YILDIZ TEKNİK ÜNİVERSİTESİ

ELEKTRİK-ELEKTRONİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

2020-2021 GÜZ YARIYILI

BLM3021 ALGORİTMA ANALİZİ DERSİ

GRUP-2

3.ÖDEV 1.PROBLEM RAPORU

Konu: Dinamik Programlama

Hazırlayan: Mehmet Hayri Çakır – 16011023

Dersin Yürütücüsü: Dr. Öğr. Üyesi Mehmet Amaç GÜVENSAN

İstanbul, Aralık 2020

İçindekiler

1. Yöntem

1.1. Problem

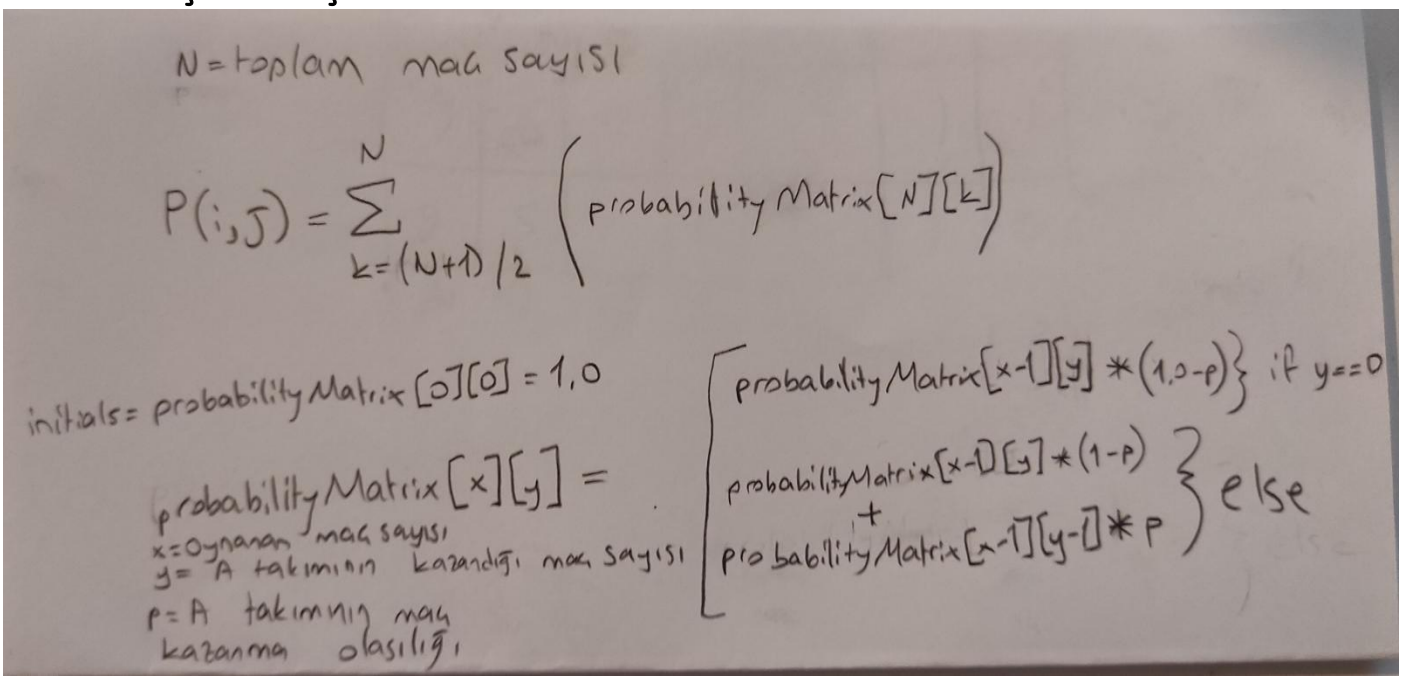
A ve B takımları, içlerinden biri n galibiyet alana kadar maç yapacaklardır. A takımının bir maçı kazanma olasılığı her maç için p, kaybetme olasılığı ise 1-p'dir. Dolayısıyla beraberlik ihtimali mevcut değildir. A takımının seriyi kazanmak için i tane daha maç kazanması, B takımının da seriyi kazanmak için j tane maç kazanması gereken durumda A'nin seriyi kazanma olasılığı P(i,j)'dir. Bu problemin Brute-Force ile çözülmesi zahmetli bir iştir.

1.2. Çözüm yolu

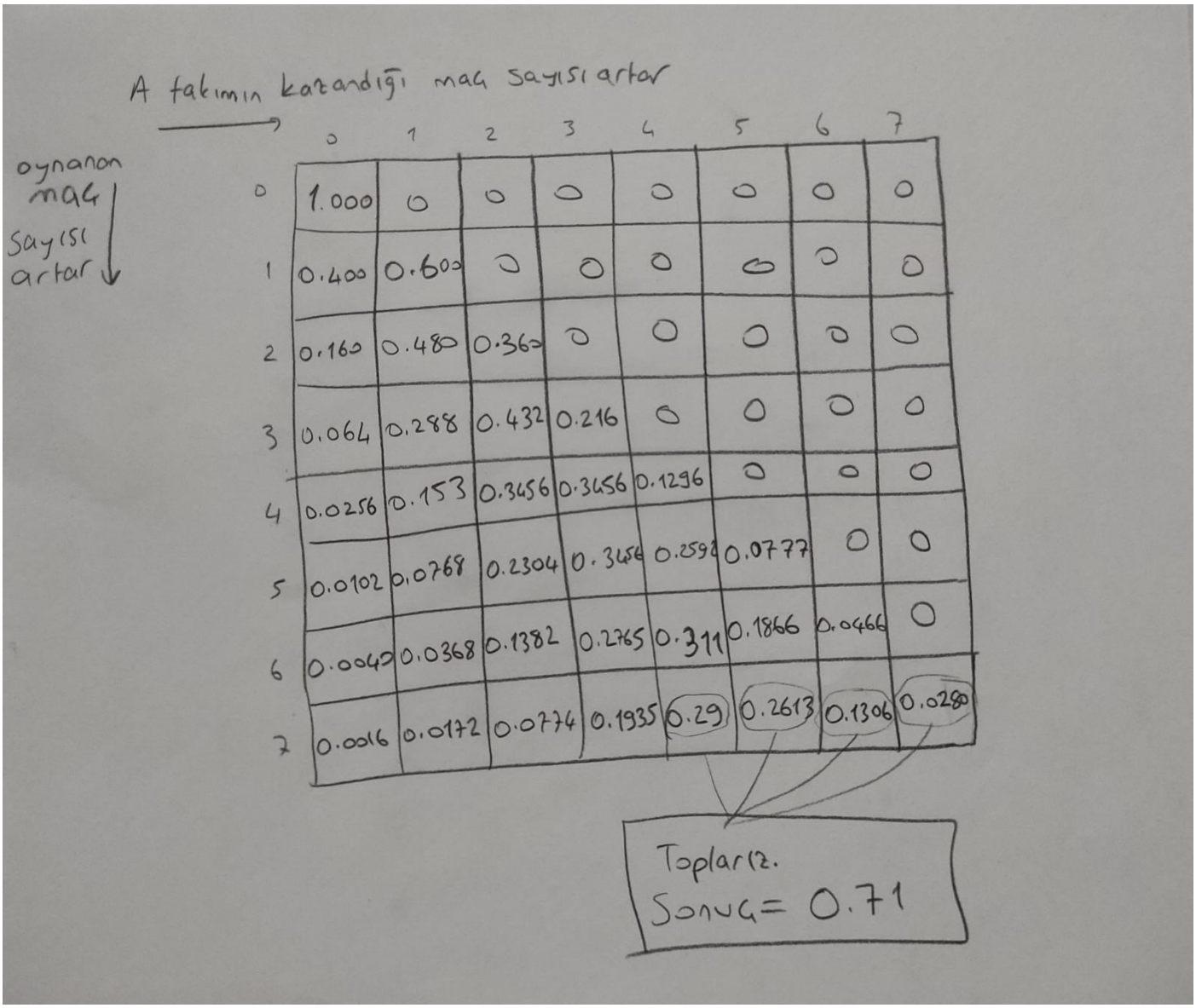
Bunun yerine bu ödevde, bu problem dinamik programlama ile çözülecektir.

2. Çözümler

2.1. A Şıkkının Çözümü



2.2. B Şıkkının Çözümü



2.3. C Şıkkının Çözümü

2.3.1. Fonksiyonlar

Main fonksiyonu hariç toplam 3 fonksiyon kullanılmıştır.

- `double P(double p, int N)`
- `double sumProbabilities(double** probabilityMatrix, int N)`
- `void printMatrix(double** probabilityMatrix, int N)`

2.3.1.1. double P(double p, int N)

Olasılıklar matrisinin tanımlandığı, matrise yer açıldığı ve matrisin içinin doldurulduğu programın en büyük ve önemli fonksiyonudur.

2.3.1.2. *double sumProbabilities(double** probabilityMatrix, int N)*

P fonksiyonu işini bitirip matrisin içini doldurduktan sonra bu fonksiyon çağrılır ve ilgili gözlerdeki değerleri toplayarak A takımının seriyi kazanması olasılığını hesaplayıp döndürür.

2.3.1.3. *void printMatrix(double** probabilityMatrix, int N)*

Matrisin içi P fonksiyonunda doldurulduktan sonra bu fonksiyon çağrılarak matris ekrana yazdırılır.

2.3.1.4. *int main()*

A takımının bir maçı kazanma olasılığı ve toplam maç sayısı kullanıcıdan girdi olarak alınır. P fonksiyonu çağrılıp sonuç ekrana yazdırılır.

2.3.2. Ekran Çıktıları

Bu bölümde programın kullanımı sırasında farklı değerlerden kaynaklı olarak ortaya çıkan farklı sonuçları gösteren ekran çıktıları eklenmiştir.

2.3.2.1. *p = 0.4 ve N = 7*

A takımının kazanma olasılığı 0.4 ve toplam maç sayısının 7 olduğu durum için ekran çıktısı aşağıdadır.

```
A takiminin kazanma olasiligini giriniz (ornek 0.4): 0.4
Toplam oynanacak mac sayisini giriniz (ornek 7): 7
_____________________________________________________________________
|1.000000|0.000000|0.000000|0.000000|0.000000|0.000000|0.000000|0.000000|
|0.600000|0.400000|0.000000|0.000000|0.000000|0.000000|0.000000|0.000000|
|0.360000|0.480000|0.160000|0.000000|0.000000|0.000000|0.000000|0.000000|
|0.216000|0.432000|0.288000|0.064000|0.000000|0.000000|0.000000|0.000000|
|0.129600|0.345600|0.345600|0.153600|0.025600|0.000000|0.000000|0.000000|
|0.077760|0.259200|0.345600|0.230400|0.076800|0.010240|0.000000|0.000000|
|0.046656|0.186624|0.311040|0.276480|0.138240|0.036864|0.004096|0.000000|
|0.027994|0.130637|0.261274|0.290304|0.193536|0.077414|0.017203|0.001638|
=====================================================================
0.289792
```

2.3.2.2. *p = 0.6 ve N = 7*

A takımının kazanma olasılığı 0.6 ve toplam maç sayısının 7 olduğu durum için ekran çıktısı aşağıdadır.

```
A takiminin kazanma olasiligini giriniz (ornek 0.4): 0.6
Toplam oynanacak mac sayisini giriniz (ornek 7): 7
_____________________________________________________________________
|1.000000|0.000000|0.000000|0.000000|0.000000|0.000000|0.000000|0.000000|
|0.400000|0.600000|0.000000|0.000000|0.000000|0.000000|0.000000|0.000000|
|0.160000|0.480000|0.360000|0.000000|0.000000|0.000000|0.000000|0.000000|
|0.064000|0.288000|0.432000|0.216000|0.000000|0.000000|0.000000|0.000000|
|0.025600|0.153600|0.345600|0.345600|0.129600|0.000000|0.000000|0.000000|
|0.010240|0.076800|0.230400|0.345600|0.259200|0.077760|0.000000|0.000000|
|0.004096|0.036864|0.138240|0.276480|0.311040|0.186624|0.046656|0.000000|
|0.001638|0.017203|0.077414|0.193536|0.290304|0.261274|0.130637|0.027994|
=====================================================================
0.710208
```

2.3.3. Kod

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


void printMatrix(double** probabilityMatrix, int N)
{
    int i;//dongu degiskeni
    int j;//dongu degiskeni
    printf("_____________________________________________________________________\n");
    //matrisi yazdirma islemi
    for (i = 0; i < N + 1; i++)
    {
```

```
        printf("|");
        for (j = 0; j < N + 1; j++)
        {
            printf("%lf|", probabilityMatrix[i][j]);
        }
        printf("\n");
    }
    printf("================================================================================\n");
}

double sumProbabilities(double** probabilityMatrix, int N)
{
    int i;//dongu degiskeni
    double result = 0.0;//A takiminin seriyi kazanma ihtimali
    //A takiminin seriyi kazanmasi icin en az oynanan mac sayisinin yarisindan bir fazla kadar mac
kazanmasi gerekir. ornegin 7 macin 4 u.
    //i degiskenini bu sayidan baslatarak toplam mac sayisina kadar ilerler ve
    //A takiminin seriyi kazanmasi ihtimalini hesaplariz.
    //ornegin A takiminin 7 macin 4'unu(i = 4), 5'ini (i=5), 6'sini, 7'sini kazanmasi ihtimallerini
toplayarak sonuca ulasiriz.
    for (i = (N + 1) / 2; i < N + 1; i++)
    {
        result += probabilityMatrix[N][i];
    }
    return result;
}

double P(double p, int N)
{
    int i;//dongu degiskeni
    int j;//dongu degiskeni
    double** probabilityMatrix = (double**)calloc(N + 1, sizeof(double*));//olasiliklar matrisi
    //matrisin satir indeksi oynanan mac sayisini, sutun indeksi de A takiminin kazandigi mac
sayisini belirtiyor.
    //ornegin probabilityMatrix[1][1] = 1 mac oynanip o maci da A takiminin kazanmasi ihtimalidir.
    //ornek 2: probabilityMatrix[7][4] = oynanan 7 macin 4 unu A takimi, 3 unu B takimi kazanmistir.
    //bu sebeple; hesaplamalarimiz bittikten sonra probabilityMatrix[N][(n+1)/2 + 1] den baslayarak
probabilityMatrix[n][n] e kadar
    //olan degerleri toplayarak A takiminin seriyi kazanma olasiligini hesaplamis oluruz.
    //matrisin sag ust tarafini (sutun indeksinin satir indeksinden fazla oldugu durum) 0 ile
doldurdum A takiminin kazandigi mac sayisi
    //toplam mac sayisindan fazla olamaz.

    for (i = 0; i < N + 1; i++)
    {
        probabilityMatrix[i] = (double*)calloc(N + 1, sizeof(double));
    }
    probabilityMatrix[0][0] = 1.0;

    //i her adimda degiserek toplam oynanan mac sayisini temsil ediyor.
    for (i = 1; i < N + 1; i++)
    {
        //j A takiminin kazandigi mac sayisini temsil ediyor.
        for (j = 0; j < i + 1; j++)
        {
            //A takimi i kadar macta hic mac kazanamamasi durumunda bu if'e girilir.
            //ayni kolonda ust satirdaki degeri tasiyip B takiminin kazanmasi ihtimaliyle carpariz.
            if (j == 0)
            {
                probabilityMatrix[i][j] = probabilityMatrix[i - 1][j] * (1.0 - p);
            }
            else
            {
                probabilityMatrix[i][j] = probabilityMatrix[i - 1][j] * (1.0 - p) +
probabilityMatrix[i - 1][j - 1] * p;
            }
```

```
        }
    }
    printMatrix(probabilityMatrix, N);
    return sumProbabilities(probabilityMatrix, N);
}

int main()
{
    // A takiminin bir maci kazanma olasiligi
    double p;
    //toplam mac sayisi
    int n;

    printf("A takiminin kazanma olasiligini giriniz (ornek 0.4): ");
    scanf("%lf", &p);
    printf("Toplam oynanacak mac sayisini giriniz (ornek 7): ");
    scanf("%d", &n);

    printf("%f\n", P(p, n));
    return 0;
}
```