Faculty Electrical Engineering, Mathematics and Computer Science

TODO title

TODO subtitle

Kelong Cong

Supervisors:
Dr. J. Pouwelse
Dr. Unknown
Prof. Unknown

Delft, Month 2017

Abstract

THIS IS MY ABSTRACT

Preface

Please write all your preface text here. If you do so, don't forget to thank your supervisor, other committee members, your family, colleagues etc. etc.

Contents

Chapter 1

Introduction

Nothing here. . .

Chapter 2

Checkpoint Consensus

2.1 Preliminaries

2.1.1 Requirements

- Permissionless
- Byzantine fault tolerant
- No PoW
- Works under churn
- Underlying data structure is TrustChain
- Detects forks or double-spends
- No step in the protocol blocks transactions
- Application independent

2.1.2 Assumptions

- We elect N consensus promoters in every round, we assume the number of faulty promoters is f and $N = 3f + 1$.
- Promoters have the complete history of the previously agreed set of transactions.

2.1.3 Notations

- $y = H(x)$ is a cryptographically secure hash function (random oracle), the domain x is infinite and the range is $y \in \{0, 2^{256} - 1\}$.
- Every node in the system has an identifier i and a blockchain

$$B_i = \{b_{i,j} : j \in \{1 \dots h\}\},$$

  where h is the height of the chain. Note that $h = |B_i|$
- Each block $b_{i,j}$ has a type $t \in \{\tau, \gamma\}$, denoted by $b^t_{i,j}$. Blocks without the superscript can be of any type.
- $T(b_{i,j}) = \{\tau, \gamma\}$ is the type function, where its domain is a block and outputs the corresponding type of the block.

- A block of type τ is a *transaction block*. It is a six-tuple, i.e.

$$b^{\tau}_{i,j} = (H(b_{i,j-1}), h_s, h_r, s_s, s_r, m).$$

  h_s and h_r denote the height (the sequence number for when the transaction is made) of the sender and receiver respectively. s_s and s_r denote the signature of the sender and the receiver respectively.

- A block of type γ is a *checkpoint block*. It is a three-tuple, i.e.

$$b^{\gamma}_{i,j} = (H(b_{i,j}), H(\mathcal{C}_r), p)$$

  where $\mathcal{C}_r$ is the consensus result in round r and $p \in 0, 1$ which indicates whether i wish to become a promoter in the following consensus round.

- Given the input $b^{\tau}_{i,j}$, we define the *get-neighbouring-checkpoints function*

$$C(b^{\tau}_{i,j}) = (b^{\gamma}_{i,a}, b^{\gamma}_{i,b})$$

  where $a = \arg\min_{k,k<j,T(b_{i,k})=\gamma}(j-k)$ and $b = \arg\min_{k,k>j,T(b_{i,k})=\gamma}(k-j)$.

- Given two γ transactions, we define *get-piece function*

$$P(b^{\gamma}_{i,a}, b^{\gamma}_{i,b}) = \{b_{i,j} : b_{i,j} \in B_i, a \le j \le b\}.$$

- Given two blockchains and a message, we define the *do-transaction function*

$$X_\tau(B_s, B_r, m) = (B'_s, B'_r)$$

  where $B'_s = \{(H(b_{s,h_s}), h_s{+}1, h_r{+}1, s_s, s_r, m)\} \cup B_s$, $B'_r = \{(H(b_{r,h_r}), h_s{+}1, h_r{+}1, s_s, s_r, m)\} \cup B_r$, $h_s = |B_s|$ and $h_r = |B_r|$.

- Given a blockchain, we define the *do-checkpoint function*

$$X_\gamma(B_i, r, p) = \{(H(b_{i,h}), H(\mathcal{C}_r), p)\} \cup B_i$$

  where $h = |B_i|$, and $\mathcal{C}_r$ is the latest consensus result.

- Note that X_τ and X_γ perform a state transition.

- The result of a consensus in round r is a set of two-tuple of checkpoints. Namely, $\mathcal{C}_r = \{(b^{\gamma}_{i,a}, b^{\gamma}_{i,b}) : a < b, \text{agreed by the promoters}\}$.

2.2 Checkpoint consensus

2.2.1 Promoter registration

Node i can register as a promoter when the latest consensus result is announced (suppose after the completion of round $r-1$), then it generates a new block using $b = T_\gamma(B_i, r-1, 1)$. The current promoters (in round r) may decide to include b in the new consensus result. If b is in it, then i becomes one of the promoter of round $r+1$.

We can fix the number of promotors to N by sorting the promotors by their “luck value”' and taking the first N.

2.2.2 Setup phase

We begin in the state where $\mathcal{C}_{r-1}$ has just been agreed but has not been disseminated yet.

Lemma 1. *Nodes in the network can always verify the validity of the consensus result.*

Proof. □

Lemma 2. *The new set of promoter for the next consensus round is consistent with respect to all the nodes in the network.*

Lemma 3. *Promoters waiting for a some time Δ to collect transactions does not violate the asynchronous assumption.*

Corollary 1. *The setup phase satisfies the validity, correctness and termination properties.*

2.2.3 Consensus phase

2.3 Fraud detection

Here we provide two techniques for fraud detection. The first guarantees fraud detection but is not practical. The second is a randomised solution that detects fraud with a high probability.

2.3.1 Breadth first search

2.3.2 Random sampling

Bibliography