Very-efficient flipping of many coins

(between two parties)

Luís T. A. N. Brandão

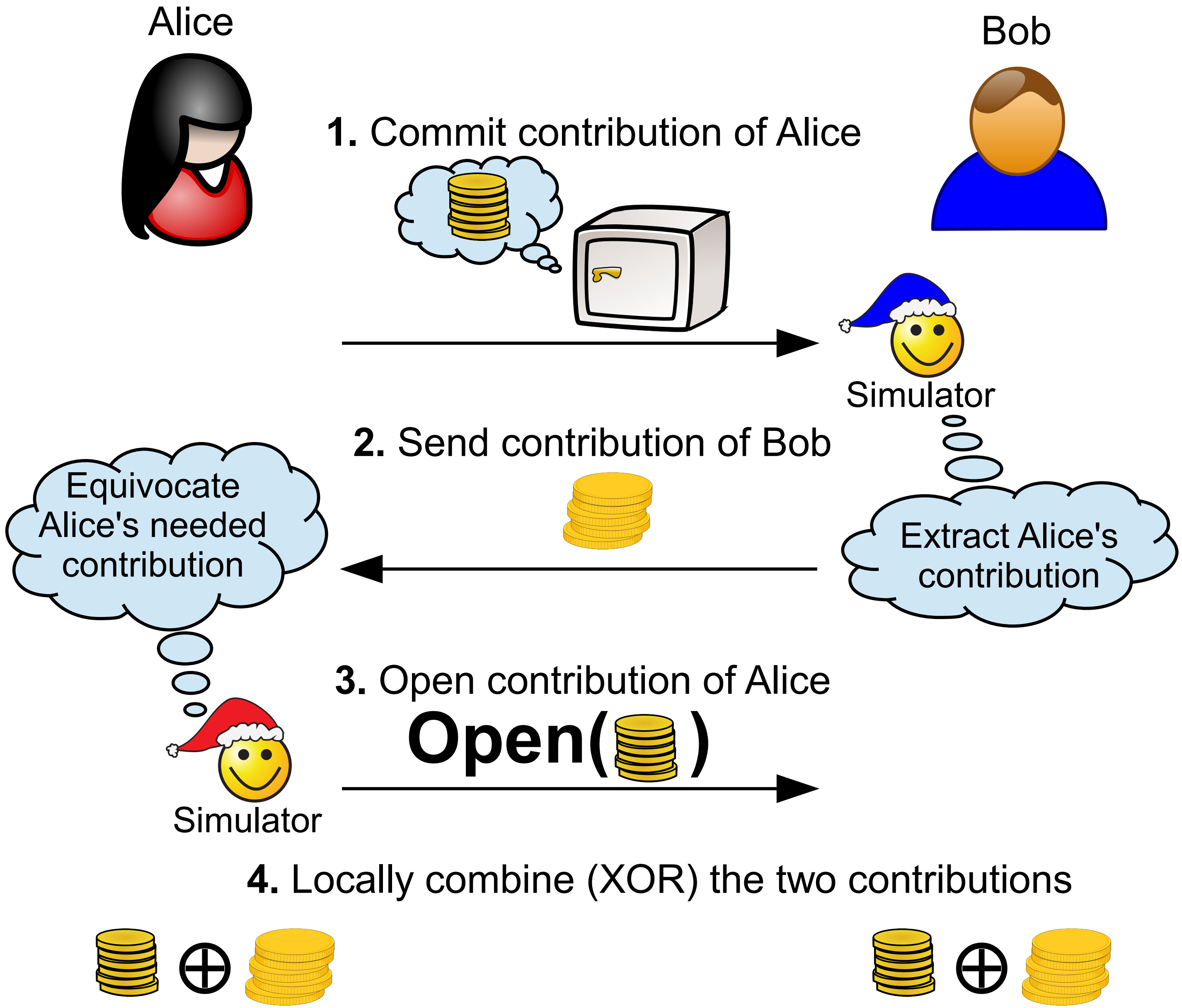
University of Lisbon and Carnegie Mellon University

Early presentation of results at ***rump session***
of Theory of Cryptography Conference 2014
February 25, San Diego, USA

(Minor adjustments on Feb 28, when preparing upload to Internet)

The author is a Ph.D. student at FCUL-DI and CMU-ECE. Support for this research was provided by the Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) through the Carnegie Mellon Portugal Program under Grant SFRH/BD/33770/2009.

The traditional coin-flipping template



Alice

Bob

1. Commit contribution of Alice

Simulator

2. Send contribution of Bob

Equivocate Alice's needed contribution

Extract Alice's contribution

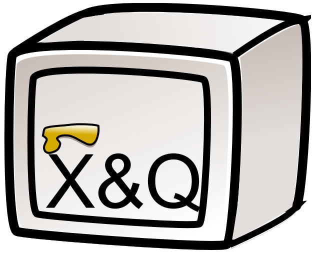
3. Open contribution of Alice

Open( )

Simulator

4. Locally combine (XOR) the two contributions

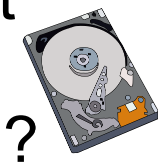
Commitment scheme needs be both ***extractable*** **(X)** and ***equivocable*** **(Q)**, i.e., be X&Q.



Several constructions exist ... with group-elements or group-operations in number or with size proportional to # coins

(Two recent independent works devise more efficient methods)

What if we want to flip MANY coins, e.g., 2TB?

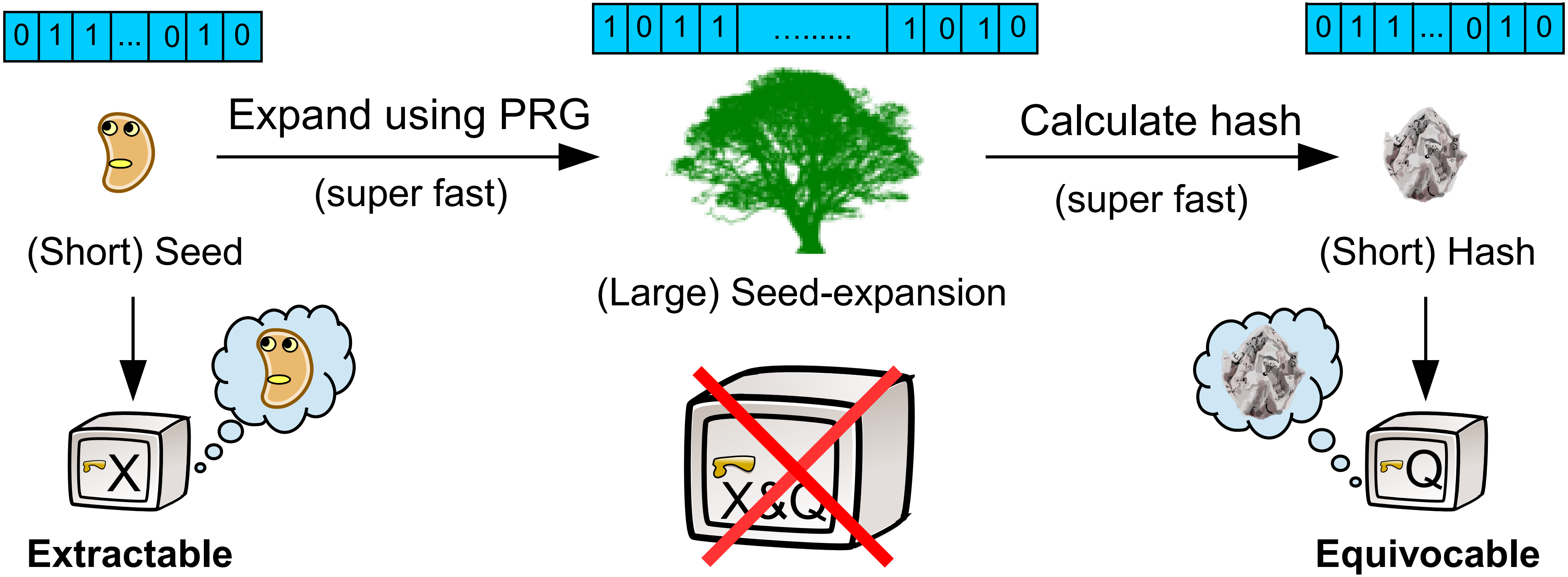


A new approach

Can we achieve a BIG X&Q commitment using only:
- a FEW SMALL X-commits and a FEW SMALL Q-commits;
- and symmetric primitives (PRG, hash function, XORs)

?
Yes!

An initial intuition



0 | 1 | 1 | ... | 0 | 1 | 0

1 | 0 | 1 | 1 | .... | 1 | 0 | 1 | 0

0 | 1 | 1 | ... | 0 | 1 | 0

Expand using PRG

(super fast)

Calculate hash

(super fast)

(Short) Seed

(Large) Seed-expansion

(Short) Hash

Extractable commitment

Extractable and Equivocable commitment

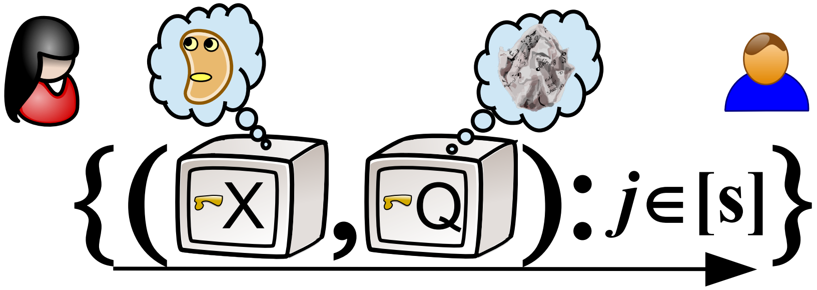
Equivocable commitment

One-pass simulatable coin-flipping

(cut-and-choose based technique)

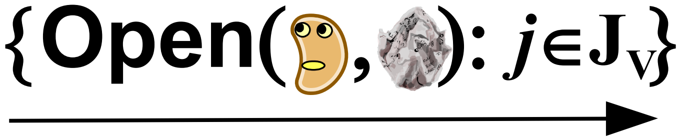
0. Prepare seeds and hashes

0.1. Alice commits seeds and hashes



0.2. Cut-and-choose: [s]=J_V + J_E

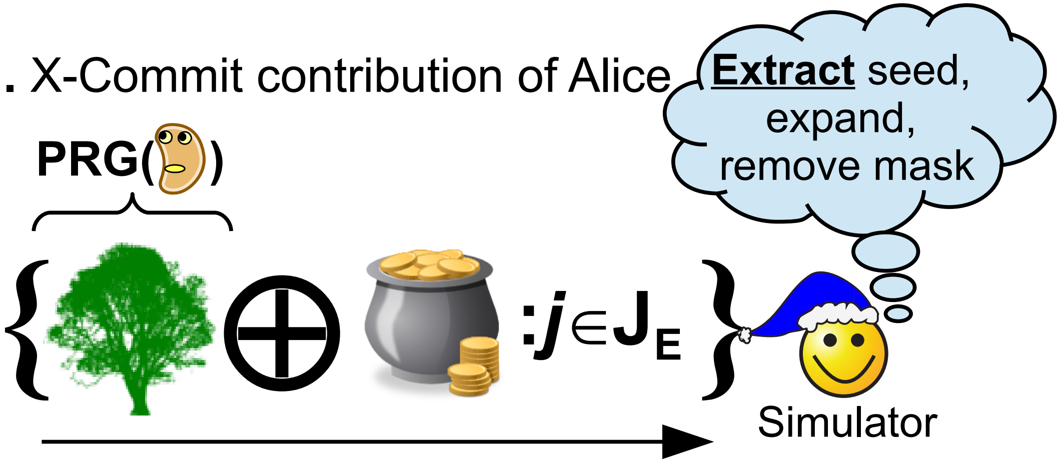
0.3. Alice opens *verification* instances (J_V)



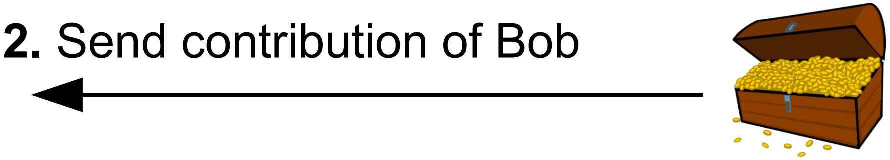
0.4. Bob verifies (J_V): = Hash(PRG( ))

⇒ high probability that a portion of remaining instances (J_E) are consistent

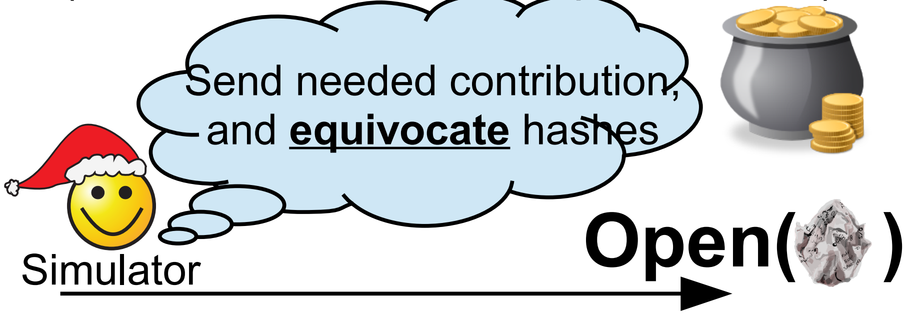
1. X-Commit contribution of Alice



2. Send contribution of Bob



3. Q-open contribution of Alice (send contribution and open hashes)



4. Locally combine contributions:

Summary

- A new approach for flipping many coins
  - Uses few X-commits of seeds and Q-commits of hashes
  - Leverages throughput of PRG and hash function

- Overlooked in this short presentation:
  - Verifiability condition for simulator to check that extracted hash is consistent with masked contribution.
  - How to reduce communication, by fragmenting the contribution using an efficient *Information dispersal algorithm* (and respective reconstruction).
  - Probabilities associated with the cut-and-choose.
  - A much simpler solution exists if rewinding is allowed.

Thank you for your attention!

Very-efficient flipping of many coins

(soon to be on eprint)

lbrandao at {alunos.fc.ul.pt, cmu.edu}

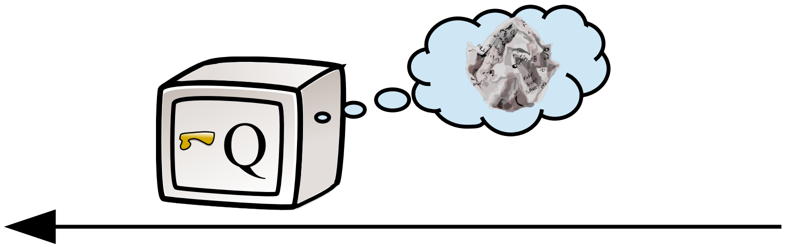
The images used in this presentation were taken from openclipart.org and clker.com, with the expectation of being in the public-domain and available for free usage.

If rewinding is OK, use another template

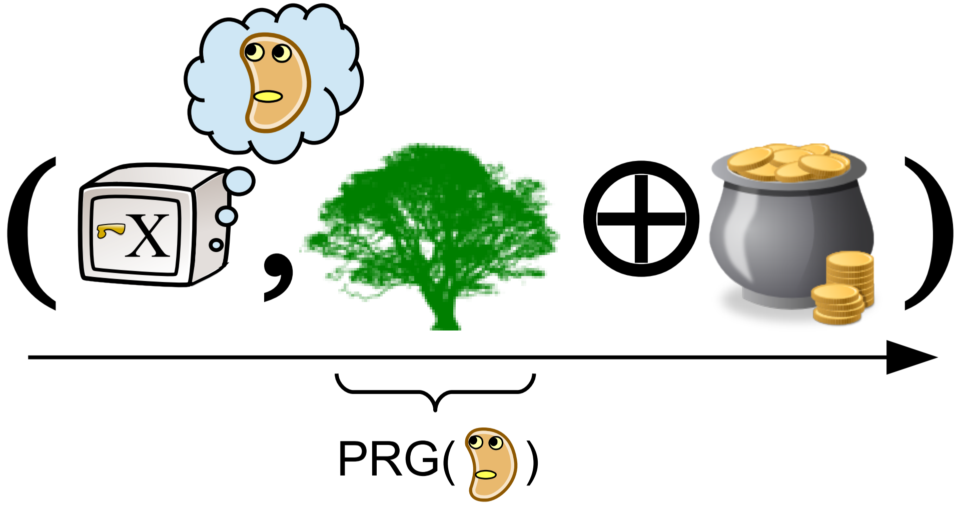
(Slide prepared for rump session but not shown due to time constraint)

Alice Bob

1. Bob Q-commits hash of his contribution

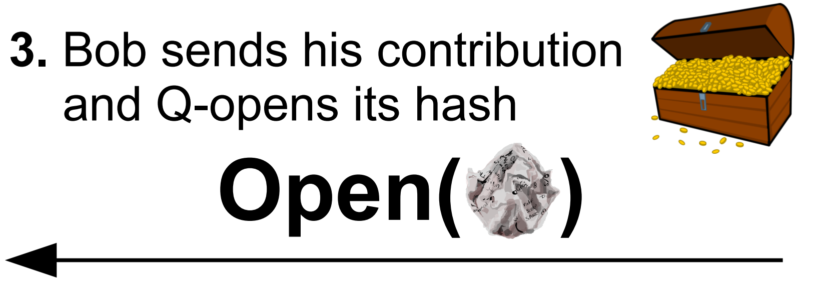


2. Alice X-commits seed of a mask, and sends her masked contribution

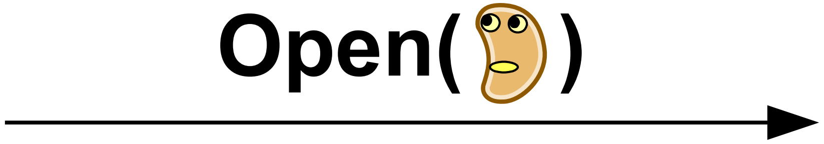


Alice Bob

3. Bob sends his contribution and Q-opens its hash



4. Alice opens the seed of her contribution mask



5. Locally unmask contribution of Alice and combine contributions