Qualificação

Guilherme

FGV

2019

Bitcoin

- É uma cripto-moeda descentralizada, sendo uma forma de dinheiro eletrônico.
- É considerada a primeira moeda digital mundial descentralizada, constituindo um sistema econômico alternativo (peer-to-peer electronic cash system), e responsável pelo ressurgimento do sistema bancário livre.
- O Bitcoin permite transações financeiras sem intermediários, mas verificadas por todos usuários (nodos) da rede, que são gravadas em um banco de dados distribuídos, chamado de blockchain, uma rede descentralizada.

Agda

- Linguagem funcional e com sistema de tipos expressivo, capaz de representar especificações.
- Capaz de especificar e programar em um único lugar. Processo de verificação acontece no compilador.

Agda - II

- Linguagem não possui Built-in como em Python. Tipos como inteiros, ponto flutuantes, strings, vetores deve, ser definidos pelo próprio usuário.
- Tipos em Agda são uma generalização de tipos de dados algébricos encontrados em Haskell e ML.

Agda — III

- Definindo os números naturais com axiomas de peano

```
data ℕ : Set where
  zero : ℕ
  suc  : ℕ → ℕ
```

Agda — IV

- Adição em Agda:

$$
\begin{aligned}
&\_+\_ : \mathbb{N} \to \mathbb{N} \to \mathbb{N} \\
&\mathsf{zero} \quad + \; n = n \\
&\mathsf{suc}\; m + \; n = \mathsf{suc}\; (m + \; n)
\end{aligned}
$$

Agda - V

- Dizemos que um certo tipo é dependente se este depende de um valor.
- Exemplo — Listas indexadas por seu tamanho:

```
data Vec (A : Set) : ℕ → Set where
  []  : Vec A 0
  _::_ : ∀ {n} → A → Vec A n → Vec A (suc n)
```

Agda - VI

```
data GenesisBlock : ℕ → Set where
  block : (n : ℕ) → (sb : SimpleBlock) → n ≡ hashBlock sb → GenesisBlock n

data Block : ℕ → ℕ → Set where
  block : (n : ℕ) → (m : ℕ) → (sb : SimpleBlock) → m ≡ hashBlock sb → Block n m

data Blockchain : ℕ → Set where
  gen : {n : ℕ} → GenesisBlock n → Blockchain n
  cons : {n m : ℕ} → Block n m → Blockchain n → Blockchain m
```

Objetivos

- Programar uma criptomoeda (parecida com o Bitcoin) em Agda. Uma linguagem com tipos dependentes.

Justificativa

- Programar um protocólo de criptomoedas livre de BUGs
- Com Agda, além da programação da criptomoeda, temos a sua especificação de como ela deveria se comportar

O que já foi feito

- Programada cripto-moeda em python
- Programada parte da blockchain em Agda
- Pelo paper, já foi feito parte de transações. UTXO (Unspent transaction output)

Próximos passos

- Juntar a parte da blockchain com transações
- Se necessário, provar alguns teoremas relacionados ao bitcoin

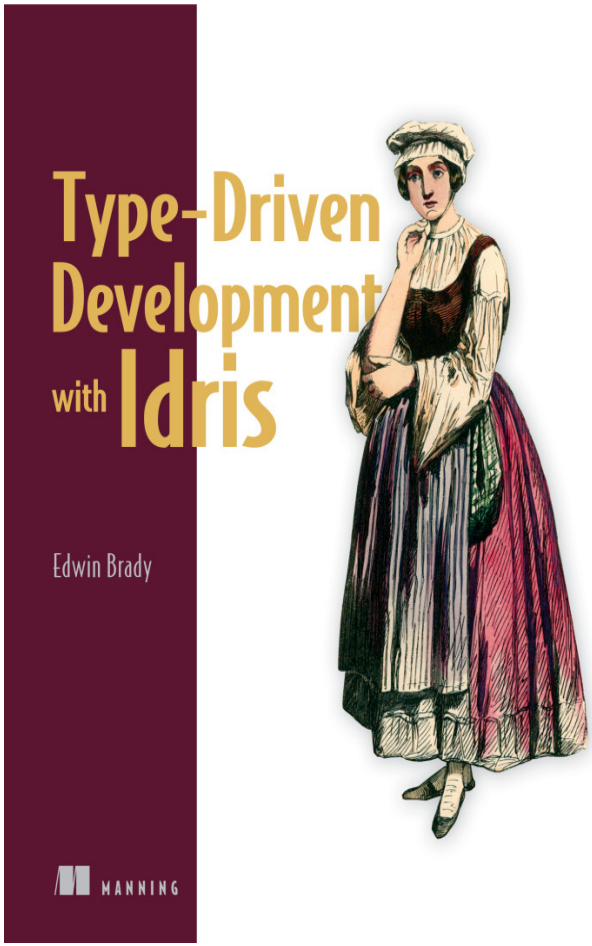
Teoremas

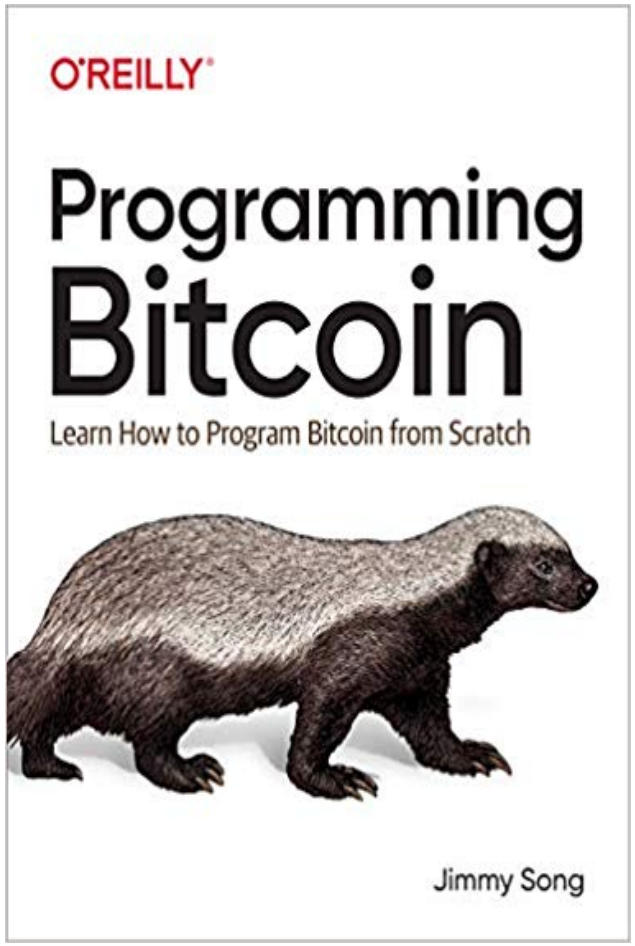
- Se uma transação tem algum output e ele não foi usada em nenhuma outra transação. Então ela deve estar na lista de outputs transactions não usados
- Se uma transação tem algum output que foi gasto. Ele não pode ser usado novamente.
- Provar que transações e mensagens ids são únicos

O que não será feito

- Modelo de cripto-moeda em que é possível algum tipo de fork. No bitcoin, é possível que exista algum tipo de fork temporário
- Pool de transações. Sua utilidade é apenas para guardar as transações que ainda não foram adicionadas na blockchain. Isso pode ser feito fora do protocolo principal
- Otimização e protocolos RPC. O objetivo do projeto é definir as propriedades da cripto-moeda, não como ela será implementada e usada

Livros


Type-Driven
Development
with Idris
Edwin Brady
MANNING


O'REILLY
Programming
Bitcoin
Learn How to Program Bitcoin from Scratch
Jimmy Song

Bibliografia

- Programming Language Foundations in Agda
- Paper Modelling Bitcoin in Agda de Anton Setzer. Dept. of Computer Science, Swansea University, Singleton Park, Swansea SA28PP, UK