

## İÇİNDEKİLER

Seim Sistemi Paralel Algoritmasının Programlama Dili ile Kodlanarak rneklenmesi .....	2
Tanım .....	2
Ama .....	2
Kısıtlar .....	2
Referanslar ve Yararlanılan Kaynaklar .....	2
Algoritma Adımları .....	3
Kaynak Kod .....	4
Kaynak Kodun Derlenmesi .....	4
Gzlem ıktılarının Elde Edilmesi .....	5
Gzlem Sonuları .....	6
Gzlem Ortamı .....	6

## SEİM SİSTEMİ PARALEL ALGORİTMASININ PROGRAMLAMA DİLİ İLE KODLANARAK RNEKLENMESİ

### TANIM

Seimlerde sandıkların ilerindeki oyların eř zamanlı olarak sayılması ve genel sayım sonucun daha kısa srelerde bulunması gerek hayatta uygulanan bir paralelleřtirme yaklařımıdır. Bize bu rneęi vererek paralel algoritmaları aıklayan Prof.Dr. Mahir DURSUN hocamıza teřekkr bor bilirim.

### AMA

Seimlerde ki oy sayım iřlemlerine benzeyen paralel ve sıralı algoritmalar oluřturularak bu yntemler arasında ki bařarımın ve performansın kıyaslanması.

### KISITLAR

Bu dokmanı okuyanların temel dzeyde C# programlama dilini bildięi varsayılır. Dokman ierisinde .net core 3.1 SDK'sı kullanılarak bir komut satırı (Console) uygulaması yapılacaktır.

### REFERANSLAR VE YARARLANILAN KAYNAKLAR

Prof.Dr. Mahir DURSUN	Gazi niversitesi Biliřim Enstits
Task.Run	<a href="https://docs.microsoft.com/tr-tr/dotnet/api/system.threading.tasks.task.run?view=netcore-3.1">https://docs.microsoft.com/tr-tr/dotnet/api/system.threading.tasks.task.run?view=netcore-3.1</a>
ConcurrentBag<>	<a href="https://docs.microsoft.com/tr-tr/dotnet/api/system.collections.concurrent.concurrentbag-1?view=netcore-3.1">https://docs.microsoft.com/tr-tr/dotnet/api/system.collections.concurrent.concurrentbag-1?view=netcore-3.1</a>
.Net Core 3.1	<a href="https://docs.microsoft.com/tr-tr/dotnet/?view=netcore-3.1">https://docs.microsoft.com/tr-tr/dotnet/?view=netcore-3.1</a>
.Net 3.1 SDK	<a href="https://dotnet.microsoft.com/download/dotnet-core/3.1">https://dotnet.microsoft.com/download/dotnet-core/3.1</a>
GitHub	<a href="https://github.com/">https://github.com/</a>

#### ALGORİTMA ADIMLARI

- Oy adında aşığıdaki elemanlara sahip bir nesne olsun
  - Id
  - İl
  - İle
  - Mahalle
  - Okul
  - Sandık
  - Parti (“A partisi”, “B partisi”)
- İller, İleler, Mahalleler, Okullar, Sandıklar şeklinde 5 adet ierisinde rastgele verilerin olduėu diziler tanımlayalım.
- Bu dizileri kullanarak dizi sayısı + her sandıkta olacak oy sayısı kadar oylar adında bir dizi oluřturup ierisine rastgele bu dizilerdeki verileri dolduracaėız.
- Yukarıdaki iřlemi 3 ayrı algoritma kullanarak kodlayacaėız.
  - Sıralı birbirini bekleyen iřler şeklinde
  - İlleri gevlere (task) blp her grev altında sıralı olarak dngler kurup oy dizisini il sayısı kadar grevde eř zamanlı doldurarak.
  - Tm dngleri Paralel ktphanesinin For metodu ile dngleri ile kurup iřlemlerin tamamının paralel yapılmasını saėlayarak
- Rast gele veriler 3 ayrı yntem ile oluřturulup her bir yntem iin geen sre hesaplanıp ekrana geen sre ile birlikte oluřturulan kayıt sayısı yazılarak bu 3 algoritma arasındaki en verimli olanın gzlenmesi.

## KAYNAK KOD

Bu deneyin kaynak kodları kiřisel GitHub hesabımda Public ve MIT lisansı ile saklanmaktadır. Repoya ařağıdaki baėlantı adresinden ulaşabilirsiniz.

Paralel Algoritmalar kök deposu

<https://github.com/ahmetaltay33/parallel-algorithms>

Bu deneyin deposu

<https://github.com/ahmetaltay33/parallel-algorithms/tree/master/dotnetcore/election-sample>

## KAYNAK KODUN DERLENMESİ

Bilgisayarınızda .net core 3.1 SDK sının yüklü olması gerekmektedir. Bilgisayarınızda Visual Studio 2019 un güncel versiyonu bulunuyor ve yükleme sırasında .net core geliştirme ortamını işaretledi iseniz SDK hali hazırda yüklü demektir.

SDK yüklü olup olmadığını veya yüklü olan sürümü bulmak için komut satırına ařağıdaki kodu yazıp test edebilirsiniz.

**dotnet --version**

.Net Core SDK 3.1 yüklemek için [referanslarda](#) bulunan baėlantıdan bilgisayarınıza uygun sürümü indirip yükleyebilirsiniz.

Kaynak kodları bilgisayarınıza indirip Visual Studio içerisinde F5 komutu ile çalıştırabilirsiniz.

Geliştirme ortamı kullanmadan komut satırından ařağıdaki komutlar ile kaynak kodu derleyebilirsiniz.

**dotnet build D:\election-sample\election-sample.sln**

Derlenmiş uygulama proje klasörü içerisinde **.\bin\debug\netcoreapp3.1 \election-sample.exe** konumunda olacaktır.

Derleme ile ilgili problem yařanır ise kaynak kodların derlenmiş hali GitHub da ařağıdaki url de bulunmaktadır.

<https://github.com/ahmetaltay33/parallel-algorithms/blob/master/dotnetcore/election-sample/output/election-sample.zip>

## GÖZLEM ÇIKTILARININ ELDE EDİLMESİ

Program kaynak kodları derlendikten sonra election\_sample.exe adında bir konsol uygulaması elde edilir. Program dışarıdan 2 adet parametre almaktadır.

**İlk parametre:** her sandık için oluşturulacak oy sayısıdır.

**İkinci parametre:** her oy oluşturma işlemi için geçecek olan işlem süresidir. Gerçek ortamda bu süre veri tabanına yazma ya da bir web servise veri gönderme işleminde geçecek zaman olarak değerlendirilir.

Program aşağıda verilen parametreler ile çalıştırılmış ve sonuçları ekran görüntüsü alınarak eklenmiştir.

Çalıştırılan Komut: election-sample.exe 5 3

```
PS D:\Github\ahmetaltay33\parallel-algorithms\dotnetcore\election-sample> .\bin\Release\netcoreapp3.1\election-sample.exe 5 1
Veriler sıralı algoritma ile oluşturuluyor... (Paralel yok, sıralı)
Veriler oluşturuldu. Kayıt sayısı: 39375 Geçen süre ms: 519710

Task kullanılarak sadece illerin paralelleştirildiği yöntem ile veriler oluşturuluyor... (Sadece iller paralel)
Veriler oluşturuldu. Kayıt sayısı: 39375 Geçen süre ms: 87995

Tüm döngüler Parallel.For ile çalıştırılarak veriler oluşturuluyor... (Tüm döngüler paralel)
Veriler oluşturuldu. Kayıt sayısı: 39375 Geçen süre ms: 49110
```

Çalıştırılan Komut: election-sample.exe 100 0

```
PS D:\Github\ahmetaltay33\parallel-algorithms\dotnetcore\election-sample> .\bin\Release\netcoreapp3.1\election-sample.exe 100 0
Veriler sıralı algoritma ile oluşturuluyor... (Paralel yok, sıralı)
Veriler oluşturuldu. Kayıt sayısı: 787500 Geçen süre ms: 2909

Task kullanılarak sadece illerin paralelleştirildiği yöntem ile veriler oluşturuluyor... (Sadece iller paralel)
Veriler oluşturuldu. Kayıt sayısı: 787500 Geçen süre ms: 1270

Tüm döngüler Parallel.For ile çalıştırılarak veriler oluşturuluyor... (Tüm döngüler paralel)
Veriler oluşturuldu. Kayıt sayısı: 787500 Geçen süre ms: 1531
```

Çalıştırılan Komut: election-sample.exe 1000 0

```
PS D:\Github\ahmetaltay33\parallel-algorithms\dotnetcore\election-sample> .\bin\Release\netcoreapp3.1\election-sample.exe 1000 0
Veriler sıralı algoritma ile oluşturuluyor... (Paralel yok, sıralı)
Veriler oluşturuldu. Kayıt sayısı: 7875000 Geçen süre ms: 31690

Task kullanılarak sadece illerin paralelleştirildiği yöntem ile veriler oluşturuluyor... (Sadece iller paralel)
Veriler oluşturuldu. Kayıt sayısı: 7875000 Geçen süre ms: 13205

Tüm döngüler Parallel.For ile çalıştırılarak veriler oluşturuluyor... (Tüm döngüler paralel)
Veriler oluşturuldu. Kayıt sayısı: 7875000 Geçen süre ms: 17379
```

Çalıştırılan Komut: election-sample.exe 1 10

```
PS D:\Github\ahmetaltay33\parallel-algorithms\dotnetcore\election-sample> .\bin\Release\netcoreapp3.1\election-sample.exe 1 10
Veriler sıralı algoritma ile oluşturuluyor... (Paralel yok, sıralı)
Veriler oluşturuldu. Kayıt sayısı: 7875 Geçen süre ms: 115489

Task kullanılarak sadece illerin paralelleştirildiği yöntem ile veriler oluşturuluyor... (Sadece iller paralel)
Veriler oluşturuldu. Kayıt sayısı: 7875 Geçen süre ms: 17588

Tüm döngüler Parallel.For ile çalıştırılarak veriler oluşturuluyor... (Tüm döngüler paralel)
Veriler oluşturuldu. Kayıt sayısı: 7875 Geçen süre ms: 12519
```

## GZLEM SONULARI

Gzlem ıktıları incelendiğinde her bir iřlem iin geen zamanın artırılması paralel olan iřlemlerin daha kısa srede iři bitirmesine sebep oluyor. iřlem sayısının artması bir iřlem iin geen srenin sıfır olması yada sıfıra yaklařması durumlarında paralel iřlemlerin ok fazla etkisinin olmadığı gzlemleniyor.

Sonuç olarak dngsel bir paralel hesaplamada toplam geen zamana olan etki; dngnn eleman sayısından ok, dngnn her bir turda harcadığı zamana baėlıdır.

## GZLEM ORTAMI

Bu dokmanda bahsi geen algoritmanın gzlemi ařaėıdaki donanım zelliklerine sahip bir ortamda yapılmıřtır.

- Intel Core i7 1. Nesil 1.7 Ghz Turbo ile max 2.4 Ghz 4 ekirdekli 8 kanallı iřlemci (740QM)
- DDR3 1333 Mhz Clock 9 16 GB Ram
- SSD disk
- Paylařımsız ekran kartı
- Diz st bilgisayar