ALGORİTMA ANALİZİ-ÖDEV 3

Kaba Kod

1. Başla
2. Matris boyutunu kullanıcıdan al
3. İki n x n matris oluştur
4. Matrislerin elemanlarını kullanıcıdan al veya rastgele oluştur
5. Matris çarpımını bul
6. Matris çarpımını ekrana yazdır
7. Bitir

ALGORİTMANIN ANALİZİ

1. Özyinelemeli (Recursive) Algoritma

```
static void MultiplyMatricesRecursive(int[,] matrix1, int[,] matrix2, int[,] result,
int n, int row, int col, int idx)
{
    if (idx >= n)                                                   //n
        return;                                                      //1

    if (col >= n)                                                   //n
    {
        col = 0;                                                     //1
        row++;                                                      //1
    }

    if (row >= n)                                                   //n
        return;                                                     //1

    int sum = 0;
    for (int i = 0; i < n; i++)                                     //1+n-1+n=2n
    {
        sum += matrix1[row, i] * matrix2[i, col];                   //n * n = n^2
    }

    result[row, col] = sum;                                         //1

    MultiplyMatricesRecursive(matrix1, matrix2, result, n, row, col + 1, idx + 1);//1

}
}                                                                  //5n + n^2 + 5
```

- Bu algoritma, her adımda matrisleri daha küçük alt matrislere böler ve ardından alt matrisler üzerinde çarpım yaparak sonuçları birleştirir.
- Bu yaklaşımın zaman karmaşıklığı, Standart Matris Çarpımı Algoritmasında olduğu gibi O(n^3) olmasına rağmen, daha fazla işlem maliyeti ve bellek kullanımı gerektirir.
- Bu nedenle, genellikle pratik uygulamalarda tercih edilmez.

2. Yinelemeli (Iteratif) Algoritma

```
static void MultiplyMatrices(int[,] matrix1, int[,] matrix2, int[,] result)
    {
        int n = matrix1.GetLength(0);
        for (int i = 0; i < n; i++)                          // 1 + n + 1 + n = 2n + 2
        {
            for (int j = 0; j < n; j++)                       // 1 + n + 1 + n = 2n + 2
            {
                for (int k = 0; k < n; k++)                   // 1 + n + 1 + n = 2n + 2
                {
                    result[i, j] += matrix1[i, k] * matrix2[k, j];   // 1
                }                                                    //(2n+2)^3+1
            }
        }
    }
```

- İteratif matris çarpımı algoritması, üç katmanlı bir iç içe döngü kullanır.
- Bu algoritmanın zaman karmaşıklığı O(n^3) olarak hesaplanır, çünkü her biri n kez dönen üç iç içe döngü vardır.

ÖZET

- İki yaklaşım da doğru sonucu verecektir, ancak performans ve kod karmaşıklığı açısından iteratif yaklaşım tercih edilir.