Charlie Crisp

Building a Blockchain Library for OCaml

Computer Science Tripos – Part II

Pembroke College

January 18, 2018

Proforma

Name:	**Charlie Crisp**
College:	**Pembroke College**
Project Title:	**Building a Blockchain Library for OCaml**
Examination:	**Computer Science Tripos – Part II, July 2018**
Word Count:	**????**[1]
Project Originator:	KC Sivaramakrishnan
Supervisor:	KC Sivaramakrishnan

Original Aims of the Project

To build a library in OCaml, which can be used as a building block for Blockchain applications. The library should allow participating nodes to own a shared copy of a Blockchain data structure, agreed upon using consensus. Nodes should also be able to commit transactions to the blockchain, which should then be visible to other participating nodes.

Work Completed

All that has been completed appears in this dissertation.

Special Difficulties

None

[1]This word count was computed by `detex diss.tex | tr -cd '0-9A-Za-z \n' | wc -w`

Declaration

I, Charlie Crisp of Pembroke College, being a candidate for Part II of the Computer Science Tripos, hereby declare that this dissertation and the work described in it are my own work, unaided except as may be specified below, and that the dissertation does not contain material that has already been used to any substantial extent for a comparable purpose.

Signed [signature]

Date [date]

Contents

List of Figures

Acknowledgements

I would like to thank KC Sivaramakrishnan for being an extremely helpful supervisor throughout the duration of the dissertation, as well as over the past three years.
I would also like to thank Anil Madhavapeddy for allowing me to use his laptop for the duration of the dissertation, and being a very supportive DoS.
Finally I'd like to thank my friends and family for supporting me through my final year.

Chapter 1

Introduction

The lecture suggests putting a paragraph here to explain the project, and how well it's been completed. Question - how is this different to what is on page 3?

1.1 What is the Blockchain?

The blockchain, in its simplest form, is a series of blocks of data, where each block contains the cryptographic hash of the previous block in the chain. This means that the chain can exhibit arbitrary branching. Figure 1.1 is a graphical representation of a typical blockchain data structure.
The significance of blocks containing hashes of previous blocks, is that if a change were to be made to a previous block, it would change the hashes of all subsequent blocks in the chain. This means that once a block is committed to the blockchain, it cannot be modified.

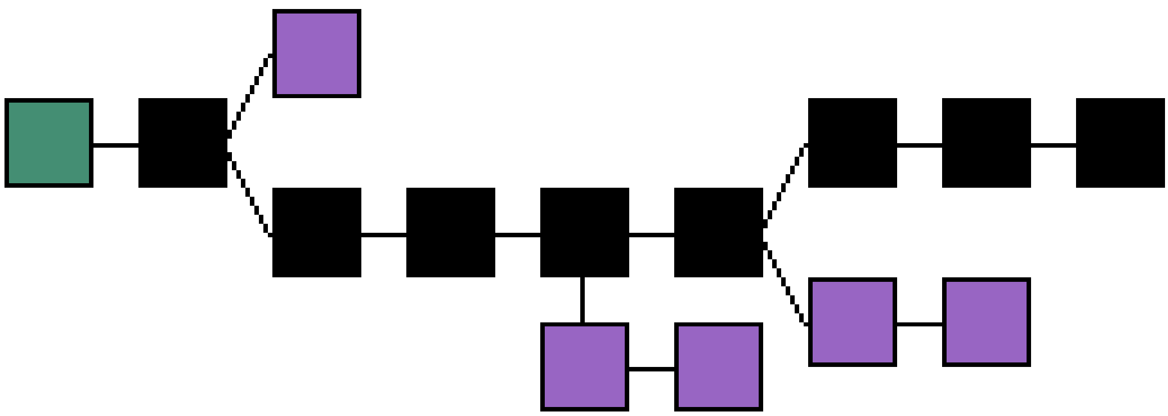

Figure 1.1: A typical blockchain structure

1.2 The History of the Blockchain

The blockchain, as a cryptographically secure chain of blocks, was first conceptualised by Stuart Haber and W. Scott Stornetta in 1990 [1]. However, until the creation of Git [4] in 2005, the blockchain was still a relatively niche concept.
Git uses the blockchain as a basis for storing a secure history of code in the form of commits made to branches. We will explore the use of the Git as a blockchain during chapter ???

The invention of Bitcoin[2] in 2008 is seen by many as the most pivotal moment in the history of blockchain technologies. Bitcoin combines the blockchain with a 'Proof of Work' consensus algorithm and the result of this is a decentralised, trust-less, peer to peer network which doesn't suffer from the double spending problem.

1.3 Blockchain Today

At the time of writing, cryptocurrencies are generating a huge amount of excitement and cynicism in technology and economics communities. The value of Bitcoin is fluctuating on a day-by-day basis and many alternative currencies are gaining in popularity.
For example, Ethereum is a blockchain platform that allows participants to run arbitrary[1] code on the blockchain. These are known as Smart Contracts.

[1]Whilst there are restrictions to code run on the blockchain (e.g. it has to be deterministic), the Ethereum language 'Solidity' is Turing complete.

Chapter 2

Preparation

2.1 Starting Point

The project will build upon functionality provided by Irmin [1] which is a distributed database system. Irmin is fast, durable and has the branching capabilities which are required to build a blockchain.
The project will also make use of Ezirmin [3] which provides a simplified API to Irmin.

2.2 Using OCaml

I had not used OCaml before so I had to learn it. Lwt and all

2.3 Requirements Analysis

In order to build a blockchain there were a few different technical parts of the project which I needed to research. Understanding these concepts was crucial because it allowed me to make informed decisions on how to build my project. In this section, I have outlined some of the key technical concepts that I tackled when preparing to build my project.

2.3.1 Git as a blockchain

Git provides a data structure which can be interacted with via a command line API. However, it is not trivial why these data structure is classed as a blockchain.

In order to convince oneself of this, it is worth considering what features are required for a data structure to be considered a blockchain. Whilst there is no universally agreed definition of a blockchain, it is commonly accepted that it will exhibit the following features:

1. Data is stored in blocks.

2. Blocks are ordered.

3. Each block contains a link to its parent block.

Now we can consider the promises that Git makes and ensure that the above conditions are satisfied.

1. Git stores data as sequences of 'commits' which can be thought of as blocks.
2. Git history is stored as a tree structure with arbitrary branching. This imposes an order on commits or blocks.
3. Each Git commit contains the hash of the previous or parent commit.

2.3.2 Consensus Algorithms

Typically, cryptocurrencies (which rely on blockchain technology) use a Proof of Work protocol to achieve consensus. There are, however, other ways of achieving consensus, and I evaluated a number of these in order to inform my approach to building consensus into my project.

Proof of work

The proof of work method for achieving consensus is actually very simple. Whenever a block is received by a participating node, the node will check that the block contains a proof of computational work done. In the field of cryptocurrencies, this proof takes the form of a random sequence of data which, when appended to the end of the block, causes its hash to be prefixed with a set number of 0s. Because the data appended to the end of a block can only be found by a brute force method, which will require lots of computation/work.

Proof of stake

Paxos

Raft

2.3.3 Mempools

When making a transaction using Bitcoin or some other cryptocurrencies, a request for this transaction is placed in what is known as a Mempool[1]. This Mempool can be thought of as a waiting room for transactions. When miners are assembling and mining blocks, this is where they will pick the transactions from.

[1] Mempool is short for Memory Pool

Chapter 3

Implementation

Chapter 4

Evaluation

Chapter 5

Conclusion

Bibliography

[1] Stuart Haber and W. Scott Stornetta. How to time-stamp a digital document. 1990.

[2] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[3] KC Sivaramakrishnan. Ezirmin: An easy interface to the irmin library. `http://kcsrk.info/ocaml/irmin/crdt/2017/02/15/an-easy-interface-to-irmin-library/`, 2017.

[4] Linus Torvalds. Git: A distributed version control system, 2005.

Appendix A

Project Proposal

Computer Science Tripos - Part II - Project Proposal

Building a Blockchain Library for OCaml

Charlie Crisp, Pembroke College

December 29, 2017

Project Supervisor: KC Sivaramakrishnan
Director of Studies: Anil Madhavapeddy
Project Overseers: Timothy Jones & Marcelo Fiore

Introduction

The blockchain, in its simplest form, is a tree-like data structure. Chunks of data are stored in 'blocks' which contain the hash of the contents of the previous block. This creates a 'blockchain' which can exhibit branching in the same way that a tree data structure can (see Figure 1). One of the most important features of a blockchain, is that a change in a block, will alter the block's hash, thereby altering all the future blocks in the chain. This makes it very easy to validate that the data in a blockchain is trustworth, by verifying the hash in a block, is the same as the hash of it's parent's content.

Blockchain technology has generated a lot of interest in recent times, but

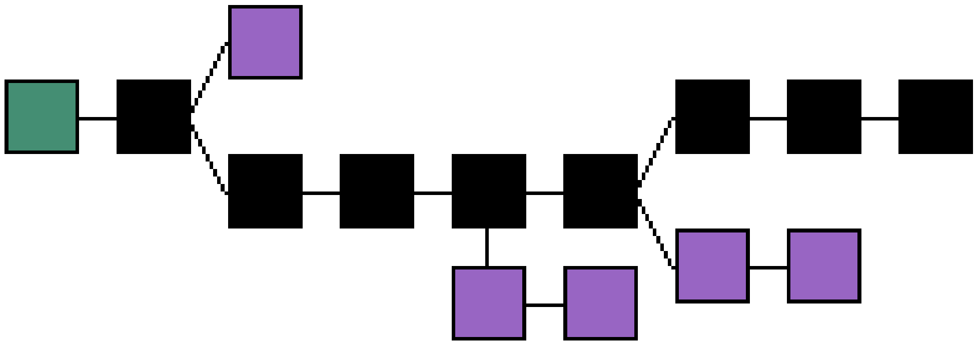

Figure 1: A typical blockchain structure [3]

mostly in the field of cryptocurrencies. With a simple Proof of Work consensus algorithm, the blockchain can be used to build a secure, distributed ledger of transactions. However, whilst the uses of the blockchain are far wider reaching than cryptocurrencies, progress outside of this field has been much slower.

I will build a pure OCaml, reusable blockchain library to allow the creation of distributed, secure ledgers, which are agreed upon by consensus. The library will allow users to create and add entries to a distributed blockchain ledger with just a few lines of code. The users will also be able to trust that entries in the blockchain are exactly replicated across all nodes in the network.

It will be built on top of Irmin [1] - a distributed database with git-like version control features. Being pure OCaml, the blockchain nodes can be compiled to unikernels or JavaScript to run in the browser. I will evaluate the blockchain by prototyping a decentralised lending library and evaluating the platform's speed and resilience.

Starting point

The project will build upon functionality provided by Irmin [1] which is a distributed database system. Irmin is fast, durable and has the branching capabilities which are required to build a blockchain.

Resources required

I will be using a Macbook provided by OCaml Labs [2] in order to develop the source code for the project. If the Macbook fails, then I will easily be able to transfer my work onto the MCS machines, as my project has no special requirements.
My work will also be backed up to a git repository hosted on GitHub and saved to a dedicated memory stick on a daily basis.
During the evaluation stage I will be running my platform on different cloud based devices and/or Raspberry Pi's. There are many possible providers for cloud computing, including Amazon Web Services and Microsoft Azure. OCaml Labs [2] will provide the necessary funds to acquire these resources.

Background

Consensus

Consensus is a group process where a network of nodes will reach a general agreement. There are different ways of achieving consensus but here are some of the most common:

1. **Proof of Work**: Trust is given to nodes which can prove that they have put in computational work. This is the consensus mechanism used by Bitcoin.

2. **Proof of Stake**: Nodes are selected to validate blocks based on their stake in the blockchain. There are few variations on this algorithm which introduce notions such as delegation or anonymity.

3. **Raft Consensus**: A leader is elected and acts as a governing authority until it fails or disconnects, whereupon a new leader is elected.

Work to be completed

The work for this project will be split into the following major parts.

1. Design and build a module to allow nodes to create and maintain a blockchain ledger. This will include allowing nodes to add blocks to the chain and to form new branches.

2. Design and build a module to allow nodes to interact over a network and to achieve consensus. As highlighted the Background section, there are many different ways to achieve consensus, and a large part of this work will be to determine which method is most suitable. This decision will take into account a method's failure tolerance in terms of nodes failing and network failure, as well as general speed and any requirements (e.g. computational work for a Proof of Work algorithm).

3. Design an application using these modules. This will take the form of a book lending platform where nodes will be able to register books and lend them to other nodes in the network. This application has been chosen, because the blockchain library should allow for typically centralised applications to be created in a decentralised way. It will also allow for testing of critical features, for example, books should never be 'doubly-spent', i.e. if one user believes they have ownership of a book, then no other user will think the same.

4. Design an evaluation program to simulate different load on the lending platform. This will be run in different configurations in order to measure the performance of the platform.

Evaluation metrics and success criteria

I will consider the project to be a success if the following criteria are achieved:

1. Nodes in the network are able to connect and communicate information.

2. Nodes are able to achieve consensus about the state of the distributed ledger.

3. Nodes are able to reconnect after being individually disconnected.

4. Nodes are able to re-converge after a network partition.

In order to evaluate the performance of the system, I will measure the *throughput* and *speed* of transactions of the book lending platform. Throughput will be measured in transactions per second, and speed will be quantified as time taken to complete a transaction. I will evaluate how these properties vary with respect to the following metrics:

1. **Number of nodes**: I will scale the number of nodes in the network between the range of 2 and 5.

2. **Rate of transactions**: I will vary the number of transactions made per second.

Should I achieve and be able to measure the above criteria within the time frame of my project, I will further test system against the following metrics:

1. **Network latency between nodes**

2. **Network bandwidth of nodes**

Timetable

1. **Michaelmas Weeks 2-4** (12/10/17 - 01/11/17):
   Set up an environment for developing OCaml and familiarise myself with the language and it's module system. This is important because the blockchain library needs to be reusable, and therefore well isolated.

2. **Michaelmas Weeks 5-6** (02/11/17 - 15/11/17):
   Familiarise myself with Irmin and it's data structures. This is important as I have never used the library before, but it will be used to build the blocks in the blockchain library. In this time I will also begin to design the API of my library.

3. **Michaelmas Weeks 7-8** (16/11/17 - 29/11/17):
   Finalise the API and start to build the module for creating and interacting with a distributed ledger. This will also involve investigating which hashing algorithms can be used to form the blockchain data structure.

4. **Christmas Vacation** (30/11/17 - 17/01/18):
   Finalise the API of the module for achieving consensus between multiple nodes. This work will also include investigating different methods of consensus and their suitability for my project.

5. **Lent Weeks 1-2** (18/01/17 - 31/01/18):
   Build the module for achieving consensus between modules. I will also start work on an lending library application which will be used to evaluate the performance of the blockchain library.

6. **Lent Weeks 3-4** (01/02/18 - 14/02/18):
Finish work on the lending library application and install it on a number of Raspberry Pi and/or cloud based devices. I will also begin work on my dissertation and I aim to complete the Introduction and Preparation chapters.

7. **Lent Weeks 5-6** (05/02/18 - 28/02/18):
Evaluate the performance of the platform by simulating load from each of the devices and measuring the speed of transactions. A stretch goal for this period is also to evaluate a range of further metrics. Additionally I will continue work on my dissertation and aim to complete the Implementation chapter.

8. **Lent Weeks 7-8** (01/03/18 - 14/03/18):
Finish a first draft of my the dissertation by writing the Evaluation and Conclusion chapters. I will also send the dissertation to reviewers to get feedback.

9. **Easter Vacation** (15/03/18 - 25/04/18):
With a first draft of the dissertation completed, I will use this time to review the draft and to make improvements. I will also incorporate feedback from reviewers, and complete the Bibliography and Appendices chapters.

10. **Easter Weeks 1-2** (26/04/18 - 09/05/18):
Conclude work on dissertation by incorporating final feedback from reviewers.

11. **Easter Week 3-Submission Deadline** (10/05/18 - 08/05/18):
I aim to have completed the dissertation by this point, and to be focusing on my studies. However, this time may be needed to make any final changes.

References

[1] Irmin - A pure OCaml, distributed database that follows the same design principles as Git.
https://github.com/mirage/irmin

[2] OCaml Labs - An initiative based in the Computer Laboratory to promote research, growth and collaboration within the wider OCaml community **http://ocamllabs.io/**

[3] Image of blockchain data structure from Wiki Commons. **https://commons.wikimedia.org/wiki/File:Blockchain.png**