Samurai Sudoku Multithread ile Çözme

Feridun Suay Bayar-Egehan Demir

Bilgisayar Mühendisliği Kocaeli Üniversitesi

170202064@kocaeli.edu.tr- 180202040@kocaeli.edu.tr

Özet- Txt dosyasından okuyarak oluşturduğumuz samuray sudokunu multithread ile çözdürme ve çözüm sonuçlarını grafik olarak yazdırma.

Anahtar Kelimeler – C# , Multithread , Dosya okuma, Matris, Chart, Matrix, Math.

I. GİRİŞ

Dosyadan okuduğumuz sudoku verilerini alıp, bir arayüz oluşturup multithread ile çözüme ulaştırmak ve zamana bağlı olarak çözülmüş olan kutu sayısını chart ile grafik haline getirmek.

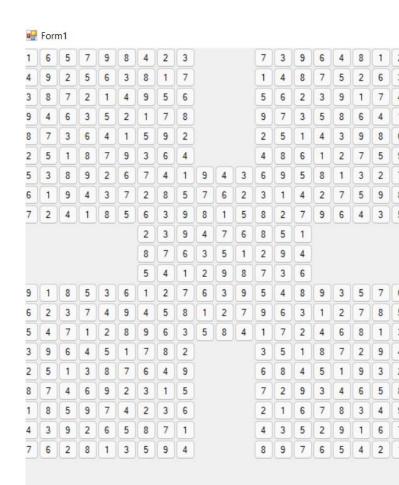
II. YÖNTEM

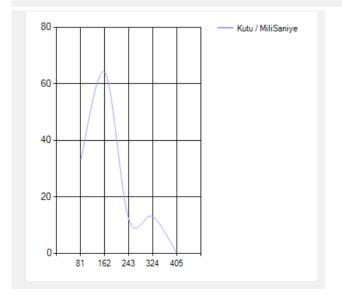
Program C# ve onun üzerinden Multithread yapısını kullanarak geliştirilmiştir. Ayrıca chart toolu ile grafik oluşturulmuştur.

III. DENEYSEL SONUÇLAR

Sudokunun çözülmüş hali görsellerdedir. İşlemcinin hızına göre grafik oluşturulmuştur.

Burada ise programın milisaniye cinsinden grafiği verilmiştir. 5 adet 9*9'luk matrisler için toplam kutu sayısı x ekseninde, milisaniye ise y ekseninde verilmiştir.





IV. YALANCI KOD 1.BAŞLA

2.Butonları "array" olarak tanımla 3.Yazdir fonksiyonu için buton1 e atama yap 4. for(i = 0; i < k; i++) for(j = 0; j < k; j++) yap map[i,j] yi buton1 textine eşitle 5.BİTİR

6.Yazdir2 metodunda buton2 için aynısını yap

7. Yazdir3 metodunda buton3 için aynısını yap

8. Yazdir4 metodunda buton4 için aynısını yap

9. Yazdir 5 metodunda buton 5 için aynısını yap

10.fonksiyon1 metoduna BAŞLA

11.p <- board.GetLenght(0)

12.saniye <- DateTime.Now.Millisecond

13.if(KontrolET(board,p)=TRUE)

then(tablo[0]<-DateTime.Now.Millisecond-saniye,

board ve uzunluğunu yazdır(yazdir(board,p))

14.BİTİR

15.fonksiyon3 metodunda tablo[1] için aynısını yap 16.fonksiyon4 metodunda tablo[2] için aynısını yap 17.fonksiyon5 metodunda tablo[3] için aynısını yap 18.fonksiyon6 metodunda tablo[4] için aynısını yap

19.BAŞLA

20.for(i = 0; i < harita.GetLenght(0); i++)

for(j = 0 ; j < harita.GetLenght(0) ; j++)

aranan sayı için mevcut satırı kontrol et(if(harita[satır,i] eşitse num)

then(return false))

21.for(i = 0; i < harita.GetLenght(0); i++)

for(j = 0; j < harita.GetLenght(0); j++)

aranan sayı için mevcut sütunu kontrol et(if(harita[i,sutun] eşitse num)

then(return false))

22.karekök <- (int)Math.Sqrt(harita.GetLength(0))

23.satirBasla <- satır - satır % karekök

24.sutunBasla <- sütun - sütun % karekök

25.Aranan sayının bulunan küçük 3'lü matriste olup olmadığını kontrol et

(for(int = satirbasla; i<satirbasla+karekök; i++)

for(int = sutunbasla; i<sutunbasla+karekök; i++)

if(harita[i,j] eşitse num)

then(return false)

)

26.Eğer sayi her aşamadan false ile geri dönderilmezse aranan sayi odur.(return true)

(BACTRACKING ALGORITHM)

(GERİ İZLEME ALGORİTMASI)

27.BİTİR

28.Form Yükle metoduna BAŞLA

29.dosya yolu<-oku.txt

30.Dosyadan okumak için FileStream fs nesnesi oluştur.(FileStream fs <- new FileStream(dosya_yolu, FileMode.Open, FileAccess.Read))

31.StreamReader sınıfından sw nesnesini üret ve fs yi ata(StreamReader sw <- new StreamReader(fs))

32.while(yazı!=NULL)

do(metin <- metin + yazi,

yazi <- sw.ReadLine())

33.okuma bittikten sonra kapat(sw.Close(),fs.Close())

34. 268.-331. satır aralarında winformdaki sodukuları oluştur

(

for(i = 0; i < 9; i++)

for(j = 0; j < 9; j++)

Buton Dizi1 oluştur

Buton_Dizi1 text

Buton Dizi1 location

Buton_Dizi1 Size ayarla

35. Aynı işlemleri Buton Dizi2 için yap

36. Aynı işlemleri Buton Dizi4 için yap

37. Aynı işlemleri Buton Dizi5 için yap

38. Aynı işlemleri Buton_Dizi3 için yap

39. 333. ve 792. satırlar arasında okunan txt dosyasını

Buton_Dizi1, Buton_Dizi2, Buton_Dizi3, Buton_Dizi4 ve Buton Dizi5 elemanlarına tek tek ata

(Buton Dizi1[0, 0].Text = "" + metin[0];

Buton_Dizi1[0, 1].Text = "" + metin[1];

```
Buton_Dizi1[0, 2].Text = "" + metin[2];
        Buton Dizi1[0, 3].Text = "" + metin[3];
        Buton Dizi1[0, 4].Text = "" + metin[4];
        Buton_Dizi1[0, 5].Text = "" + metin[5];
        Buton_Dizi1[0, 6].Text = "" + metin[6];
        Buton Dizi1[0, 7].Text = "" + metin[7];
        Buton Dizi1[0, 8].Text = "" + metin[8];
        Buton Dizi2[0, 0].Text = "" + metin[9];
        Buton Dizi2[0, 1].Text = "" + metin[10];
        Buton_Dizi2[0, 2].Text = "" + metin[11];
        Buton Dizi2[0, 3].Text = "" + metin[12];
        Buton Dizi2[0, 4].Text = "" + metin[13];
        Buton_Dizi2[0, 5].Text = "" + metin[14];
        Buton Dizi2[0, 6].Text = "" + metin[15];
        Buton Dizi2[0, 7].Text = "" + metin[16];
        Buton_Dizi2[0, 8].Text = "" + metin[17];
        Buton_Dizi1[1, 0].Text = "" + metin[18];
        Buton Dizi1[1, 1]. Text = "" + metin[19];
        Buton Dizi1[1, 2]. Text = "" + metin[20];
        Buton_Dizi1[1, 3].Text = "" + metin[21];
        Buton Dizi1[1, 4]. Text = "" + metin[22];
        Buton Dizi1[1, 5]. Text = "" + metin[23];
        Buton Dizi1[1, 6].Text = "" + metin[24];
        Buton_Dizi1[1, 7].Text = "" + metin[25];
        Buton Dizi1[1, 8]. Text = "" + metin[26];
                   40. BAŞLA
           41."*" olan yerleri "0" yap
             (for(i = 0; i < 9; i ++)
                for(j \ 0 \ 0 \ ; j < 9 \ ; j++)
         if(Buton Dizi1[i,j].Text eşitse ("*"))
           then(Buton Dizi1[i,j] eşittir "0")
42. Aynı "0" yapma işlemini Buton Dizi2 için yap
43. Aynı "0" yapma işlemini Buton Dizi4 için yap
```

44. Aynı "0" yapma işlemini Buton Dizi5 için yap

45. Aynı "0" yapma işlemini Buton Dizi3 için yap (Sırasıyla) 46.Thread1 oluştur 47.new Thread(new ThreadStart(() => fonksiyon1(matris1))) 48.Thread1 BAŞLA 49. Thread 2 oluştur 50.new Thread(new ThreadStart(() => fonksiyon1(matris2))) Thread2 BAŞLA 51. Thread3 olustur 52.new Thread(new ThreadStart(() => fonksiyon1(matris3))) Thread3 BAŞLA 53. Thread4 oluştur 54.new Thread(new ThreadStart(() => fonksiyon1(matris4))) Thread4 BAŞLA 55.Thread5 oluştur 56.new Thread(new ThreadStart(() => fonksiyon1(matris5))) Thread5 BAŞLA 57. Threadleri uyut 58.tablo[0],tablo[1],tablo[2],tablo[3],tablo[4] add points in tablo[0],tablo[1],tablo[2],tablo[3],tablo[4] into graph V. SORUNLAR VE EKSİKLER

Projemizin çözümü bir nokta üzerinden değil tüm noktalar üzerindendir. Bu nedenle 10 thread yerine 5 thread kullanılmıştır.

VI. SONUÇ

Projeyi geliştirirken algoritma yazma ve thread konularında kendimizi geliştirdik. Ayrıca recursive fonksiyon yapısını yeniden öğrenmemiz ve tekrardan çalışmamız gerekti.

VII. KAYNAKÇA

HTTPS://WWW.UDEMY.COM/COURSE/SİFİRDAN-İLERİ-SEVİYE-CSHARP-PROGRAMLAMA/LEARN/LECTURE/8388226?START=15#OVERVİ

PROGRAMLAMA/LEARN/LECTURE/8388226?START=15#OVERV EW

HTTP://WWW.CSHARPNEDİR.COM/ARTİCLES/READ/?İD=163

HTTPS://WWW.GEEKSFORGEEKS.ORG/MULTİTHREADİNG-C-2/

HTTPS://WWW.OYUNBİLİM.COM/ZEKA/SUDOKU/