araligi ... 60 Adet
456 - 505 Araligi ... 58 Adet
506 - 555 Araligi ... 55 Adet
556 - 605 Araligi ... 46 Adet
606 - 655 Araligi ... 56 Adet
656 - 705 Araligi ... 48 Adet
706 - 755 Araligi ... 47 Adet
756 - 805 Araligi ... 39 Adet
806 - 855 Araligi ... 50 Adet
856 - 905 Araligi ... 53 Adet
906 - 955 Araligi ... 54 Adet
956 - 1005 Araligi ... 39 Adet

Hesaplama Islemi Sona Erdi : 2745.000000 saniye surdu.
Hesaplanan Sure Farki : 2.693000 saniyedir.
```

Dizi boyutu 100 olarak verilmiştir, Sutun sayilari ise en fazla 20 olarak tanimlanmistir.

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <float.h>
#include <conio.h>
#include <string.h>
#include <iostream>
#include <math.h>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
```

```
    int sayac = 0, i, sutunSayisi = 0, yendi zi, min, max, degerAraligi, aralikOlcumu,
    baslangic[21], bitis[21];
```

```

int dizi [1000];

clock_t HesaplamaYasla, HesaplamaYi Bitir;
double SureFarki;

srand(time(NULL));

printf("\nHesaplama Isl emine Basl adi : ");
HesaplamaYasla = clock();
printf("%f sani ye surdu. \n", (double) HesaplamaYasla );

printf("\n");
for(i=0; i<1000; i++)
{
    dizi [i]= rand()%1000;
}

min = dizi [0];
max = dizi [0];

for(i=1; i<1000; i++)
{
    if(dizi [i]<min) min = dizi [i];
    if(dizi [i]>max) max = dizi [i];
}

for(i=1; i<1000; i++)
    for(int j=i; j<1000; j++)
    {
        if(dizi [i]>dizi [j])
        {
            yeni dizi = dizi [i];
            dizi [i]=dizi [j];
            dizi [j]=yeni dizi ;
        }
    }

cout << "Rastgele Uretil en 1000 Deger ; " << endl ;
for(i = 1; i<1000; i++)
    cout << dizi [i] << ", ";

printf("\n");
cout << "\nDizinin En Buyuk Elemani : " << max << endl ;
cout << "Dizinin En Kucuk Elemani : " << min << endl ;

degerAraligi = max - min;
cout << "Dizinin Deger Araligi (max-min) : " << degerAraligi << endl ;

cout << "Histogram Icin Sutun Sayisi Giriniz : ";
cin >> sutunSayisi;

aralikOl cumu = ((float) degerAraligi) / ((float) sutunSayisi) + 0.5f;
cout << "Aralik Ol cum Sonucu : " << aralikOl cumu << endl ;

baslangic[0] = min;

for(i = 0; i<sutunSayisi; i++)
{

```

```

        baslangic[i+1] = (baslangic[i] + aralikOlumu);
        bitis[i] = (baslangic[i+1] - 1);
        bitis[i+1] = (bitis[i] + aralikOlumu);
    }

    for(int j = 0; j<sutunSayisi; j++)
    {
        sayac = 0;
        for(i = 1; i<1000; i++)
        {
            if ((dizi[i] >= baslangic[j]) && (dizi[i] <= bitis[j]))
                sayac++;
        }

        printf("\n");
        printf("%d - %d Araligi ... \t %d Adet \t |", baslangic[j],
bitis[j], sayac);
        for(int k=0; k<sayac; k++)
            printf("*");

        printf("\n");
        printf("\nHesaplama Islemi Sona Erdi : ");
        HesaplamaYiBitir = clock();
        printf("%f saniye surdu. \n", (double) HesaplamaYiBitir );
        SureFarki = ((double) (HesaplamaYiBitir - HesaplamaYiBasla)) / CLOCKS_PER_SEC ;
        printf("Hesaplanan Sure Farki : %f saniye dir.", SureFarki);

        getch();
        return 0;
    }
}

```

Seri PROGRAMLAMA ANALİZ SONUCU;

Seri bir şekilde kodlanmış olan histogram hesaplama uygulamasında tek thread'in başlangıç ve bitiş sürelerini birer değişken de tuttuktan sonra farklarını alarak çalışma sürelerini belirlemiş oldum. Örnek ekran çıktısında yapılan işlemlerin süresi 2.693 saniye sürmüştür.

Paralel Programlama Kod Ve Analizleri

Paralel Programlama kodlari ve ornek ekran ciktisi su sekildedir.

```
C:\users\sevdanur\documents\visual studio 2010\Projects\HistogramCalculation\Release\HistogramCalculation.exe
Hesaplama Islemine Basladi : 59.000000 saniye surdu.
Kullandiginiz Toplam Thread Sayisi : 4
Rastgele Uretilen 1000 Deger :
1.2 3.3 3.6 7.2 8.9 8.8 9.1 10.1 11.1 11.15 15.17 18.18 19.20 21.21 21.22 22.23 24.25 27.28 28.28 30.30 31.31 35.35 35.36 37.37 38.38 39.40 40.40 41.41 41.4
3.44 44.49 50.52 53.53 55.58 58.60 60.60 61.64 67.67 70.71 71.72 72.72 72.73 75.75 75.75 77.80 80.82 82.84 84.85 86.87 87.88 88.90 93.93 97.98 99.1
01.102.103.105.106.107.108.109.110.111.112.112.113.113.114.115.115.116.116.117.118.119.123.124.125.127.129.129.129.131.132.132.139.140.141.141.142
.142.142.142.142.144.145.145.145.146.148.150.152.152.153.153.153.154.154.155.156.157.159.161.161.164.164.168.168.168.169.169.170.170.171.173.173.1
74.175.176.177.179.181.181.182.184.185.185.186.186.187.188.189.190.190.191.191.192.192.193.193.195.195.195.196.196.199.200.200.200.200.202.202.202.205
205.209.212.213.213.213.215.215.216.220.221.221.221.222.222.223.224.224.226.227.229.230.232.233.234.235.240.240.245.247.249.249.253.253.255.256.2
56.257.258.259.260.261.262.262.263.264.264.264.264.269.270.270.270.271.272.272.279.279.281.281.281.282.282.285.285.286.287.287.288.288.289.290.291
.292.292.292.292.295.296.296.297.297.299.300.302.303.303.303.303.306.308.309.309.310.313.313.313.313.314.314.314.315.316.316.317.318.318.318.3
21.322.322.323.323.324.326.328.329.330.333.333.334.334.335.337.337.342.342.343.347.348.350.350.350.350.353.353.355.355.355.355.356.357.357.357.358.359
.359.360.361.361.362.363.363.365.368.369.369.370.371.372.374.375.376.380.382.383.384.386.389.391.391.392.392.393.398.401.402.404.405.409.410.410.411.4
11.413.413.413.414.416.416.416.418.421.421.422.422.423.423.423.423.424.425.426.426.428.428.429.430.430.432.433.434.436.437.439.439.441.443.446
.447.448.450.451.452.454.455.457.457.458.459.460.461.462.463.464.464.466.466.467.467.467.472.472.474.474.476.477.477.477.478.480.481.482.483.483.483.4
84.484.484.485.485.487.487.488.488.489.489.491.492.493.496.497.498.500.503.503.503.504.505.506.508.508.510.510.511.511.512.512.514.515.515.518.519.519
.520.520.523.525.526.527.527.529.529.532.534.535.536.537.537.538.538.538.539.540.540.541.541.541.542.543.543.545.546.547.547.548.548.548.549.549.549.5
49.550.555.555.556.556.556.557.558.559.561.561.565.565.565.565.570.573.574.576.576.577.578.580.584.584.584.585.585.587.588.588.589.589.589.591.591
.593.593.595.596.596.598.600.600.600.601.601.602.604.604.605.606.607.608.608.609.610.611.616.617.617.617.618.619.619.620.622.623.624.624.625.625.6
25.625.626.626.626.627.627.627.629.629.629.629.634.634.634.635.636.637.637.637.639.641.643.644.646.646.646.648.648.648.649.650.650.651.651.652.653.654
.655.655.657.658.658.659.662.662.662.663.664.667.667.668.668.670.671.673.673.673.674.675.676.676.678.678.678.679.681.683.685.685.686.687.688.689.690.6
90.692.693.694.694.695.695.696.698.699.700.701.701.702.703.704.704.704.705.705.705.706.710.711.711.712.712.713.716.717.718.718.721.722.723.723.724.724
.725.726.726.728.729.734.734.734.734.736.737.740.741.741.745.745.748.748.750.752.753.753.754.756.756.757.757.757.758.758.758.759.759.760.760.762.763.7
63.767.766.767.769.771.771.773.773.774.775.777.778.781.783.783.786.786.787.788.788.789.790.790.796.798.798.798.800.801.802.805.807.808.811.812.813.813
.814.815.815.815.818.818.819.823.824.824.825.825.827.827.827.829.829.831.831.832.832.833.833.835.836.838.840.841.842.843.844.844.847.848.850.850.851.8
51.853.855.855.858.859.861.864.865.866.867.868.869.869.870.870.874.875.875.877.878.881.881.882.885.886.887.888.888.890.892.893.893.894
.895.896.896.898.900.900.900.900.901.902.902.902.903.905.909.909.911.912.912.913.913.913.923.923.924.924.924.926.928.929.930.931.932.932.932.934.935.9
36.937.938.938.938.940.941.941.942.942.943.944.944.944.945.945.945.946.948.948.949.949.951.954.954.954.955.956.958.958.958.959.961.961.962.962.962.962
.963.964.966.966.970.971.971.971.972.972.974.974.975.976.977.977.982.985.985.986.989.990.992.993.993.994.995.996.997.998.999.
Dizinin En Buyuk Elemani : 998
Dizinin En Kucuk Elemani : 0
Dizinin Deger Araligi (max-min) : 998
Histogram Icin Sutun Sayisi Giriniz : 20
Aralik Olcum Sonucu : 50
0 - 49 Araligi ... 58 Adet
50 - 99 Araligi ... 46 Adet
100 - 149 Araligi ... 48 Adet
150 - 199 Araligi ... 57 Adet
200 - 249 Araligi ... 44 Adet
250 - 299 Araligi ... 52 Adet
300 - 349 Araligi ... 49 Adet
350 - 399 Araligi ... 45 Adet
400 - 449 Araligi ... 48 Adet
450 - 499 Araligi ... 52 Adet
500 - 549 Araligi ... 59 Adet
550 - 599 Araligi ... 43 Adet
600 - 649 Araligi ... 62 Adet
650 - 699 Araligi ... 54 Adet
700 - 749 Araligi ... 46 Adet
750 - 799 Araligi ... 47 Adet
800 - 849 Araligi ... 44 Adet
850 - 899 Araligi ... 45 Adet
900 - 949 Araligi ... 55 Adet
950 - 999 Araligi ... 48 Adet
Hesaplama Islemi Sona Erdi : 995.000000 saniye surdu.
Hesaplanan Sure Farki : 0.936000 saniyedir._
```

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <float.h>
#include <conio.h>
#include <string.h>
#include <iostream>
#include <math.h>
#include <omp.h>
```

```
using namespace std;
```

```
int main(int argc, char *argv[])
{
    int sayac = 0, i, j, sutunSayisi = 0, temp, min, max, degerAraligi, aralikOlcumu;
    int dizi[1000], baslangic[21], bitis[21];
    int ToplamThreadSayisi, ThreadID, chunk = 10;
```

```
clock_t HesaplamaYasla, HesaplamaYaslaBitir;
double SureFarki;
```

```
srand(time(NULL));
```

```

printf("\nHesaplama Islemi ne Basladi : ");
HesaplamayaBasla = clock();
printf("%f saniye surdu. \n", (double) HesaplamayaBasla );

#pragma omp parallel shared(dizi, baslangic, bitis, ToplamThreadSayisi, chunk)
private(ThreadID, i, j, temp, sutunSayisi, sayac, min, max, degerAraligi, aralikOlumu)
{
    ThreadID = omp_get_thread_num();

    if (ThreadID == 0)
    {
        ToplamThreadSayisi = omp_get_num_threads();
        printf("Kullandiginiz Toplam Thread Sayisi : %d\n", ToplamThreadSayisi);
    }

    #pragma omp parallel for schedule (static, chunk)
    for(i=0; i<1000; i++)
    {
        dizi[i] = rand()%1000;
    }

    min = dizi[0];
    max = dizi[0];

    #pragma omp parallel for schedule (static, chunk)
    for(i=1; i<1000; i++)
    {
        if(dizi[i]<min) min = dizi[i];
        if(dizi[i]>max) max = dizi[i];
    }

    #pragma omp parallel for schedule (static, chunk)
    for(i=1; i<1000; i++)
        for(j=i; j<1000; j++)
        {
            if(dizi[i]>dizi[j])
            {
                temp = dizi[i];
                dizi[i]=dizi[j];
                dizi[j]=temp;
            }
        }

    #pragma omp single
    {
        cout << "Rastgele Uretilen 1000 Deger ; " << endl;

        for(i = 1; i<1000; i++)
            cout << dizi[i] << ", ";

        printf("\n");

        cout << "\nDizinin En Buyuk Elemani : " << max << endl;
        cout << "Dizinin En Kucuk Elemani : " << min << endl;

        degerAraligi = max - min;
    }
}

```

```

        cout << "Dizinin Deger Araligi (max-min) : " << degerAraligi <<
endl;

        cout << "Histogram Icin Sutun Sayisi Giriniz : ";
        cin >> sutunSayisi;

        aralikOlcumu = ((float) degerAraligi) / ((float) sutunSayisi) +
0.5f;
        cout << "Aralik Olcum Sonucu : " << aralikOlcumu << endl;
    }

    baslangic[0] = min;

    #pragma omp parallel for schedule (static, chunk)
    for(i = 0; i<sutunSayisi; i++)
    {
        baslangic[i+1] = (baslangic[i] + aralikOlcumu);
        bitis[i] = (baslangic[i+1] - 1);
        bitis[i+1] = (bitis[i] + aralikOlcumu);
    }

    #pragma omp parallel for schedule (static, chunk)
    for(j = 0; j<sutunSayisi; j++)
    {
        sayac = 0;
        for(i = 1; i<1000; i++)
        {
            if ((dizi[i] >= baslangic[j]) && (dizi[i] <= bitis[j]))
                sayac++;
        }

        printf("\n");
        printf("%d - %d Araligi ... \t %d Adet \t |", baslangic[j],
bitis[j], sayac);

        for(int k=0; k<sayac; k++)
            printf("*");

    }

    printf("\n");
    printf("\nHesaplama Islemi Sona Erdi : ");
    HesaplamaYiBitir = clock();
    printf("%f saniye surdu. \n", (double) HesaplamaYiBitir );
    SureFarki = ((double) (HesaplamaYiBitir - HesaplamaYasla)) / CLOCKS_PER_SEC ;
    printf("Hesaplanan Sure Farki : %f saniyedir.", SureFarki);

    getch();
    return 0;
}

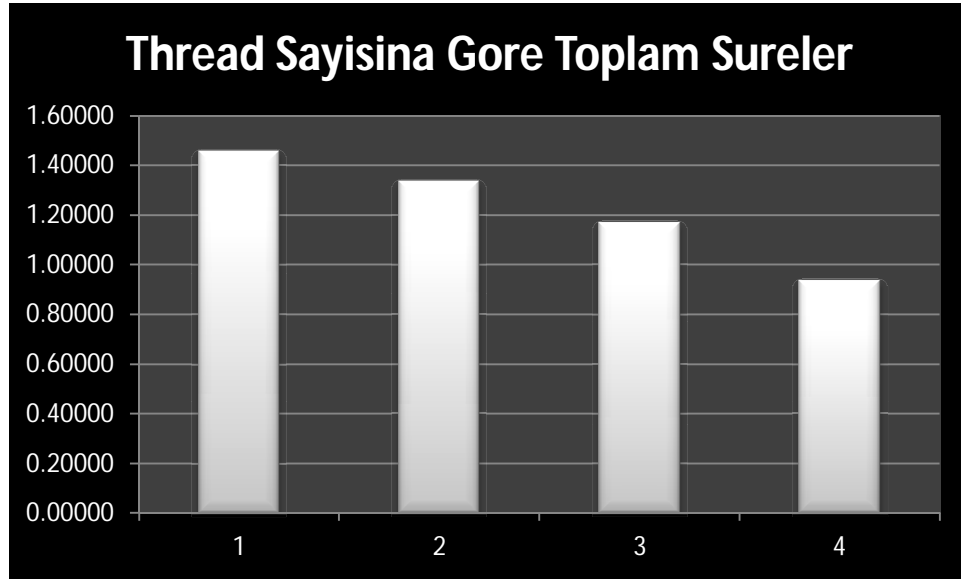
```

PARALEL PROGRAMLAMA ANALİZ SONUCU;

Paralel bir sekile donusturulmus olan histogram hesaplama uygulamasinda dort thread'le birlikte calisilmis olup, baslangic ve bitis surelerini birer degisken de tuttukten sonra farklarini alarak calisma surelerini belirlemis oldum. Ornek ekran ciktisinda yapilan islemlerin suresi 0.936 saniye surmustur.

Paralel Programlama Analiz Sonuclari

Thread Sayisi	Paralel Prog. Toplam Suresi
1	1.45800
2	1.33700
3	1.16800
4	0.93600



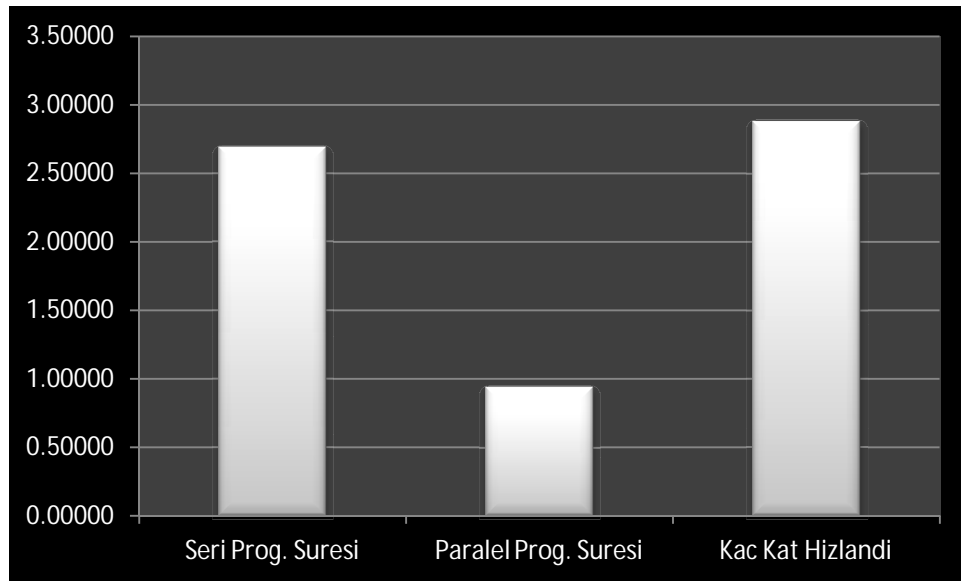
Paralel koda cevrimis uygulamamizi Omp_Num_Threads=1 degerine gore 1.458 saniye , Omp_Num_Threads=2 degerine gore 1.3370 saniye, Omp_Num_Threads=3 degerine gore saniye ve Omp_Num_Threads=4 degerine gore de 0.9360 saniyeye kadar thread'lerin calisma surelerinin geriledigini gozlemleyebiliriz.

Seri/Paralel Programlama'da Hiz Karsilastirilmesi (Paralel'de 4 Thread)

Seri kodla yazmis oldugum bir uygulamayi paralel bir kodla yeniden duzenleyip derledikten sonra hiz konusunda ne kadar fazla artabilecegini gostermek icin mazx 4thred ile calisarak bir analiz yapacak olursak;

Seri / Paralel Programlama'da Hiz Karsilastirmasi (Paralel'de 4 Thread)

Seri Prog. Suresi	Paralel Prog. Suresi	Kac Kat Hizlandi
2.69300	0.93600	2.87714



Histogram Hesaplama'da 1000 tane uretilmis rastgele sayilarin seri kod ile paralel kod seklinde calistiginda aralarindaki sure farkinin saniye bakiminda gozlemlendiginde yaklasik 2.5 ile 3 saniye arasinda degistigini gozlemleyebiliriz.