Karadeniz Teknik Üniversitesi

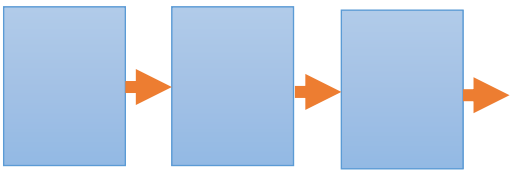
Paralel Bilgisayarlar/Paralel Hesaplama/Multicore Systems and Programming

Dönem Projesi

Ibrahim SAVRAN

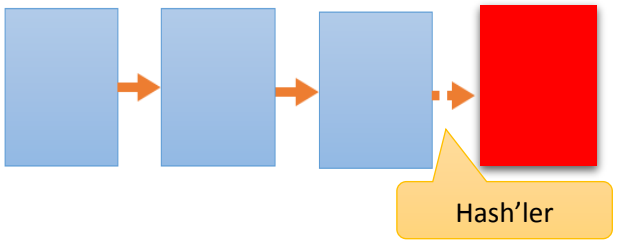
Blok Zinciri (Block Chain)

Veri yapıları dersindeki dinamik veri yapılarından "Tek bağlı listeleri" hatırlayalım. Şekil 1'de görüldüğü gibi, bilgiler blok içine (kayıt veri yapısı) kaydedilir, yeni bir kayıt eklenmek istendiğinde bellekten yer ayrılır ve yeni kayıt yapılır. Yeni blok, kayıt zincirindeki son bloğa bağlanır. Böylece yeni blok artık son blok olur.



Şekil 1: Tek bağlı liste veri yapısı

Blok zincirinde ise kayıtlar işlemleri(para transferi/sözleşmeleri), bağlantılar ise hesaplanan linklerdir (hash değerleridir). Bir önceki bağlantı linkini blok içindeki bilgiyle harmanlayarak bir değer üretilir.



Şekil 2: Yeni eklenen blok

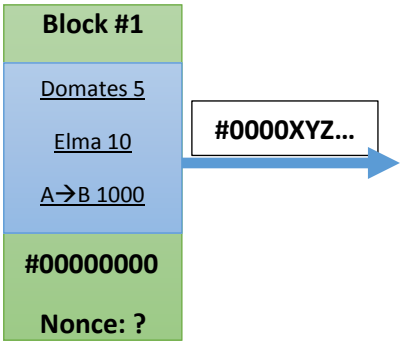
Aşağıda basit bir Python kod parçasında sha256 hash fonksiyonu kullanılmaktadır. Dikkat edilirse farklı 2 değer için birbirinden çok farklı hash değerleri üretilmektedir.

Sha256 fonksiyonu verilen veriyi işleyerek 256bit (16'lık tabanda 64 basamaklı) sayı üretir.

```
from hashlib import sha256
text = "ABC"
print (sha256(text.encode("ascii")).hexdigest())          # b5d4045c3…
text = "ABD"
print (sha256(text.encode("ascii")).hexdigest())          # 69fefb8cb1…
```

Blok Zincirindeki bir blok içinde sadece işlemler/kayıtlar (transactions) yoktur. Blok içinde Blok numarası, kayıtlar, önceki hash numarası ve nonce bulunmaktadır. Yeni hesaplanacak Hash numarası için ise kısıtlama getirilmektedir. Mesela **#0000XYZ… kısıtı** ilk 4 rakamın 0 olması gerektiğini ifade edebiliriz. Verilen bilgiler ışığında iteratif olarak farklı nonce değerlerini deneyerek kısıta uyumlu hash değeri üretebilen ilk kişi/ekip madeni çıkarmış demektir.

Aşağıdaki 1. Blok verilmiş olsun. Bu bloktaki bütün bilgileri ulayarak ilk 4 hanesi 0 olan ve daha önceden üretilen hash kodlarından farklı ilk hash'i hesaplayan bilgisayar sisteme kaydetme işlemini yapmalıdır.



Blokzincir teknolojisinin temel özelliklerinden biriside onaylama işlemidir.

Blok zincirinde (para/coin) transfer işlemlerinin yanında ayrıntılı açıklamalar da tutulabilir.

Bütün verileri girdikten sonra **nonce** değerini hesapladık.

Buldum: 53926

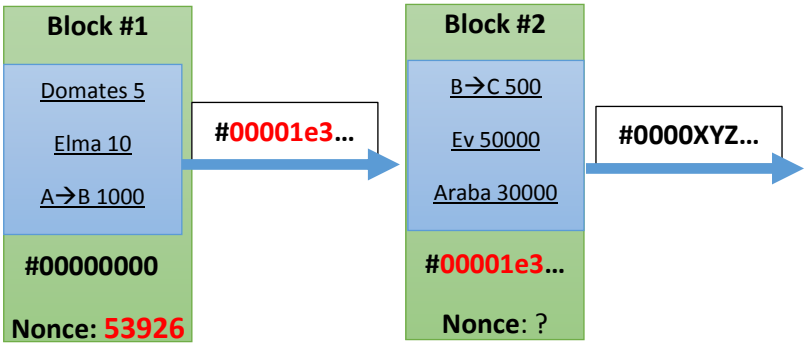
İşlem Tamamlandı: 0.4013 sn

00001e324b1a5e132e08bcd5cbf3001d5f173efc1b023bcccd18abff7901f0ab

Bulunan hash ile sonraki bloğa bağlantı yapılmış olur. Bu hash bir sonraki bloktaki kayıtlarla harmanlanarak yeni hash hesabında kullanılacaktır.

Şimdi ilk madencilik işlemi tamamlandı. Bundan sonraki aşamada hesaplanan "**nonce**" değeri ile birlikte blok bütün bilgisayarlardaki/threadlerdeki veri tabanlarına kaydedilmelidir. Burada hız önemli bir faktördür. Hesaplanan hash hızlıca ağdaki makinelerin %50+1 'ine kaydedilmelidir.

Burada asıl zorluk, kısıtlamaya uygun olarak ilk tahminin yapılmasıdır. Mesela kısıta uygun hesaplanan hash değeri önceden kullanılmış olabilir.



Proje içeriği:

Projede ilk blok aşağıda verilmiştir. İlk blokta verilen bilgileri inceleyelim:

Blok numarası: 1

Mesaj: “Bize her yer Trabzon!

Bölümün en yakışıklı hocası İbrahim Hoca’dır”

Hash: 0000000000000000000000000000000000000000000000000000000000000000

Nonce: hesaplanması gereken değer

1
Bize her yer Trabzon! Bölümün en yakışıklı hocası İbrahim Hoca’dır
00000000000000000000000000000000
Nonce: ?

Bu durumda ilk text bilgisi aşağıda verilmiştir.

text = “1Bize her yer Trabzon! Bölümün en yakışıklı hocası İbrahim Hoca’dır00000000000000000000000000000000000000000000000000000000000000000” + nonce"

Kısıtlaması olarak tek “0” içeren hash’in hesaplanması istensin. Yani hesaplanacak olan hash 0XYZ… formatına uymalıdır.

Farzedelim ki hesaplama tamamlandı ve sırasıyla “**nonce**” ve “**hash**” 1453, 0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef olsun. Bu durumda ikinci bloğun bilgisi şu şekilde olacaktır. Bu sefer ilk iki rakamı 00XYZ… olacak şekilde hash bulmalısınız.

text = “21Bize her yer Trabzon! Bölümün en yakışıklı hocası İbrahim Hoca’dır000000000000000000000000000000000000000000000000000000000000000014530123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef” + nonce"

İstenilenler:

1- (50p) Proje en az iki thread/process ile çalıştırmalıdır.
   a. Hash hesaplarken, ilk adımda bir tane "0" ile başlayan; ve her adımda bir 0 ekleyerek hash bulan blok zincir uygulamasını yazınız.
   b. 2/4/8/16 thread/process için çalıştırıp performans karşılaştırmalarını grafiklerle açıklayın.
   c. Belirtilen kriterlere göre maksimum sayıda hesaplanan "nonce" ve "hash" değerlerini zaman ölçümleri ile birlikte elde edin.
   d. Maksimum 5 saat çalışma süresinde kaç tane "nonce" ve "hash" hesapladınız?
2- (50p) **Rapor**
   a. İçinde ögrencino, ad ve soyad bilgileri bulunmalıdır.
   b. Dosya pdf formunda olmalıdır, adının formatı: **öğrencino_ad_soyad.pdf**
   c. Projede hesaplanan "nonce" ve "hash" değerlerinden ilk 15 tanesini liste halinde vermelisiniz.
   d. Blok zincir hesaplamada thread/processler arasında senkronizasyonunu nasıl gerçekleştirdiğinizi ayrıntılı bir şekilde açıklayınız.
3- (+15p) **Video hazırlarsanız**
   a. Video minimum 5dk olmalı,
   b. **Video'yu online drive'a veya youtube/vimeo'ya yükleyiniz**,
   c. Video moodle'a **yüklenmemelidir**,
   d. **Video linkini rapor içinde paylaşmalısınız**,
   e. Video'yu online drive'lara yüklerseniz **erişim hakkı** vermelisiniz.
4- **Kod** ve **raporu** ZIPleyip tek dosya halinde Moodle sistemine yükleyiniz.

Notlandırma:

Projeyi puanlama seçeneği aşağıda verilmiştir.

Python %15, Java %20, OpenMP/Pthread %30, MPI %40, CUDA %40, OpenCL %45.

Kısaca projeyi Pythonda tamamlarsanız ve 100 alırsanız finale etkisi 15 puan olarak hesaplanacaktır.

Kaynaklar:

1- (Yusuf Hoca'nın Doktora Tezi izleme raporu) Moodle'da ulaşabilirsiniz.
2- (Yusuf Hoca'nın önerdiği site) https://cointelegraph.com/explained/proof-of-work-explained#:~:text=Proof%2Dof%2DWork%2C%20or,the%20network%20and%20get%20rewarded
3- Python ile blockchain uygulamasını izleyin: https://www.youtube.com/watch?v=ZhnJ1bkIWWk
4- https://www.youtube.com/watch?v=malwhCwEosk
   Code: https://github.com/nathan-149/CustomCrypocurrency/blob/master/gymcoin/blockchain.py

```
# -*- coding: utf-8 -*-
"""
Created on Wed Apr  7 00:54:11 2021
Bu kod aşağıdaki kaynaktan alınarak kendi örneğimize uyarlanmıştır.

@author: codebasics
Kaynak:
https://github.com/codebasics/cool_python_apps/blob/main/2_bitcoin_mining/bitcoin_mining.py
"""

from hashlib import sha256
MAX_NONCE = 100000000

def SHA256(text):
    return sha256(text.encode("ascii")).hexdigest()

def mine(block_number, transactions, previous_hash, prefix_zeros):
    prefix_str = '0'*prefix_zeros
    for nonce in range(MAX_NONCE):
        text = str(block_number) +  transactions + \
            previous_hash + str(nonce)
        new_hash = SHA256(text)
        if new_hash.startswith(prefix_str):
            print(f"Buldum : {nonce}")
            return new_hash

    raise BaseException(f"{MAX_NONCE} kadar denedim bulamadım")
```

```
if __name__=='__main__':

    Block_no = 1
    transactions='''
        Domates 5
        Elma 10
        A->B 1000
        '''
    difficulty = 4 # ilk 4 hane 0 olacak
    import time
    start = time.time()
    print("start mining")
    # ilk
    old_hash = '0000000000000000000000000000000000000000000000000000000000000000'
    new_hash = mine(1, transactions, old_hash, difficulty)
    total_time = str((time.time() - start))
    print(f"İşlem Tamamlandı : {total_time} sn")
    print(new_hash)
############
```