



UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
KATEDRA ZA PRIMENJENE RAČUNARSKE NAUKE

Paralelni i distribuirani algoritmi i strukture podataka

prof. dr Dušan Gajić

Zimski semestar 2025/2026.

Studijski program: Računarstvo i automatika

Modul: Računarstvo visokih performansi

O predmetu

Pregled

- Predavači
- Cilj i sadržaj predmeta
- Organizacija ispita
- Tehnologije
 - Programski jezik Go
 - Blokčejn tehnologije

Literatura i resursi

Predavači



Nastavnik:

prof. dr Dušan Gajić

E-mail: dusan.gajic@uns.ac.rs

Kancelarija: NTP-330

Konsultacije: dogovor putem mejla

Asistent:

Nebojša Horvat

E-mail: horva.n@uns.ac.rs

Kancelarija: NTP-328

Konsultacije: dogovor putem mejla

Za uništavanje jedne nacije nisu neophodne atomske bombe i projektili dugog dometa.

Dovoljni su snižavanje kvaliteta obrazovanja i dozvoljeno varanje na ispitima.

Pacijenti umiru u rukama doktora koji su položili ispite varajući.

Zgrade se urušavaju u rukama inženjera koji su položili ispite varajući.

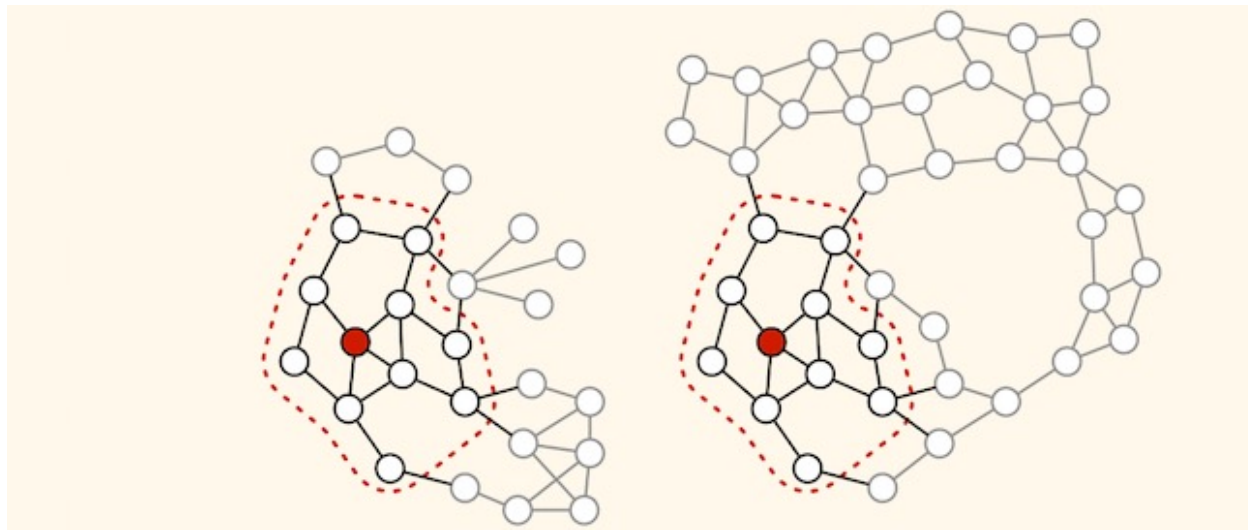
Kolaps obrazovanja je kolaps nacije.

– Natpis na ulazu Univerziteta Južna Afrika

Cilj predmeta



Ovladavanje tehnikama izbora, analize, implementacije i primene paralelnih i distribuiranih algoritama i struktura podataka, sa posebnim fokusom na blokčejn tehnologije



Sadržaj predmeta



- **Uvod u paralelne i distribuirane sisteme**
- **Algoritmi za deljenu memoriju** (engl. *shared memory*)
- **Algoritmi sa prenosom poruka** (engl. *message passing*)
- **Arhitektura softvera u distribuiranim sistemima**
- **Procesi, komunikacija i imenovanje u distribuiranim sistemima**
- **Koordinacija, konzistentnost i replikacija u distribuiranim sistemima**
- **Otpornost na greške i bezbednost u distribuiranim sistemima**
 - **Konsenzus algoritmi** – Paxos, Raft, problem vizantijskih generala
- **Blokčejn** – osnovni pojmovi, koncepti i tehnike, kriptografija
- **Blokčejn tehnologije** – javni blokčejn sistemi: Bitcoin, Ethereum, ...
- **Blokčejn tehnologija Hyperledger Fabric** – privatni blokčejn sistemi, koncepti i arhitektura, mehanizam transakcija, ostali Hyperledger projekti

Organizacija ispita



- Nastava: 3+3 (predavanja i računarske vežbe)
- Polaganje: **predispitne (70%) i ispitne (30%) obaveze**

I. Predispitne obaveze – ukupno 70 bodova

- 1. Zadatak (Go) – 20 bodova**
- 2. Projekat – 40 bodova**
- 3. Seminarski rad – 10 bodova**

2. Ispit – ukupno 30 bodova

1. Teorijske osnove paralelnih i distribuiranih sistema sa fokusom na blokčejn
2. Uslov za izlazak na ispit je osvojenih 36 bodova sa predispitnih obaveza

Kreativnost i priprema ispita



Izvor: [Bill Watterson](#) – [Calvin & Hobbes](#)

Ocenjivanje

51 – 60 : 6

61 – 70 : 7

71 – 80 : 8

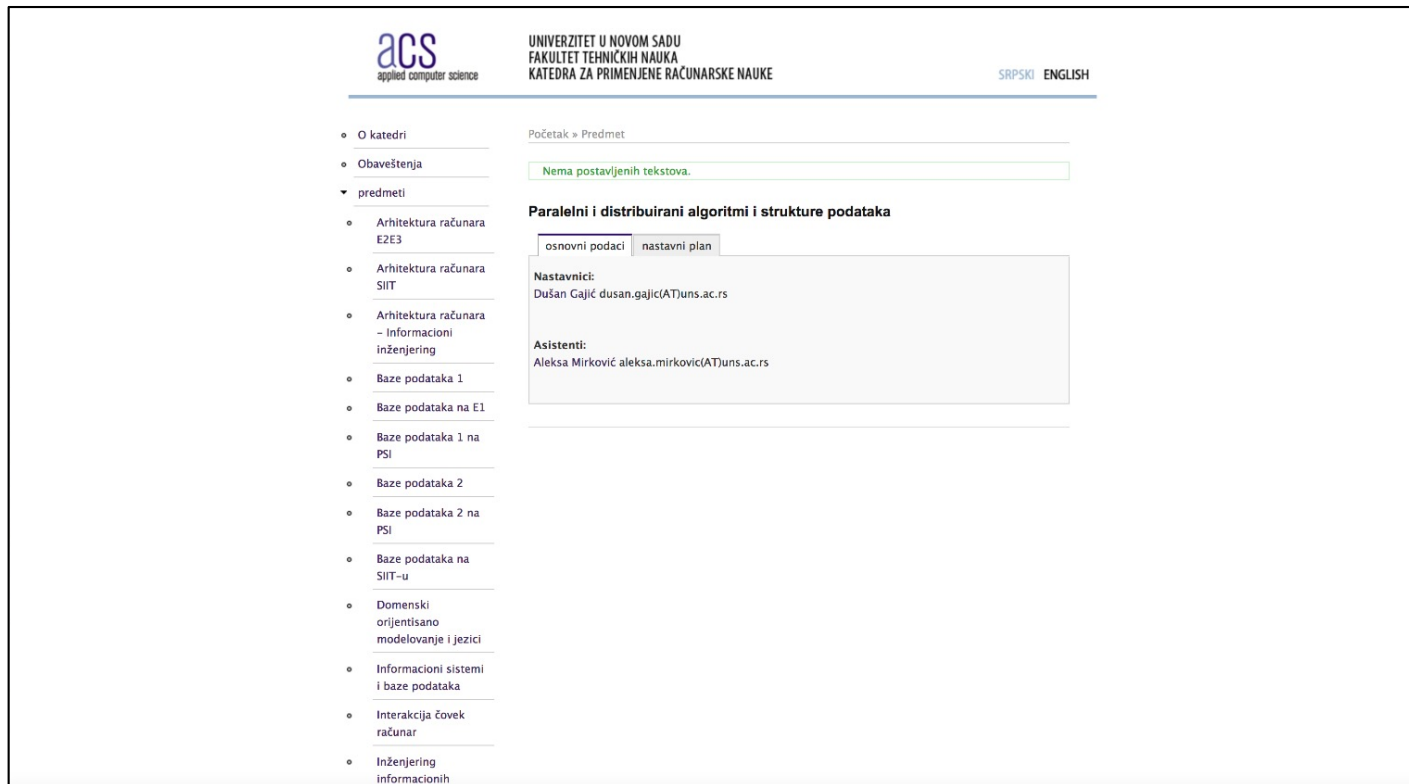
81 – 90 : 9

91 – 100 : 10

Sajt predmeta



- Sajt predmeta na: www.acs.uns.ac.rs
- [*Paralelni i distribuirani algoritmi i strukture podataka*](#)



The screenshot shows the website of the Applied Computer Science (ACS) department at the University of Novi Sad. The header includes the ACS logo, the university name in Serbian, and language options (SRPSKI, ENGLISH). A left sidebar contains a navigation menu with categories like 'O katedri', 'Obaveštenja', and 'predmeti'. The 'predmeti' section is expanded, showing a list of subjects including 'Arhitektura računara EZE3', 'Arhitektura računara SIIT', 'Arhitektura računara - Informacioni inženjering', and 'Baze podataka 1'. The main content area displays the selected subject 'Paralelni i distribuirani algoritmi i strukture podataka' with tabs for 'osnovni podaci' and 'nastavni plan'. Under 'osnovni podaci', it lists the lecturer 'Dušan Gajić' and the assistant 'Aleksa Mirković'.

Tehnologije



Go (Golang)



- **Kompajliran programski jezik sa statičkim tipovima**, efikasnost C++, čitljivost i upotrebljivost Python-a ili JS-a, i visoke performanse kod multiprocessinga i mrežnog rada
- Prvu verziju Google je predstavio 2009.
- Poznat i kao “**C za XXI vek**”, ali sa **bezbednim radom sa memorijom, sakupljanjem smeća i konkurentnom obradom** (CSP, gorutine i kanali)
- Podržava **imperativnu (strukturiranu) i konkurentu paradigmu**, implementira **CSP** (Communicating Sequential Processes) – formalni jezik Tony Hoare-a za opis interakcije u konkurentnim sistemima
- **Bogata biblioteka paketa**, dve implementacije – Google i gccgo
- **Go je korišćen za razvoj:**
 - Docker
 - Kubernetes
 - IPFS
 - Hyperledger Fabric
 - Netflix
 - Dropbox

```
package main

import "fmt"

func main() {
    fmt.Println("Hello, World")
}
```

Bitcoin

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshi@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

1. Introduction

Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for non-reversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party.

What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. Transactions that are computationally impractical to reverse would protect sellers from fraud, and routine escrow mechanisms could easily be implemented to protect buyers. In this paper, we propose a solution to the double-spending problem using a peer-to-peer distributed timestamp server to generate computational proof of the chronological order of transactions. The system is secure as long as honest nodes collectively control more CPU power than any cooperating group of attacker nodes.

1

<https://bitcoin.org/bitcoin.pdf>



- Zamišljen kao **"The World Computer"**
- Koristi virtuelnu mašinu kao okruženje za izvršavanje (engl. runtime environment) programa napisanih u **Tjuring-kompletnom programskom jeziku – Ethereum Virtual Machine (EVM)**, bajtkod u jeziku niskog nivoa zasnovan na radu sa stekom
- **Tjuring-kompletni programski jezik za opis transakcija**
 - Solidity, LLVM, Serpent
 - u suštini namenski jezici (engl. domain specific languages – DSL)
- Koncept **pametnih ugovora** (engl. smart contracts)
 - **programski kod koji dovodi do promene stanja sistema** kada se **određeni uslovi ispune**
- Koncepti **inicijalne ponude novčića – initial coin offering (ICO), NFT (non-fungible tokens)**
 - pametni ugovori koji se izvršavaju na Ethereum blokčejnu
- **Decentralizovane aplikacije (DApps) i DAO** (decentralizovane autonomne organizacije)
 - Primeri: CryptoKitties, EtherTweet, Etheria, domaći LemonMail

Hyperledger

- Open source kolaborativna inicijativa koju je vodila Linux fondacija, pokrenuta decembra 2015, prerasla u posebnu Hyperledger fondaciju



Frameworks



Permissionable smart contract machine (EVM)



Permissioned with channel support



Decentralized identity



Mobile application focus



Permissioned & permissionless support; EVM transaction family

Tools



Blockchain framework benchmark platform



As-a-service deployment



Model and build blockchain networks



View and explore data on the blockchain



Ledger interoperability



- **Hyperledger Fabric** je platforma za rešenja sa distribuiranom glavnom knjigom (engl. ledger)
 - **Mreža sa kontrolom pristupa bez native kriptovalute**
 - **Elastična i proširiva arhitektura** koja može da podrži kompleksnosti i specifičnosti koje postoje u ekonomskom ekosistemu
- IBM i Digital Asset donirali značajan deo koda

<https://github.com/hyperledger/fabric>

- Osobine:
 - **Modularnost**
 - **Poverljivost**
 - **Otpornost** (resiliency)
 - **Fleksibilnost**
 - **Skalabilnost i visoke performanse**





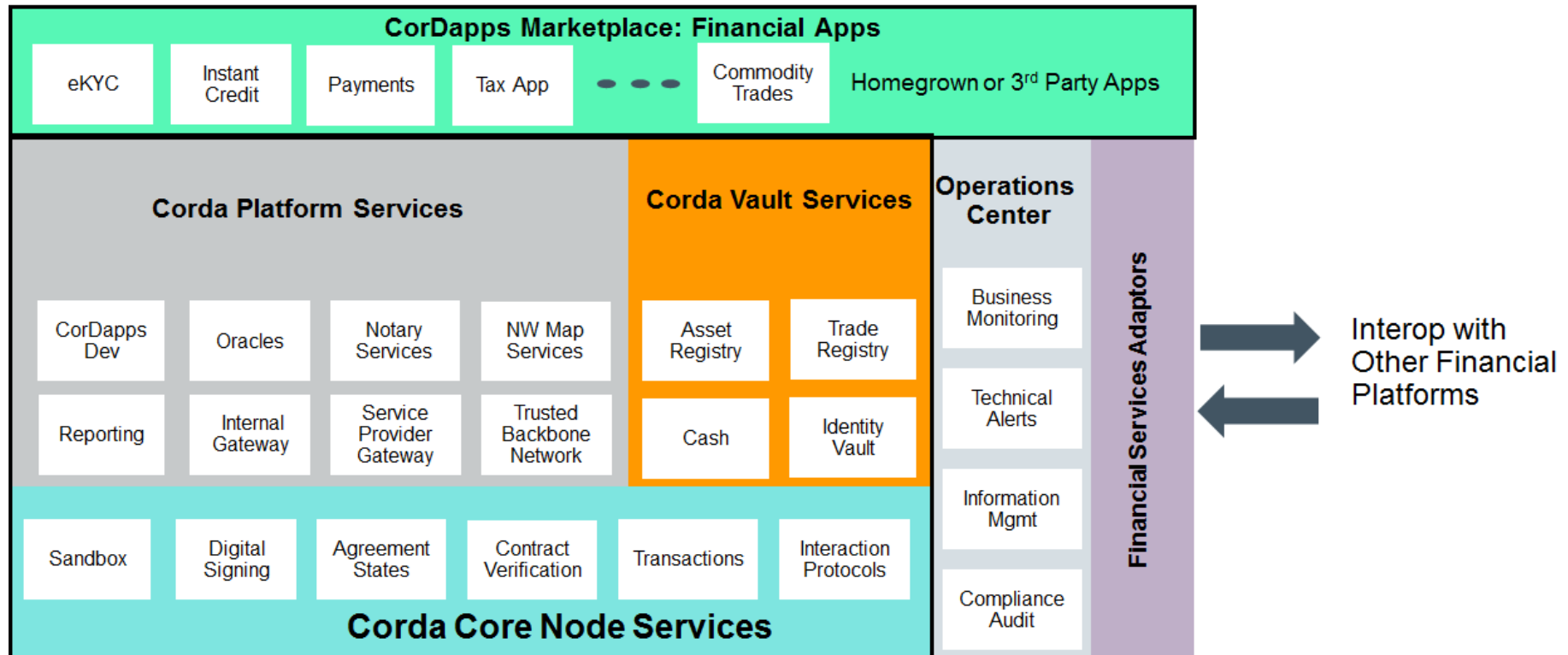
Izvor: <https://www.youtube.com/watch?v=js3Zjxbo8TM>



- **R3** nastao kao konzorcijum devet banaka (Bank of America, HSBC, UBS, Credit Suisse, ING, ...)
- Primene u **poslovnom domenu** (bankarstvo, osiguranje, tržišta kapitala, međunardona trgovina),
- **Privatna mreže sa kontrolom pristupa**
- Koristi **JVM**, pametni ugovori u **Javi** ili **Kotlinu**
- **DLT** koji **nije blokčejn**, transakcije se ne organizuju u blokove, već se obrađuju na pojedinačnom nivou u realnom vremenu
- Osobine:
 - **privatnost**
 - **performanse**
 - **skalabilnost**
 - **open source**



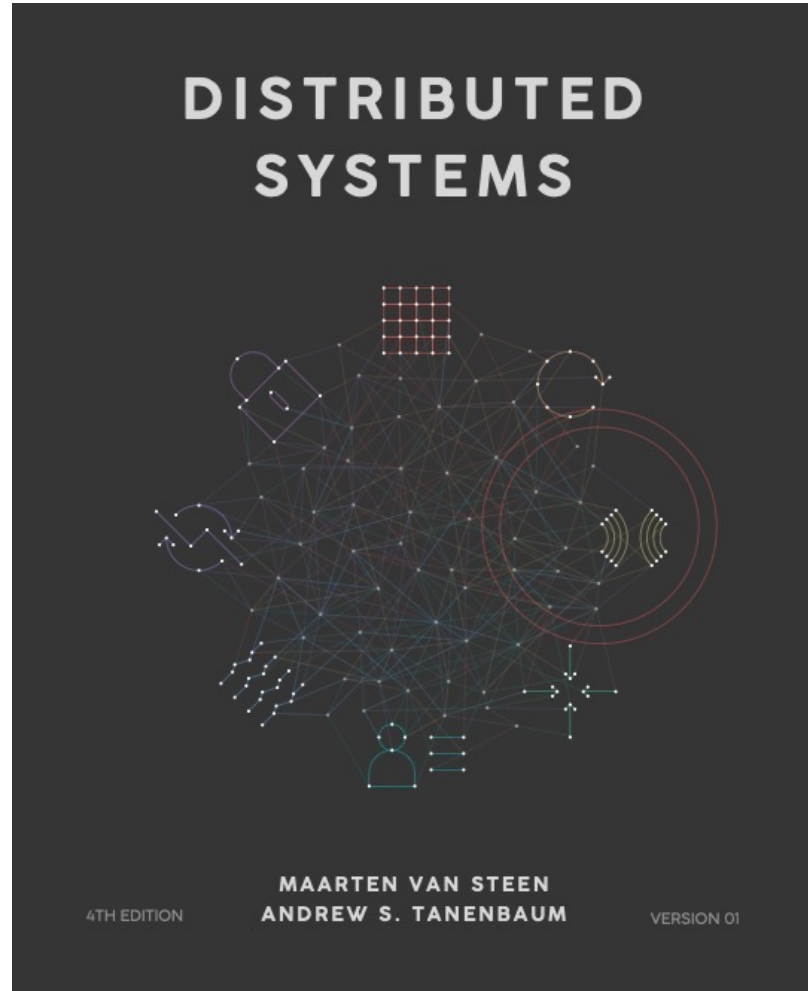
Corda Application Architecture



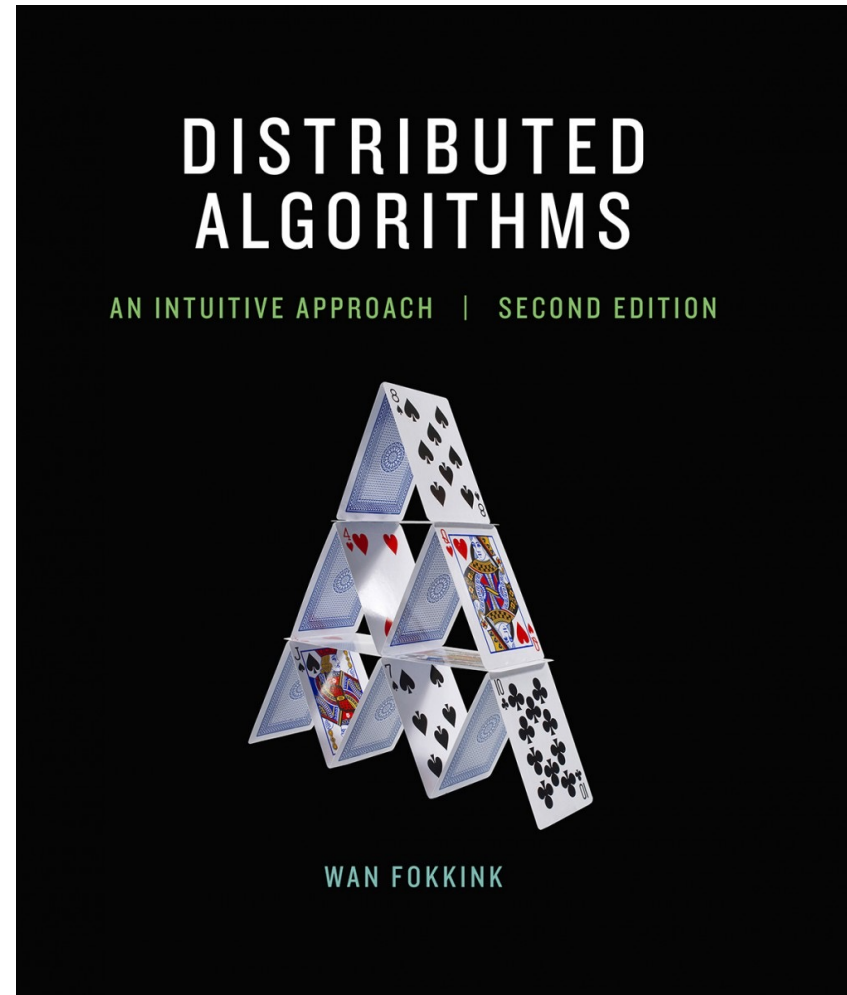
Izvor: <http://arunkottolli.blogspot.com/2017/10/r3-corda-application-architecture.html>

Reference

Literatura – Distribuirani sistemi i algoritmi



<https://www.distributed-systems.net>



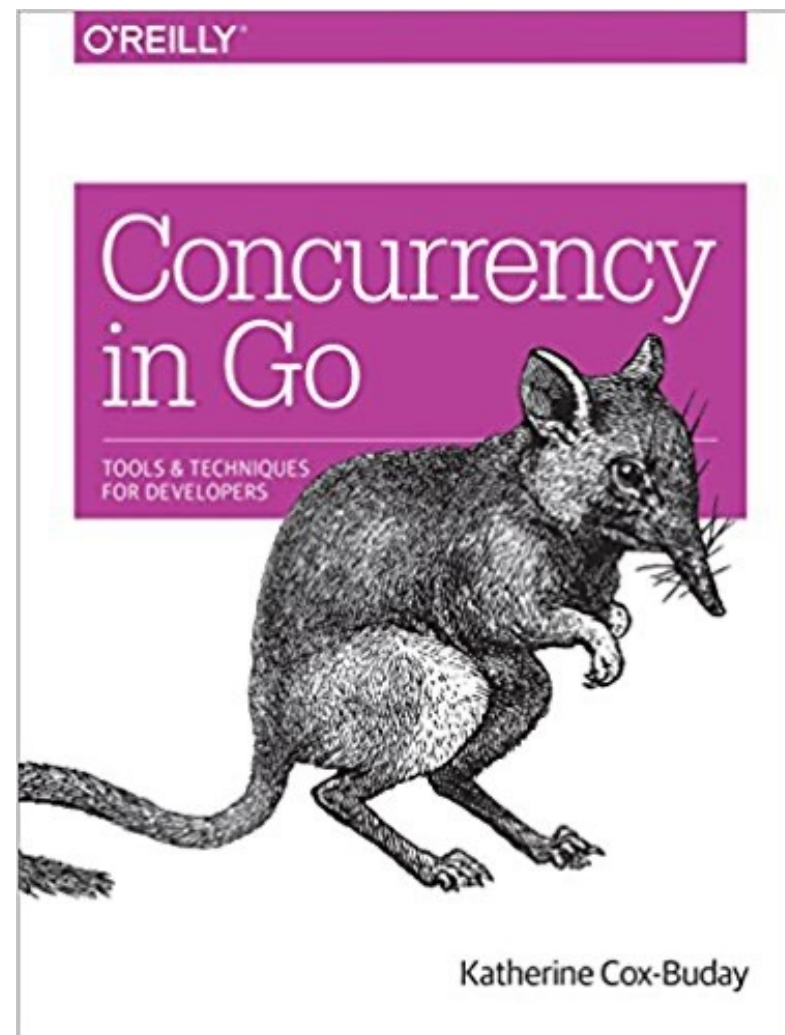
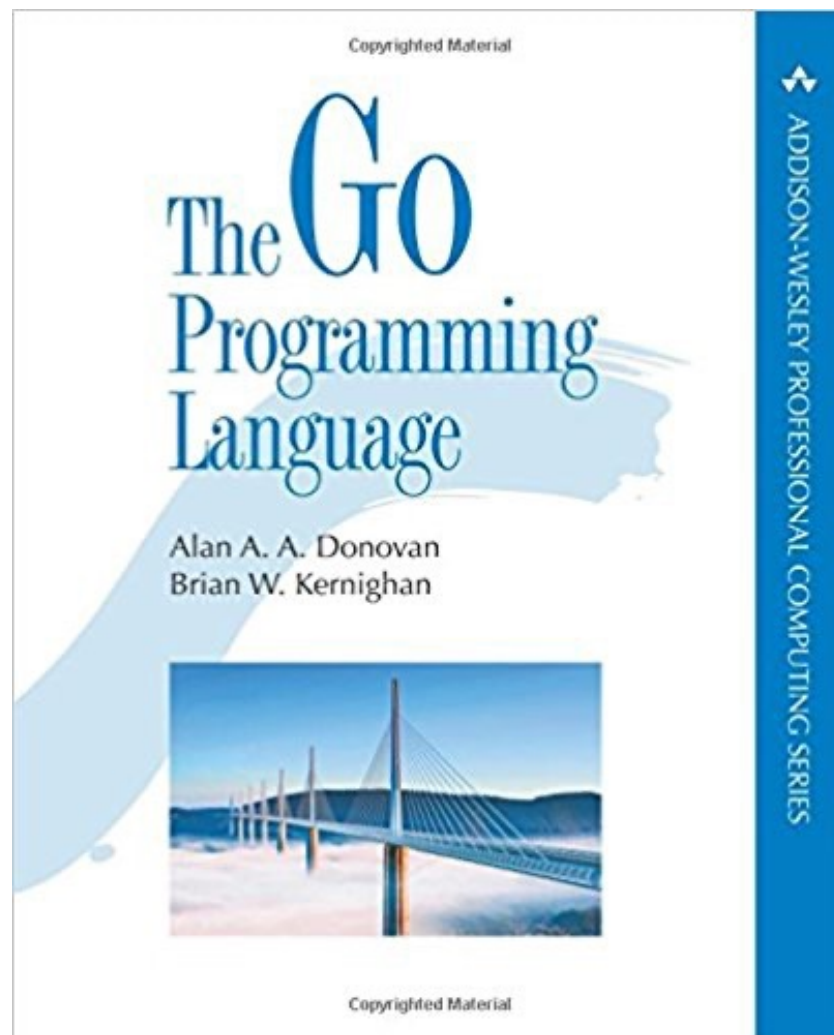
<https://www.cs.vu.nl/~tcs/da/>

Literatura – Konkurentni i distribuirani sistemi



<https://cet.rs/shop/java/konkurentni-i-distribuirani-sistemi/>

Literatura – Go



Literatura – Blokčejn

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshi@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

1. Introduction

Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for non-reversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party.

What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. Transactions that are computationally impractical to reverse would protect sellers from fraud, and routine escrow mechanisms could easily be implemented to protect buyers. In this paper, we propose a solution to the double-spending problem using a peer-to-peer distributed timestamp server to generate computational proof of the chronological order of transactions. The system is secure as long as honest nodes collectively control more CPU power than any cooperating group of attacker nodes.

1

<https://bitcoin.org/bitcoin.pdf>

Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains

Elli Androulaki
Artem Barger
Vita Bortnikov
IBM

Christian Cachin
Konstantinos Christidis
Angelo De Caro
David Enyeart
IBM

Christopher Ferris
Gennady Laventman
Yacov Manevich
IBM

Srinivasan Muralidharan*
State Street Corp.

Chet Murthy*

Binh Nguyen*
State Street Corp.

Manish Sethi
Gari Singh
Keith Smith
Alessandro Sorniotti
IBM

Chrysoula Stathakopoulou
Marko Vukolić
Sharon Weed Cocco
Jason Yellick
IBM

ABSTRACT

Fabric is a modular and extensible open-source system for deploying and operating permissioned blockchains and one of the Hyperledger projects hosted by the Linux Foundation (www.hyperledger.org).

Fabric is the first truly extensible blockchain system for running distributed applications. It supports modular consensus protocols, which allows the system to be tailored to particular use cases and trust models. Fabric is also the first blockchain system that runs distributed applications written in standard, general-purpose programming languages, without systemic dependency on a native cryptocurrency. This stands in sharp contrast to existing blockchain platforms that require "smart-contracts" to be written in domain-specific languages or rely on a cryptocurrency. Fabric realizes the permissioned model using a portable notion of membership, which may be integrated with industry-standard identity management. To support such flexibility, Fabric introduces an entirely novel blockchain design and revamps the way blockchains cope with non-determinism, resource exhaustion, and performance attacks.

This paper describes Fabric, its architecture, the rationale behind various design decisions, its most prominent implementation aspects, as well as its distributed application programming model. We further evaluate Fabric by implementing and benchmarking a Bitcoin-inspired digital currency. We show that Fabric achieves end-to-end throughput of more than 3500 transactions per second in certain popular deployment configurations, with sub-second latency, scaling well to over 100 peers.

*Work done at IBM.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
EuroSys '18, April 23–26, 2018, Porto, Portugal.
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-5584-1/18/04.
<https://doi.org/10.1145/3190508.3190538>

ACM Reference Format:

Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolić, Sharon Weed Cocco, and Jason Yellick. 2018. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *EuroSys '18: Thirteenth EuroSys Conference 2018, April 23–26, 2018, Porto, Portugal*. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3190508.3190538>

1 INTRODUCTION

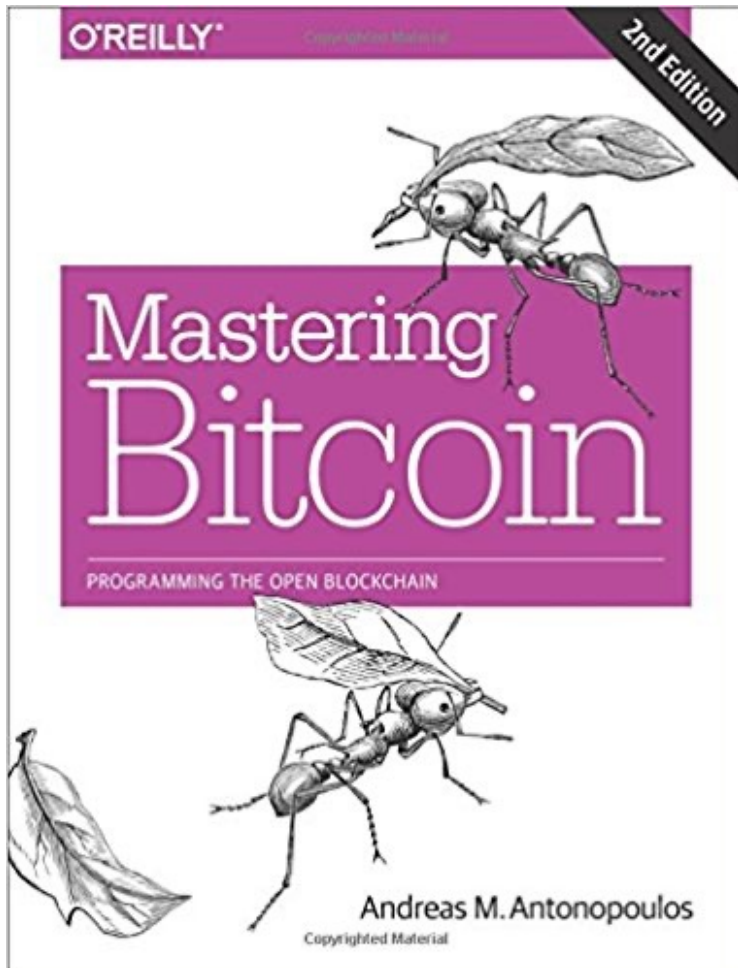
A blockchain can be defined as an immutable ledger for recording transactions, maintained within a distributed network of mutually untrusting peers. Every peer maintains a copy of the ledger. The peers execute a consensus protocol to validate transactions, group them into blocks, and build a hash chain over the blocks. This process forms the ledger by ordering the transactions, as is necessary for consistency. Blockchains have emerged with Bitcoin [3] and are widely regarded as a promising technology to run trusted exchanges in the digital world.

In a public or permissionless blockchain anyone can participate without a specific identity. Public blockchains typically involve a native cryptocurrency and often use consensus based on "proof of work" (PoW) and economic incentives. Permissioned blockchains, on the other hand, run a blockchain among a set of known, identified participants. A permissioned blockchain provides a way to secure the interactions among a group of entities that have a common goal but which do not fully trust each other, such as businesses that exchange funds, goods, or information. By relying on the identities of the peers, a permissioned blockchain can use traditional Byzantine-fault tolerant (BFT) consensus.

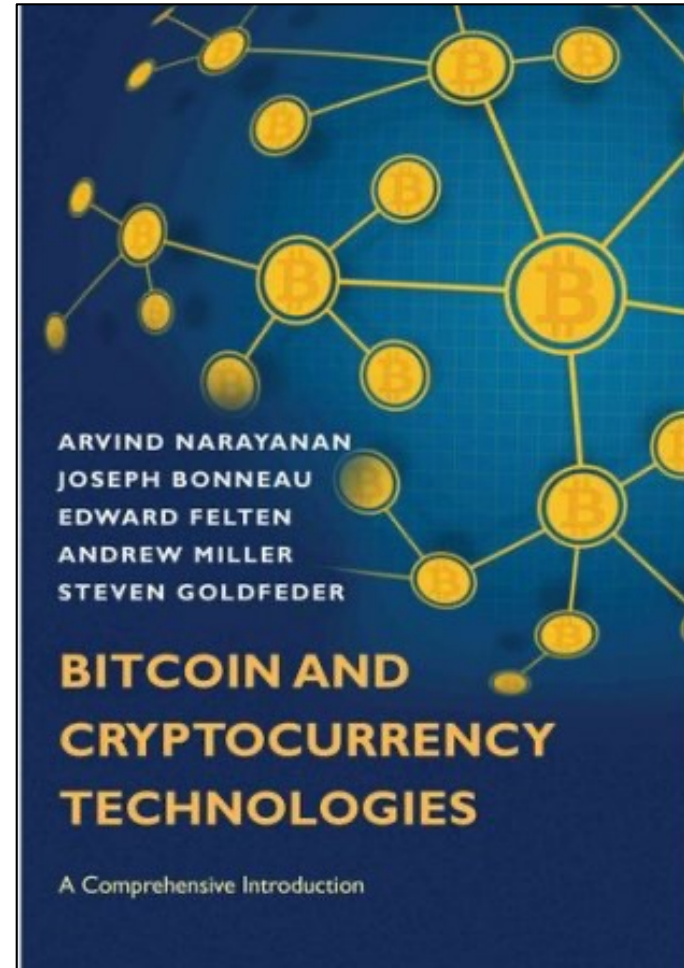
Blockchains may execute arbitrary, programmable transaction logic in the form of smart contracts, as exemplified by Ethereum [5]. The scripts in Bitcoin were a predecessor of the concept. A smart contract functions as a trusted distributed application and gains its security from the blockchain and the underlying consensus

<https://arxiv.org/abs/1801.10228>

Literatura – Blokčejn



<https://github.com/bitcoinbook/bitcoinbook>



<http://bitcoinbook.cs.princeton.edu/>

Online resursi – Hyperledger

- Hyperledger Whitepaper:
<http://www.the-blockchain.com/docs/Hyperledger%20Whitepaper.pdf>
- Hyperledger Architecture Working Group, Paper I:
https://www.hyperledger.org/wp-content/uploads/2017/08/HyperLedger_Arch_WG_Paper_I_Consensus.pdf
- Official documentation:
<https://hyperledger-fabric.readthedocs.io/en/latest/>
- Rocket Chat:
<https://chat.hyperledger.org/>
- StackOverflow:
<https://stackoverflow.com/questions/tagged/hyperledger-fabric>



Online resursi – Hyperledger

- Hyperledger Github:

The screenshot shows the GitHub organization page for the Hyperledger Project. The header includes navigation links: Features, Business, Explore, Marketplace, Pricing, and a search bar. The main section displays the Hyperledger Project logo, name, and website URL (https://www.hyperledger.org). Below this, there are tabs for Repositories (74) and People (110). The 'Pinned repositories' section lists six repositories:

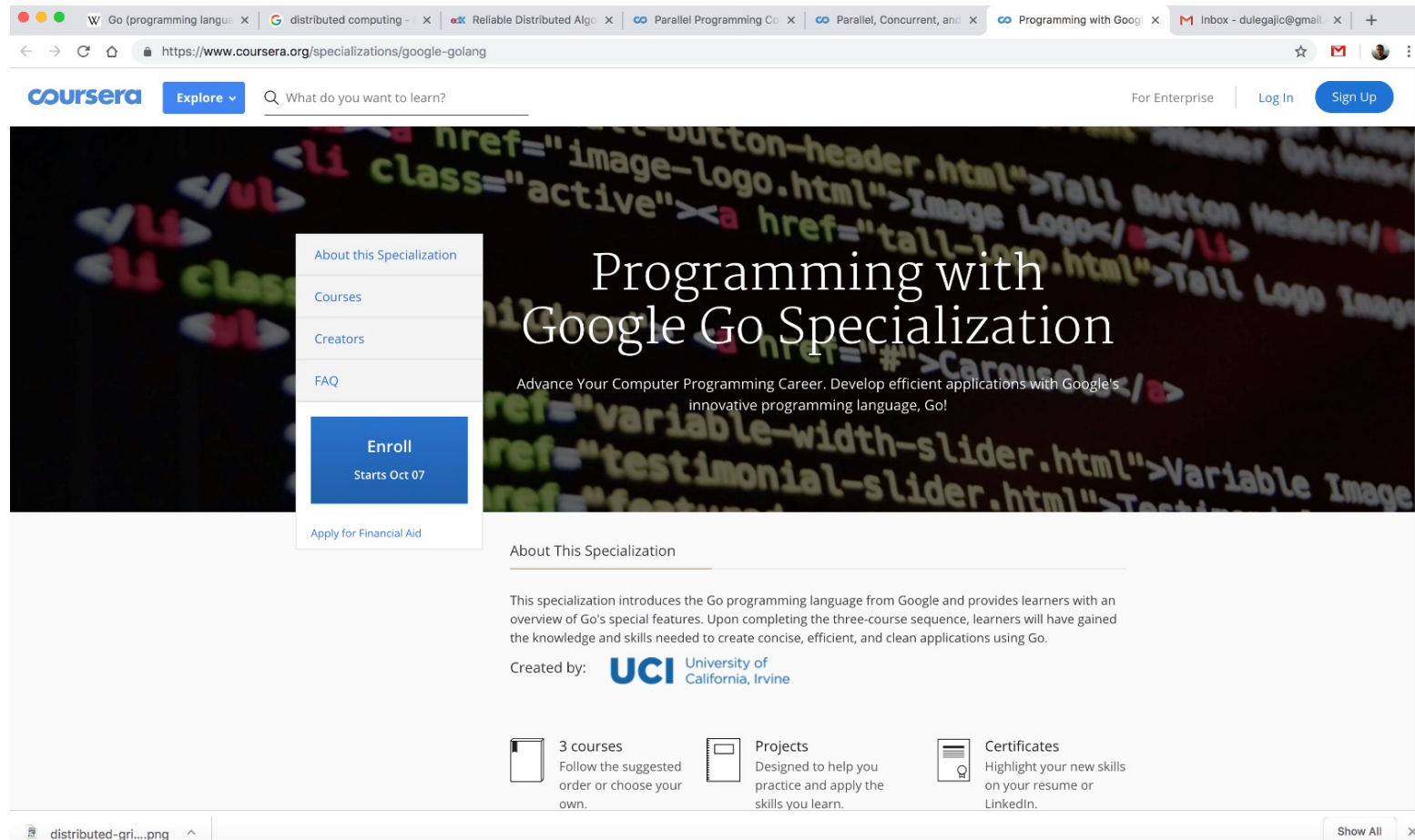
Repository Name	Description	Language	Stars	Forks
fabric	Read-only mirror of https://gerrit.hyperledger.org/r/#/admin/projects/fabric	Go	4.5k	2.7k
composer	Composer is a framework for building Blockchain business networks	JavaScript	920	434
sawtooth-core	Core repository for Sawtooth Distributed Ledger	Python	759	369
iroha	Iroha - A simple, decentralized ledger	C++	698	218
burrow	Hyperledger Burrow	Go	334	126
indy-node	Indy Node	Python	127	153

MOOCs – distribuirani sistemi

The screenshot shows the Coursera website for the 'Cloud Computing Specialization'. The browser's address bar displays the URL <https://www.coursera.org/specializations/cloud-computing#about>. The page features a dark header with the title 'Cloud Computing Specialization' and a subtitle 'Clouds, Distributed Systems, Networking. Learn about and build distributed and networked systems for clouds and big data.' A sidebar on the left contains links for 'About this Specialization', 'Courses', 'Creators', 'FAQ', and a prominent 'Enroll' button with the text 'Starts Oct 07'. Below the sidebar, there is a section titled 'About This Specialization' which describes the course structure and its connection to the University of Illinois at Urbana-Champaign's Master of Computer Science in Data Science degree program. A 'Learn More' button is provided at the bottom of this section. The page also includes a 'Sign Up' button in the top right corner.

Link: <https://www.coursera.org/specializations/cloud-computing>

MOOCs – Go



The screenshot shows the Coursera website for the 'Programming with Google Go Specialization'. The browser's address bar displays the URL <https://www.coursera.org/specializations/google-golang>. The page header includes the Coursera logo, an 'Explore' button, a search bar, and links for 'For Enterprise', 'Log In', and 'Sign Up'. The main content area features a large banner with the title 'Programming with Google Go Specialization' and a description: 'Advance Your Computer Programming Career. Develop efficient applications with Google's innovative programming language, Go!'. A sidebar on the left provides navigation links: 'About this Specialization', 'Courses', 'Creators', 'FAQ', and a large 'Enroll Starts Oct 07' button. Below the banner, the 'About This Specialization' section describes the course's focus on the Go programming language. At the bottom, three icons represent the course components: '3 courses' (Follow the suggested order or choose your own), 'Projects' (Designed to help you practice and apply the skills you learn), and 'Certificates' (Highlight your new skills on your resume or LinkedIn).

Link: <https://www.coursera.org/specializations/google-golang>