2014

MPI Matris Çarpımı

BIL 542 Paralel Hesaplama

Bu raporda C programlama dilinde yer alan MPI kütühanesi yardımı ile iki matrisin çarpımının yapılması, parelel programlamanın matris çarpımındaki etki ve hız ölçümleri yer almaktadır.

M.Yiğit Güngör
131419108
20.12.2014

İçindekiler Tablosu

1. Matris Kavramları

Bir matris sayıları düzenli bir dikdörtgen biçiminde yazılmış bir örneğidir. Yani, herhangi bir veri tablosu matristir. Bir matrisin büyüklüğü satır ve sütun sayısıyla belirtilmektedir. *k* satırlı ve n sütunlu bir matrise *k* x *n* (*k* çarpı *n*) matris denir. *i* satır sayısı, *j* sütun sayısı (*i*, *j*). giriş olarak adlandırılır. a_{ij} olarak yazılır.

Bir vektör bir satırda ya da sütunda düzenlenmiş sıralı sayılar kümesidir. Satır vektörü matrisin bir satırı, sütun vektörü matrisin bir sütununu temsil etmektedir. Örneğin, 1972 yılında gözlemlenmiş beş değişken bir satır vektörüdür. Aynı şekilde tüketim için zaman serisi değerleri bir sütun vektörüdür.

Bir matris sütunlar vektörünün bir kümesi olarak görülebilir. Bir matrisin boyutu içerdiği satır ve sütun sayısına eşittir. **A**, *n* x *k* matrisidir. Bu ifade her zaman n satır ve k sütun sayısına sahiptir anlamına gelmektedir. Eğer *n* = *k* ise, o zaman *A* matrisi kare matristir.

$$\mathbf{A} = [a_{ik}] = [\mathbf{A}]_{ik} = \begin{bmatrix} a_{11} & a_{12} & . & . & a_{1k} \\ a_{21} & a_{22} & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ a_{n1} & a_{n2} & . & . & a_{nk} \end{bmatrix}$$

2. Matris Çarpımı

A ve B matrislerinin çarpılabilmesi için A matrisinin sütun sayısı, B matrisinin satır sayısına eşit olmalıdır.

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_{11} & \boldsymbol{a}_{12} & \dots & \boldsymbol{a}_{1\boldsymbol{p}} \\ & & \dots & \\ \boldsymbol{a}_{i1} & \boldsymbol{a}_{i2} & \dots & \boldsymbol{a}_{\boldsymbol{ip}} \\ & & \dots & \\ \boldsymbol{a}_{\boldsymbol{m}1} & \boldsymbol{a}_{\boldsymbol{m}2} & \dots & \boldsymbol{a}_{\boldsymbol{mp}} \end{bmatrix} \quad x \quad B = \begin{bmatrix} b_{11} & \dots & b_{1j} & \dots & b_{1n} \\ & & \dots & & \dots \\ b_{i1} & \dots & b_{ij} & \dots & b_{in} \\ & & & \dots & \\ b_{p1} & \dots & b_{pj} & \dots & b_{pn} \end{bmatrix}$$

m x n türünde A matrisi ile n x p türünde B matrisinin çarpımı m x p türünde olur.

$$= \begin{bmatrix} c_{11} & \dots & c_{1j} & \dots & c_{1n} \\ & \dots & & \dots & \\ c_{i1} & \dots & c_{ij} & \dots & c_{in} \\ & & \dots & & \\ c_{m1} & \dots & c_{mj} & \dots & c_{mn} \end{bmatrix}$$

Çarpma işlemi birinci matrisin satırları ile ikinci matrisin sütunları çarpılıp toplanarak yapılır.

$$AB = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1p} \\ & & \dots & \\ a_{i1} & a_{i2} & \dots & a_{ip} \\ & & \dots & \\ a_{m1} & a_{m2} & \dots & a_{mp} \end{bmatrix} \begin{bmatrix} \boldsymbol{b}_{11} & \dots & \boldsymbol{b}_{1\boldsymbol{j}} & \dots & \boldsymbol{b}_{1\boldsymbol{n}} \\ & \dots & & \dots & \\ \boldsymbol{b}_{i1} & \dots & \boldsymbol{b}_{\boldsymbol{ij}} & \dots & \boldsymbol{b}_{\boldsymbol{in}} \\ & & \dots & & \\ \boldsymbol{b}_{\boldsymbol{p}1} & \dots & \boldsymbol{b}_{\boldsymbol{pj}} & \dots & \boldsymbol{b}_{\boldsymbol{pn}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}_{11} & \dots & \boldsymbol{c}_{1\boldsymbol{j}} & \dots & \boldsymbol{c}_{1\boldsymbol{n}} \\ & \dots & & \dots & \\ \boldsymbol{c}_{i1} & \dots & \boldsymbol{c}_{\boldsymbol{ij}} & \dots & \boldsymbol{c}_{\boldsymbol{in}} \\ & & \dots & & \\ \boldsymbol{c}_{\boldsymbol{m}1} & \dots & \boldsymbol{c}_{\boldsymbol{mj}} & \dots & \boldsymbol{c}_{\boldsymbol{mn}} \end{bmatrix}$$

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{ip}b_{pj} \quad , 1 \le i \le m \; ; \; 1 \le j \le n$$

3. **MPI Matris Çarpım:**

```
se364@linuxpc:~/Desktop$ mpiexec -n 1 ./matris

İşlemci Sayısı = 1

   matrix a
--------------
  3  6  7  5  3
  5  6  2  9  1
  2  7  0  9  3
  6  0  6  2  6
  1  8  7  9  2

   matrix b
--------------
  0  2  3  7  5
  9  2  2  8  9
  7  3  6  1  2
  9  3  1  9  4
  7  8  4  5  0


Çözüm Matrisi
--------------
169     78      80      136     103
156     63      52      171     119
165     69      41      166     109
102     84      80      96      50
216     82      78      169     127
```

Şekil 1 - MPI Matris Çarpımı

Şekil 1'de MPI kütüphanesi ile 5X5 matris çarpımına ilişkin sonuçlar gözükmektedir. Matris A ve B kodun içerisinde yer alan 0 ile 10 arası değişken değer üreterek matris oluşturmaktadır. En altta yer alan Çözüm Matrisi A ve B matrislerinin çarpımına ait sonuçların yer aldığı matrisi göstermektedir.

3.1. **Matris Çarpım Performansı**

Matris çarpımı performans testimizde 1500 x 1500 matris üzerinde test yapılmaktadır. Kod üzerinden elde edeceğimiz A ve B matrislerini 1,2,3 ve 4 işlemci üzerinde işlem yaptırarak. Performans farklarını gözlemleyeceğiz.

- 1 İşlemci Testi

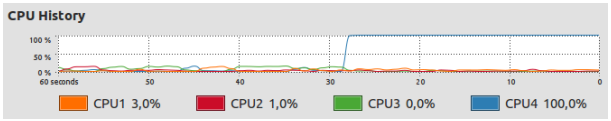
```
se364@linuxpc:~/Desktop$ mpiexec -n 1 ./matris

İşlemci Sayısı = 1

(1500 x 1500) -> İşlem Bitiş Zamanı = 45.773034 saniye
```

Şekil 2 - 1500 x 1500 (1 İşlemci)

Şekil 2'de 1500 x 1500 matris 1 işlemci ile 45.7 saniyede tamamlanmıştır.



Şekil 3 - Sistem Monitör Görüntüsü

- 2 İşlemci Testi

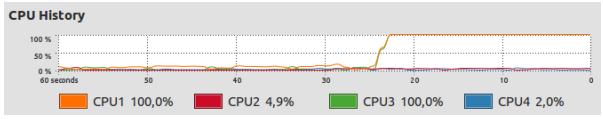
```
se364@linuxpc:~/Desktop$ mpiexec -n 2 ./matris

İşlemci Sayısı = 2

(1500 x 1500) -> İşlem Bitiş Zamanı = 30.398814 saniye
```

Şekil 4 - 1500 x 1500 (2 İşlemci)

Şekil 4'de 1500 x 1500 matris 2 işlemci ile 30.3 saniyede tamamlanmıştır.



Şekil 5 - Sistem Monitör Görüntüsü

- 3 İşlemci Testi

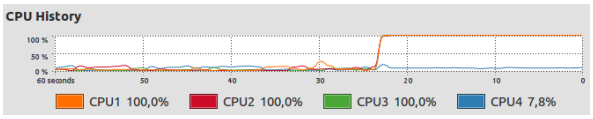
```
se364@linuxpc:~/Desktop$ mpiexec -n 3 ./matris

İşlemci Sayısı = 3

(1500 x 1500) -> İşlem Bitiş Zamanı = 25.001268 saniye
```

Şekil 6 - 1500 x 1500 (3 İşlemci)

Şekil 6'de 1500 x 1500 matris 3 işlemci ile 25 saniyede tamamlanmıştır.



Şekil 7 - Sistem Monitör Görüntüsü

- 4 İşlemci Testi

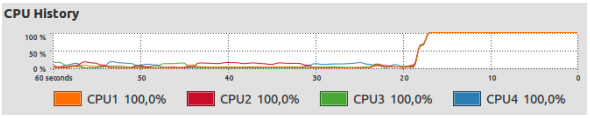
```
se364@linuxpc:~/Desktop$ mpiexec -n 4 ./matris

İşlemci Sayısı = 4

(1500 x 1500) -> İşlem Bitiş Zamanı = 26.957060 saniye
```

Şekil 8 - 1500 x 1500 (4 İşlemci)

Şekil 8'de 1500 x 1500 matris 4 işlemci ile 26 saniyede tamamlanmıştır.



Şekil 9 - Sistem Monitör Görüntüsü

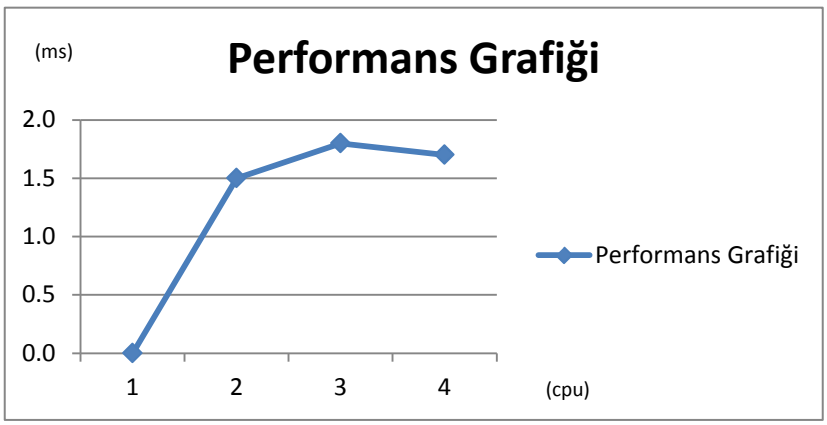
3.2 Hızlanma (Speedup) Ölçümü

$$S(p) = \frac{Tek\ işlemcide\ çalışma\ zamanı}{p\ adet\ işlemcide\ çalışma\ zamanı} = \frac{ts}{tp}$$

$$S(p) = \frac{45.7}{30.3} = 1.5$$

$$S(p) = \frac{45.7}{25} = 1.8$$

$$S(p) = \frac{45.7}{26} = 1.7$$



4. Uygulama

Uygulama kısmında C dilinde yazılan mpi matris çarpımına kodlar incelenecektir.

```c
#include"stdio.h"
#include"stdlib.h"
#include"mpi.h"

#define N 2000 // A matrisi sutun sayisi ve B Matrisi Satır Sayısı
#define X 2000 // A Matrisi Satır Sayısı
#define Y 2000 // B Sütun Sayısı

int main(int argc , char **argv)
{
    int size,rank,sum=0,i,j,k;
    int **matrix1;  //matrix1[X][N] matris tanımı - A Matrisi
    int **matrix2; //matrix2[N][Y] matris tanımı - B Matrisi
    int **mat_res; //Çarpım Matrisi mat_res[X][Y]
    double t1,t2,result;


    MPI_Status status;
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    if(rank == 0)
        printf("\nİşlemci Sayısı = %d\n", size);
```

Şekil 10 - Kod bloğu

Şekil 10'da A ve B matrisine ait satır ve sütun bilgiler ile ana fonksiyonumuzda kullanacağımız parametreler ve MPI fonksiyonları bulunmaktadır.

```c
    for(i = 0; i < X; i++)
    {
        for(j = 0; j < N; j++)
        {
            matrix1[i][j] = rand() % 10; //initialize 1 to matrix1 for all processes //rand() % 10;
        }
    }

    for(i = 0; i < N; i++)
    {
        for(j = 0; j < Y; j++)
        {
            matrix2[i][j] = rand() % 10;//initialize 2 to matrix2 for all processes //rand() % 10;
        }
    }
```

Şekil 11 - Kod bloğu

Şekil 11'de A matrisi ve B matris dizilerine 0 ile 10 arasında rastgele sayılarak seçilerek matris oluşturulur.

```c
    if(rank == 0)
        t1=MPI_Wtime();

    for(i = rank; i < X; i = i+size) //divide the task in multiple processes
    {
        for(j = 0; j < Y; j++)
        {
            sum=0;

            for(k = 0; k < N; k++)
            {
                sum = sum + matrix1[i][k] * matrix2[k][j];
            }

            mat_res[i][j] = sum;
        }
    }

    if(rank != 0)
    {
        for(i = rank; i < X; i = i+size)
            MPI_Send(&mat_res[i][0], Y, MPI_INT, 0, 10+i, MPI_COMM_WORLD);//send calculated rows to process with rank 0
    }

    if(rank == 0)
    {
        for(j = 1; j < size; j++)
        {
            for(i = j; i < X; i = i+size)
            {
                MPI_Recv(&mat_res[i][0], Y, MPI_INT, j, 10+i, MPI_COMM_WORLD, &status);//receive calculated rows from respective process
            }
        }
    }

    MPI_Barrier(MPI_COMM_WORLD);
```

Şekil 12 - Kod bloğu

Şekil 12'de A ve B matrisinin çarpım işlemi gerçekleştirilmektedir. Matris dizi değerleri hesaplanması istenen istemci sayısına göre gönderilerek hesaplanması sağlanmıştır.

```
if(rank == 0)
    t2 = MPI_Wtime();

result = t2 - t1;

if(rank == 0)
    printf("\n(%d x %d) -> İşlem Bitiş Zamanı = %f saniye\n",Y,X,result); //Bitiş Zamanı

MPI_Finalize();
return 0;
```

Şekil 13 - Kod bloğu

Şekil 13'de işlemciden gelen tüm işlem sonuçları tamamlandıktan sonra bitiş zamanı yazılır.

5. Kaynaklar

- http://www.just4tech.com/2013/10/matrix-multiplication-in-mpi.html
  - Son erişim zamanı: 20.12.2014
- www.yildiz.edu.tr/~donduran/tbb/matris.pdf
  - Son erişim zamanı: 20.12.2014