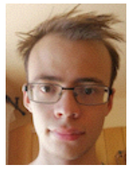
Byzantine Fault Tolerance using Entangled Quantum States

András Pályi
Budapest University of Technology and Economics (BME)

in collaboration with:

Zoltán Guba, Ákos Budai
BME



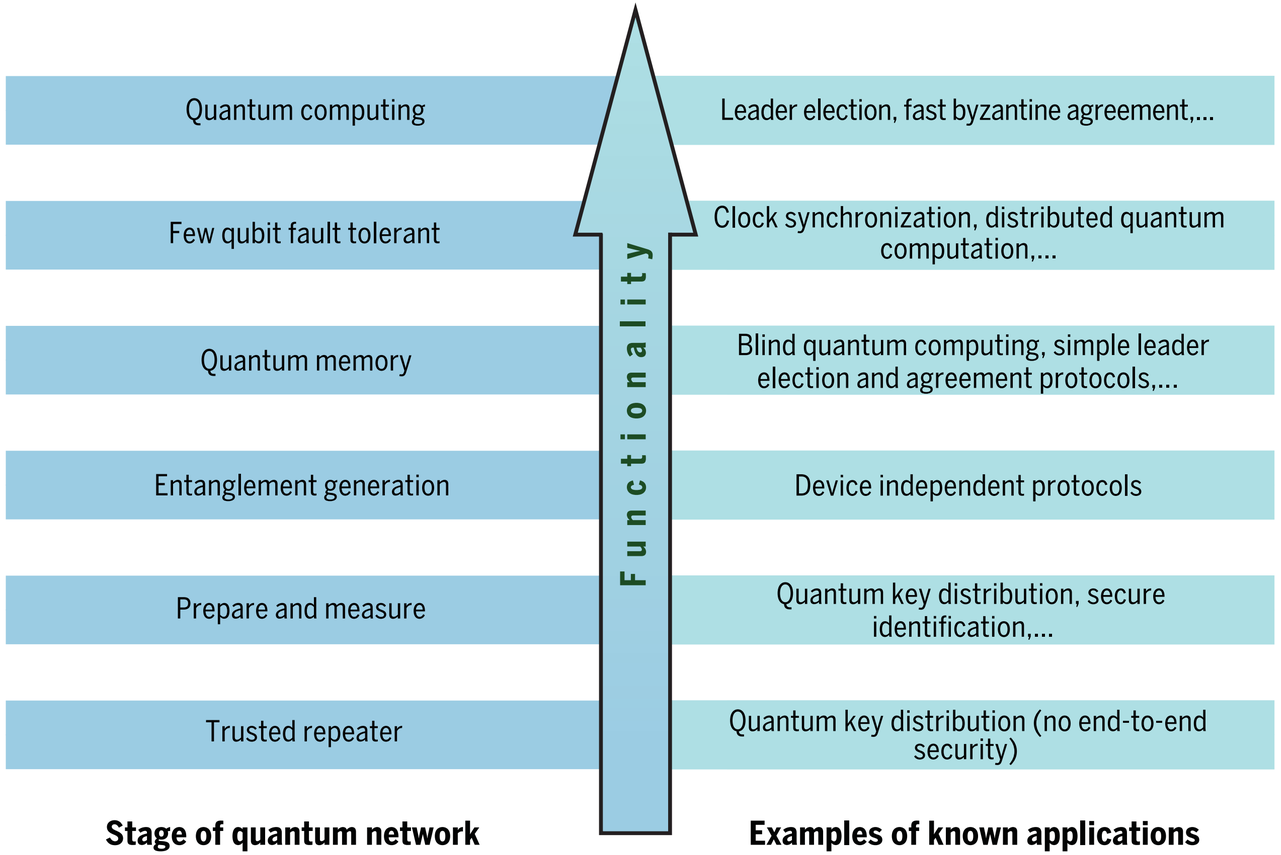
István Finta, Lóránt Farkas
Nokia Bell Labs, Budapest

Zoltán Zimborás
Wigner RCP Budapest

Quantum internet: A vision for the road ahead

Science 2018

Stephanie Wehner*, David Elkouss, Ronald Hanson

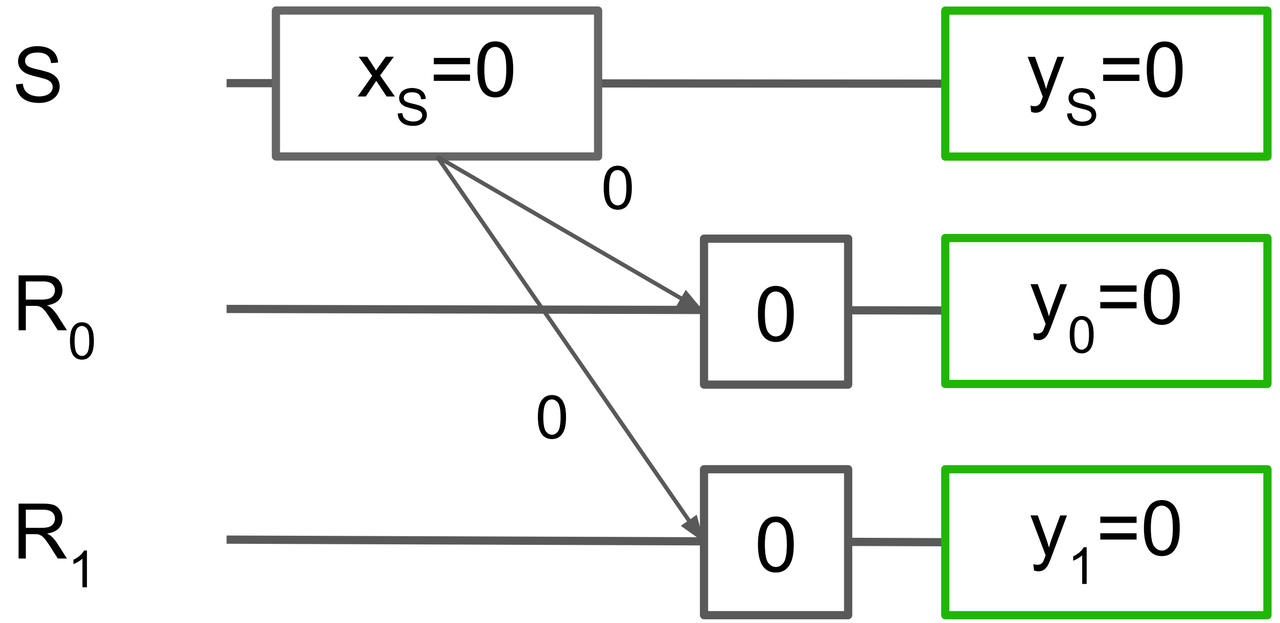


PROJECT FINANCED FROM THE NRDI FUND
MOMENTUM OF INNOVATION

ELTE Bolyai Kollégium
April 7, 2021

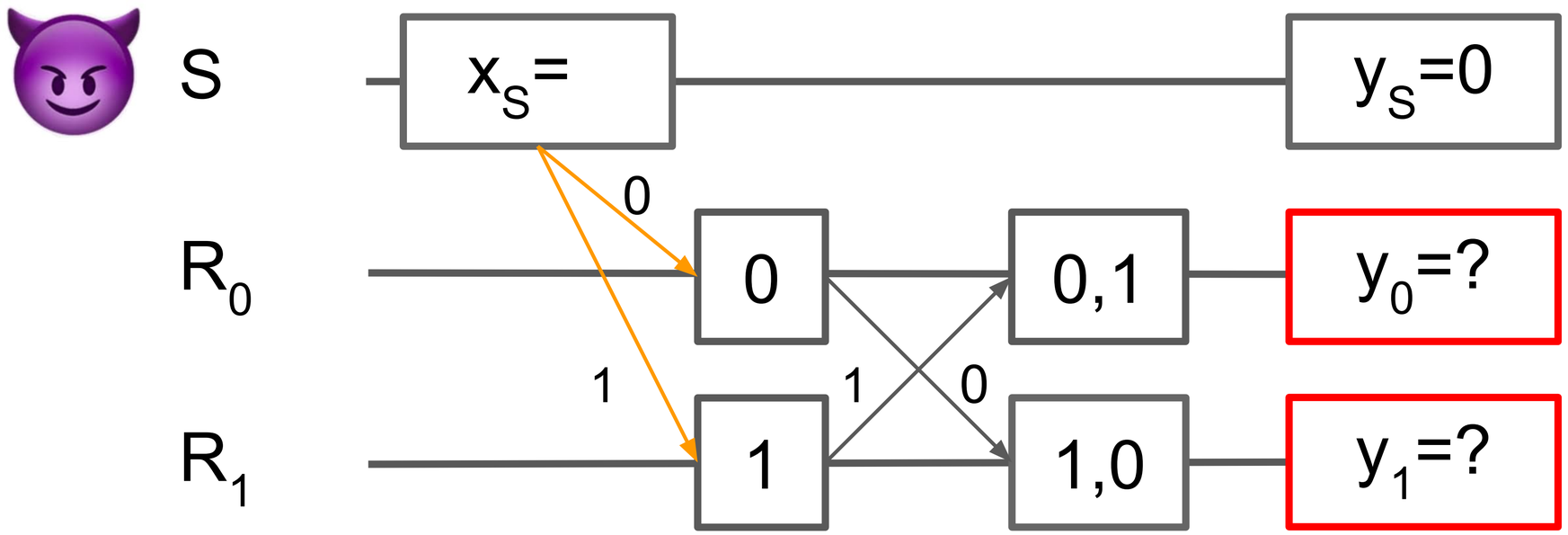
The Task

Communicate a single bit
among multiple components



The Task

Reach consensus
among multiple components
in the presence of faulty components



in the original framework (Pease, Lamport, Shostak, ~1980), n = 3, t = 1 is unsolvable

Functionality #1:
Byzantine Fault Tolerance or `Broadcast'

All Correct | S Faulty | R0 faulty | R1 faulty

1. All correct components get the same output bit.
2. If Sender is correct, then the output of all correct components is the data sent by Sender.

S	R_0	R_1	B.	B.	B.	B.
0	0	0	✓	✓	✓	✓
0	0	1				✓
0	1	0			✓	
0	1	1		✓		
1	0	0		✓		
1	0	1			✓	
1	1	0				✓
1	1	1	✓	✓	✓	✓

Functionality #2: Weak Broadcast

All Correct | S Faulty | R0 faulty | R1 faulty

S	R_0	R_1	W.B.	W.B.	W.B.	W.B.
0	0	0	✓	✓	✓	✓
0	0	1				✓
0	0	$\perp$		✓		✓
0	1	0			✓	
0	1	1		✓		
0	1	$\perp$		✓		
0	$\perp$	0		✓	✓	
0	$\perp$	1		✓		
0	$\perp$	$\perp$		✓		
1	0	0		✓		
1	0	1			✓	
1	0	$\perp$		✓		
1	1	0				✓
1	1	1	✓	✓	✓	✓
1	1	$\perp$		✓		✓
1	$\perp$	0		✓		
1	$\perp$	1		✓	✓	
1	$\perp$	$\perp$		✓		

1. If no component is faulty, then the protocol achieves broadcast.
2. If any correct component decides on an output in {0,1}, then all correct components decide on the same output or `confused' ($\perp$).
3. If the sender is correct, then all correct components decide on the sender's input.

Weak Broadcast(3,1) relevance: building block for Broadcast

Byzantine Agreement - variations, figures of merits

Classical/deterministic Byzantine Agreement

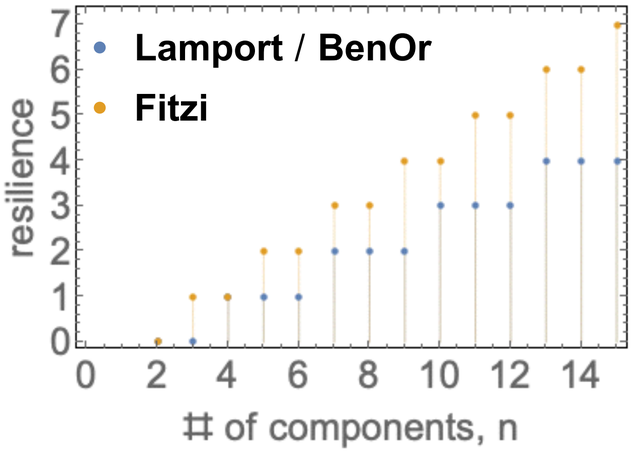
Lamport et al. ~ 1980

Resilience: Can deal with 1 faulty out of 4 components, t < n/3 in general

Functionality: Broadcast

Communication complexity: requires much communication, inefficient (~n! messages)

more efficient alternatives: Garay-Moses, PBFT, Raft, Paxos

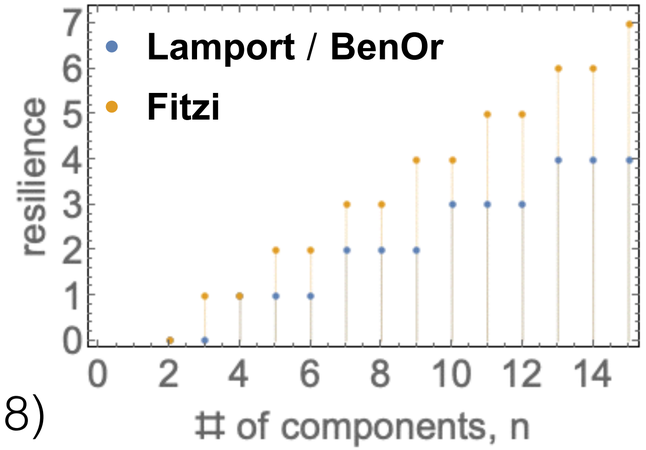


Byzantine Agreement - variations, figures of merits

Classical/deterministic Byzantine Agreement

Lamport et al. ~ 1980

Resilience: Can deal with 1 faulty out of 4 components, t < n/3 in general
Functionality: Broadcast
Communication complexity: requires much communication, inefficient (~n! messages)
more efficient alternatives: Garay-Moses, PBFT, Raft, Paxos



Quantum/randomized Byzantine agreement

Ben-Or & Hassidim 2005

Resilience: Can deal with 1 faulty out of 4 components, t < n/3 in general
Functionality: Broadcast
Communication complexity: efficient (single round).
Quantum resources: complicated quantum state required (Taherkhani et al. 2018)

Fitzi et al. 2002, Cabello 2003, Gaertner et al. 2008

Resilience: Can deal with 1 faulty out of 3 components, t < n/2 in general
Functionality: Broadcast [via Weak Broadcast(n=3,t=1)]
Communication complexity: efficient.
Quantum resources: relatively simple 4-qubit state required; how many?

our goal for today: simulate resource requirement of Weak Broadcast(n = 3,t=1)

How does the Fitzi-Cabello-Gaertner Weak Broadcast(3,1) protocol work?

1. take n = 3 components, S, R0 and R1
2. allow for at most 1 faulty component (t = 1)
3. assume there is a *global correlated random number generator*
4. for each data bit to be broadcasted, we generate N = 12 random triplets

probability	S	R0	R1
1/3	0	0	0
1/3	1	1	1
1/6	2	1	0
1/6	2	0	1

	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

How does the Fitzi-Cabello-Gaertner Weak Broadcast(3,1) protocol work?

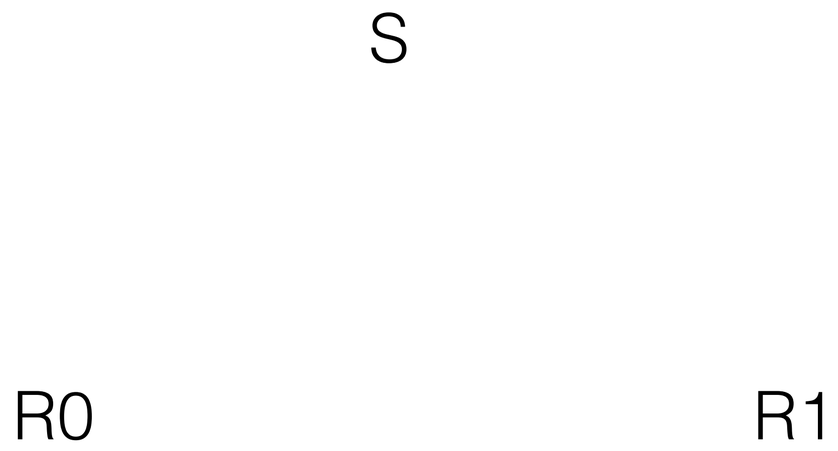
1. $S \to R_0,\ R_1\colon\ x_S, \sigma_S = \{i \in \{1,\ldots,N\} : Q_S[i] = x_S x_S\}, R_k$ receive $\{x_k, \sigma_k\}$
   $S : y_S = x_S$

2. R_k : if $(|\sigma_k| \geq m_0 \cdot N) \wedge (\{i \in \sigma_k : Q_k[i] = x_k\} = \emptyset)$ then $y_k = x_k$ else $y_k = \perp$ fi

3. $R_0 \to R_1 : y_0, \rho_0 = \{i \in \sigma_0 : Q_0[i] = 1 - x_0\}; R_1$: receives (y_{01}, ρ_{01})

4. $R_1\colon$ if $(\perp \neq y_{01} \neq y_1 \neq \perp) \wedge (|\rho_{01}| \geq m_0 \cdot N) \wedge$
   $\wedge(|\{i \in \rho_{01} : Q_1[i] = 1 - y_{01}\}| \geq \lambda \cdot |\rho_{01}|)$ then $y_1 := y_{01}$ fi

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

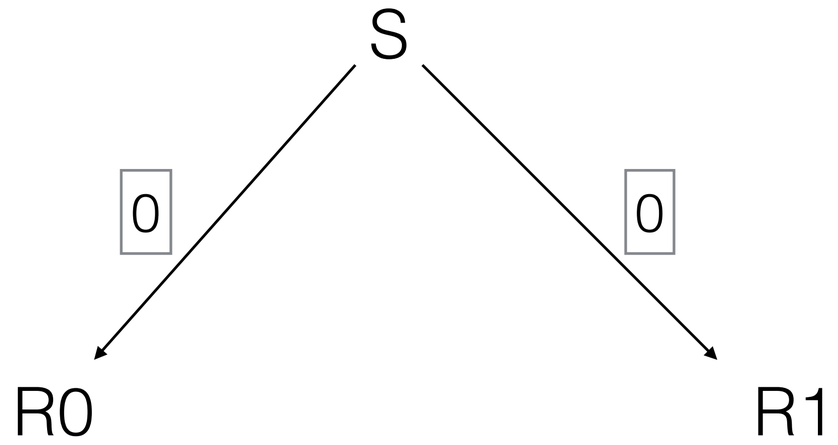


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S wants to broadcast data “0”

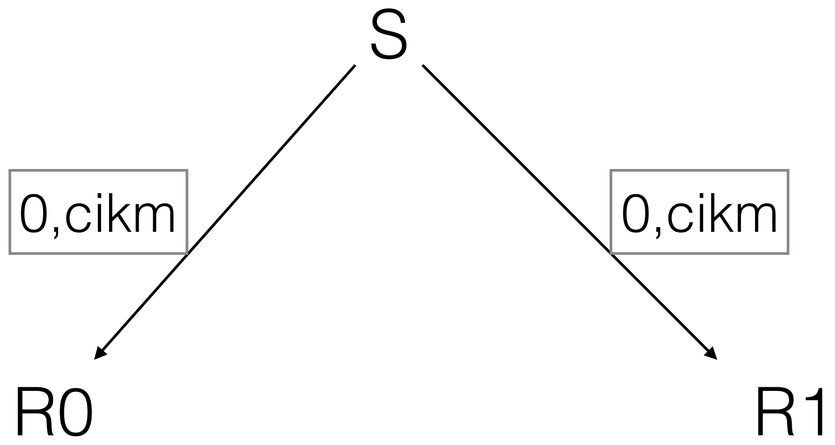


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct’ configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S also sends a `check string'

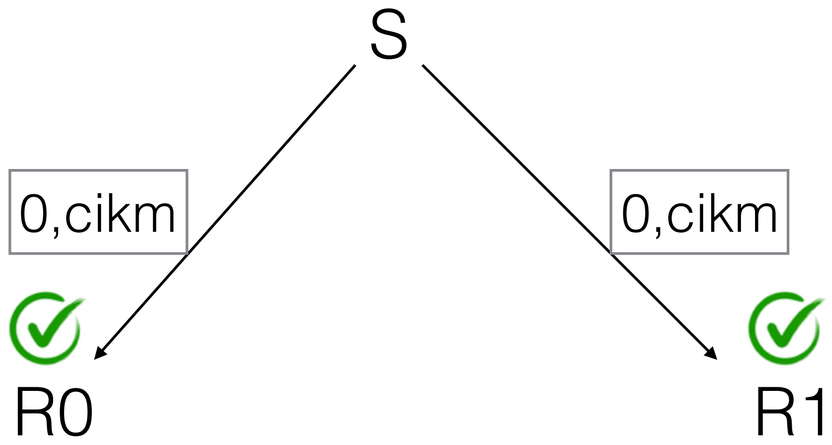


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

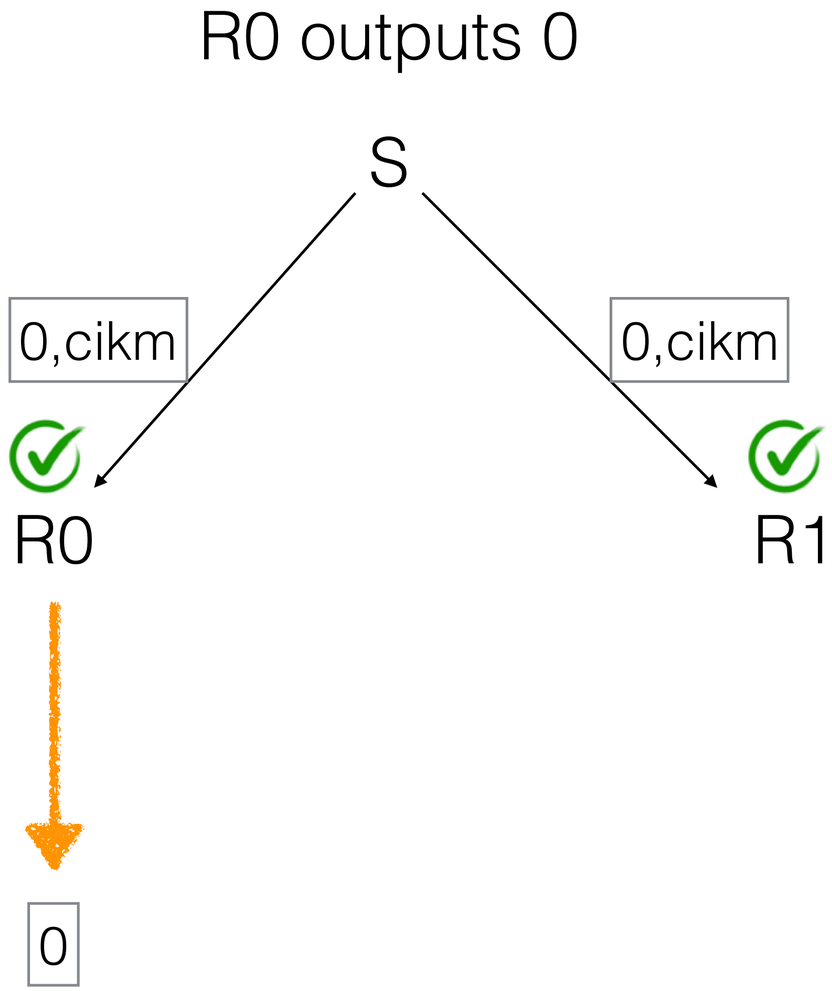
R0 and R1 checks the check string



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

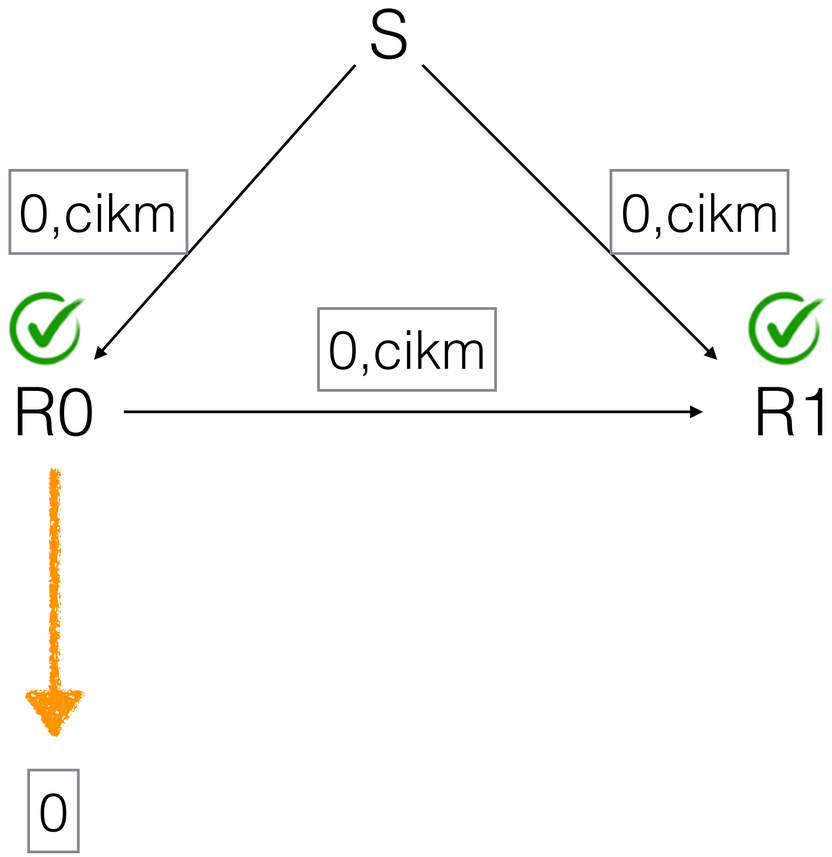


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

R0 sends data and secondary check string to R1

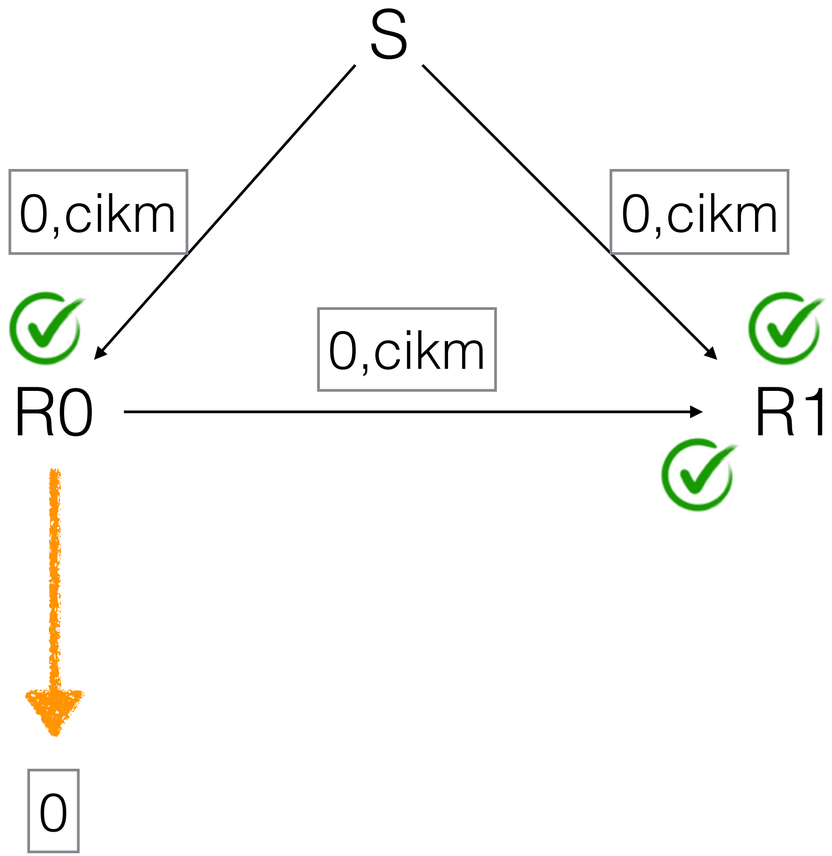


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

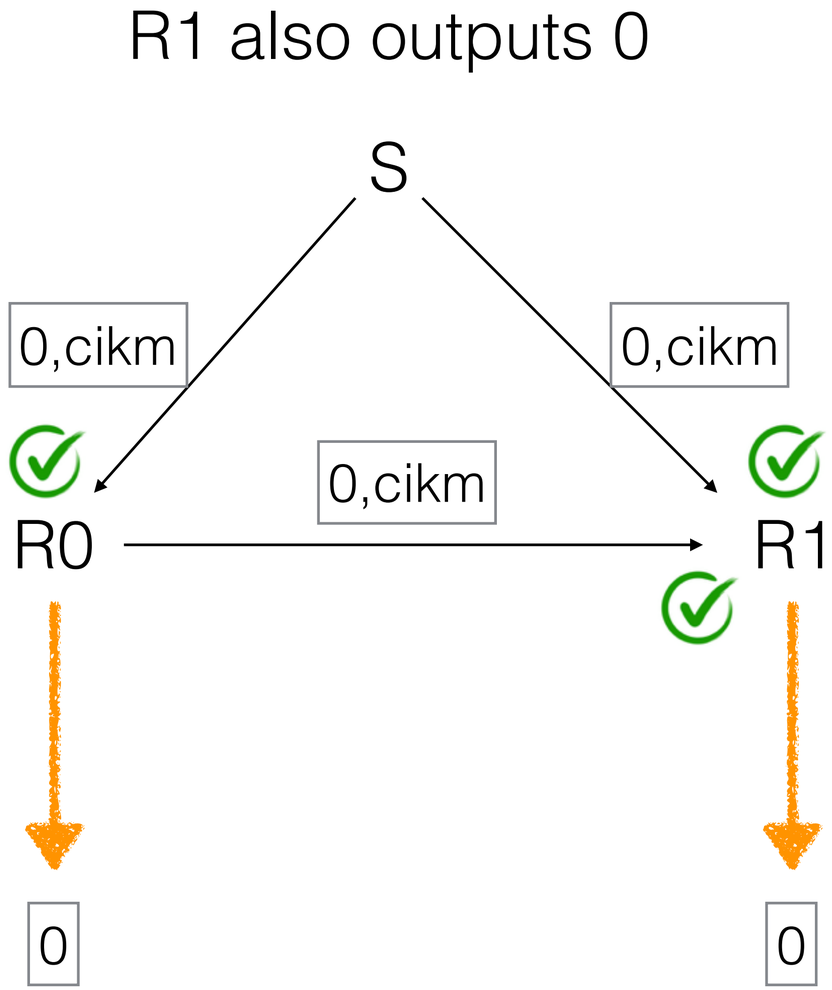
R1 checks the secondary check string



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct' configuration => protocol succeeds

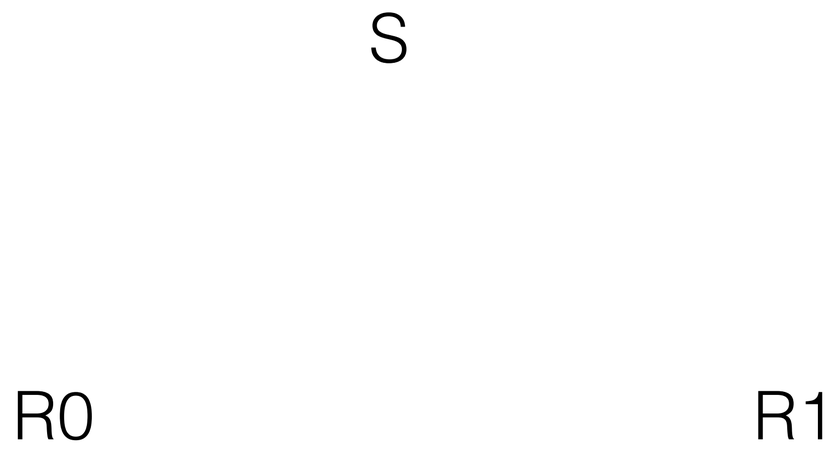
How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`All correct' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

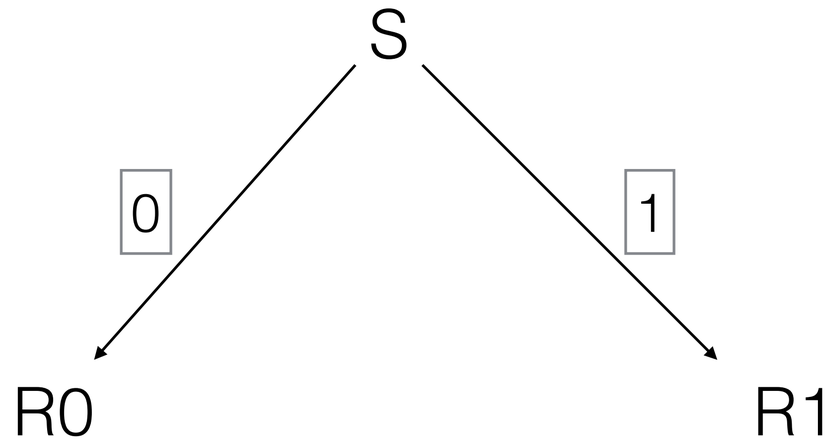


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S wants to mislead receivers

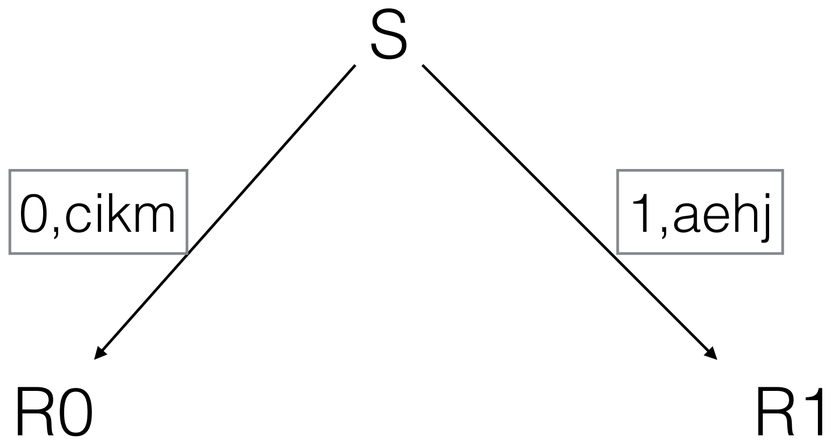


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S sends naive cheater's `check strings'

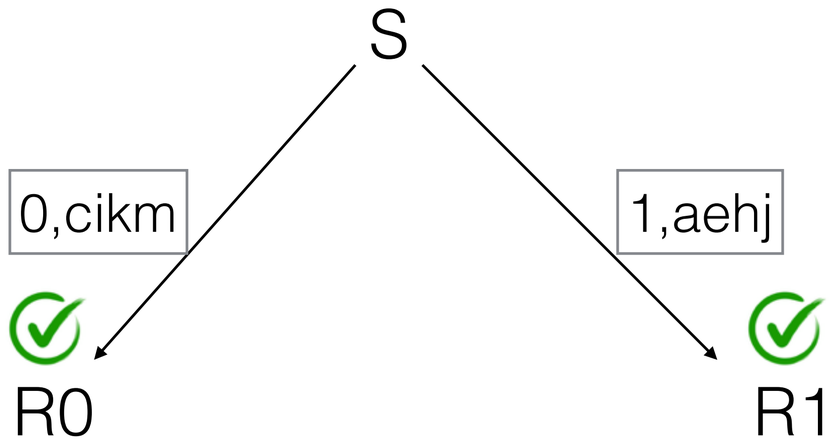


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

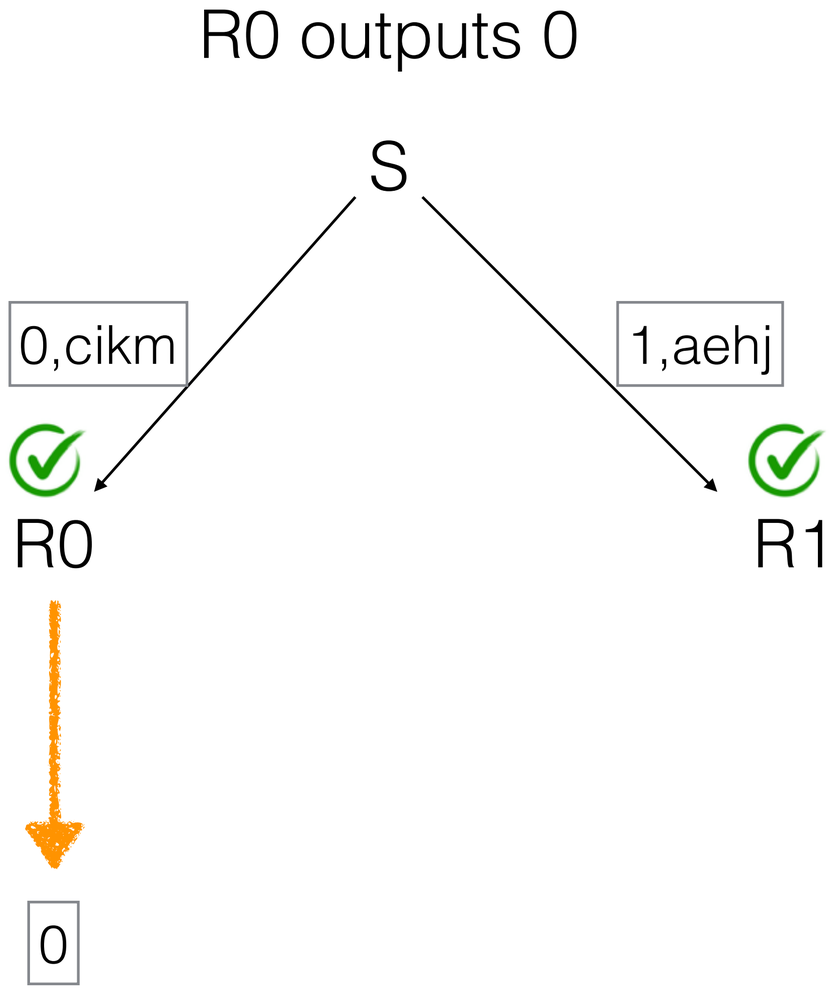
R0 and R1 check their check strings



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

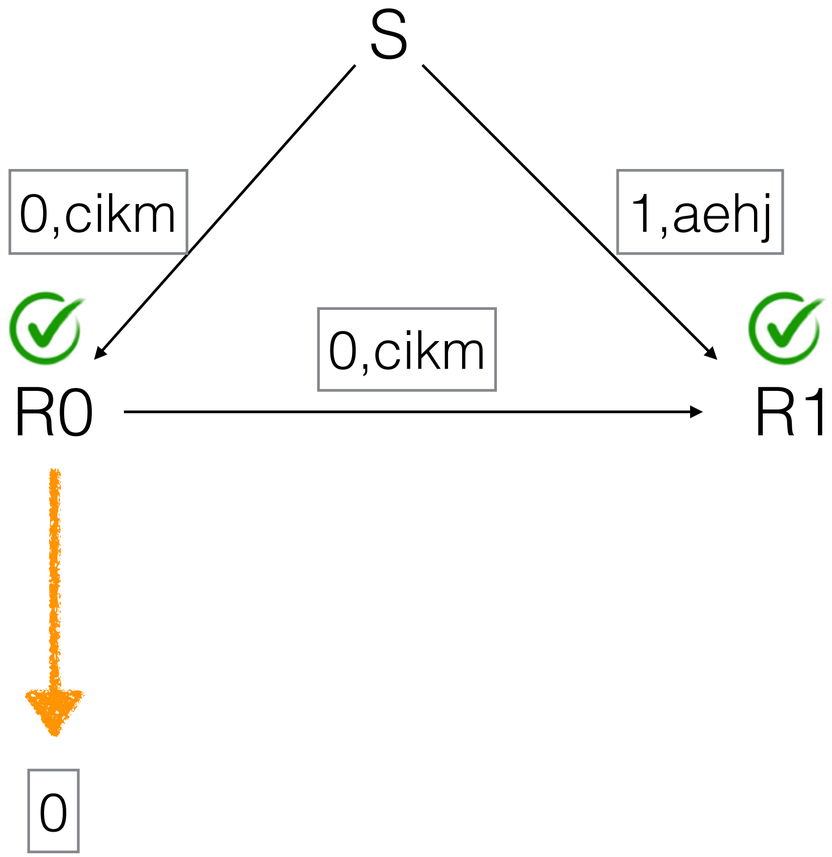


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

R0 sends data and secondary check string to R1

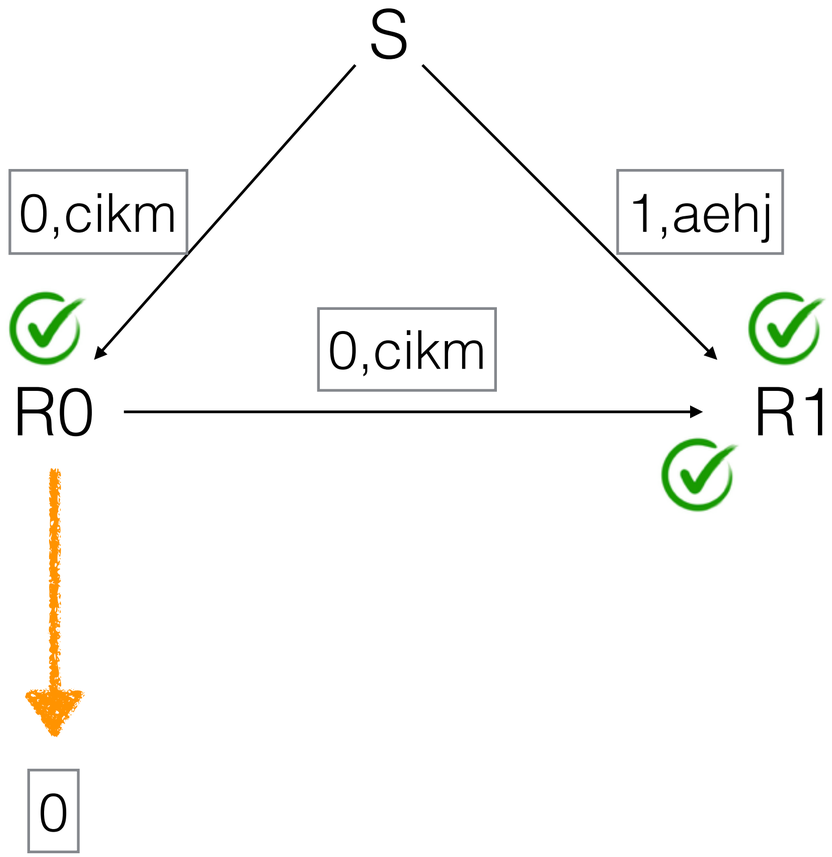


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

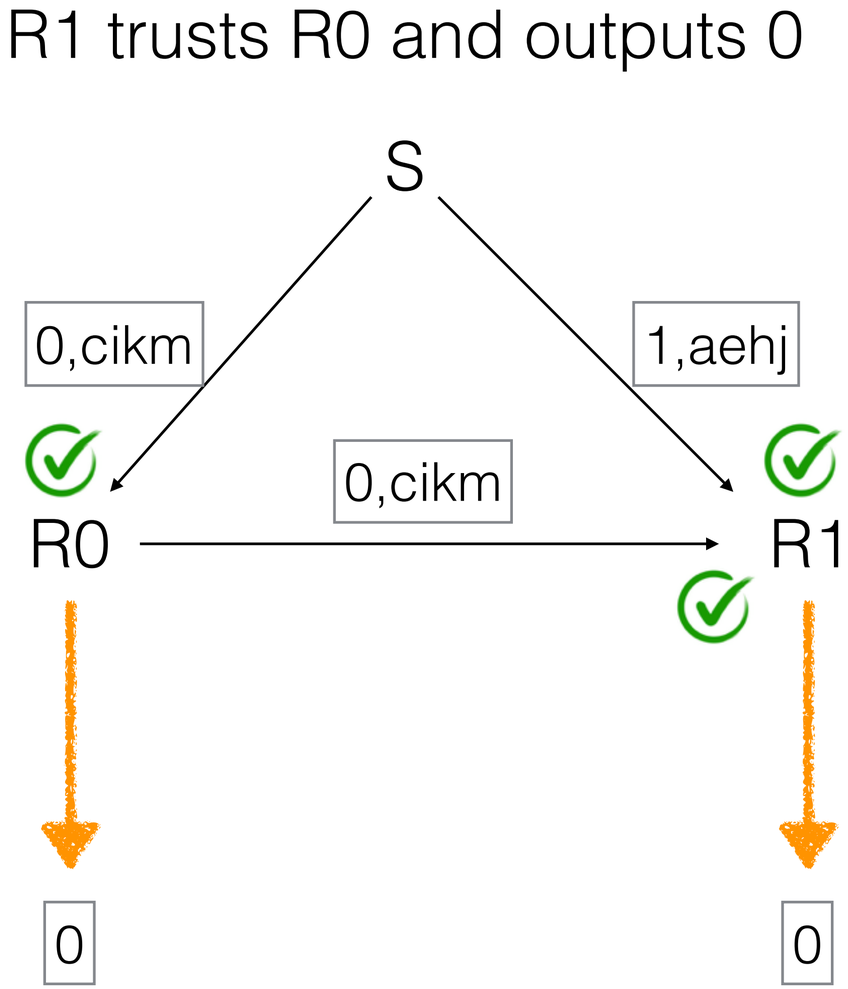
R1 checks R0's message for its internal consistency



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => protocol succeeds

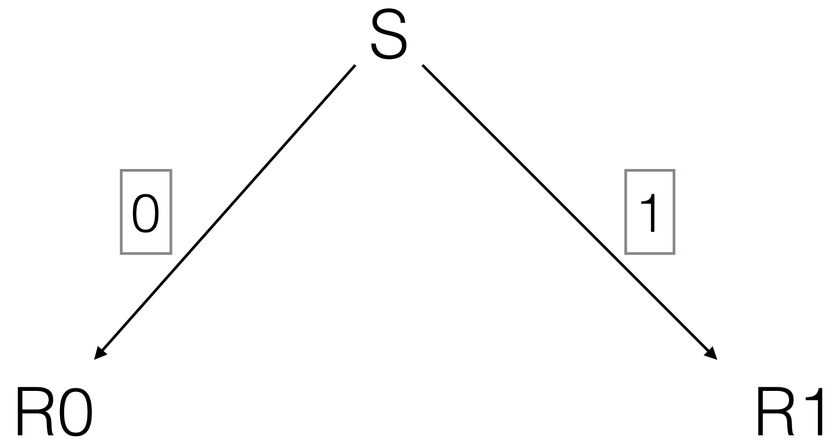
How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S wants to mislead receivers (by gambling)

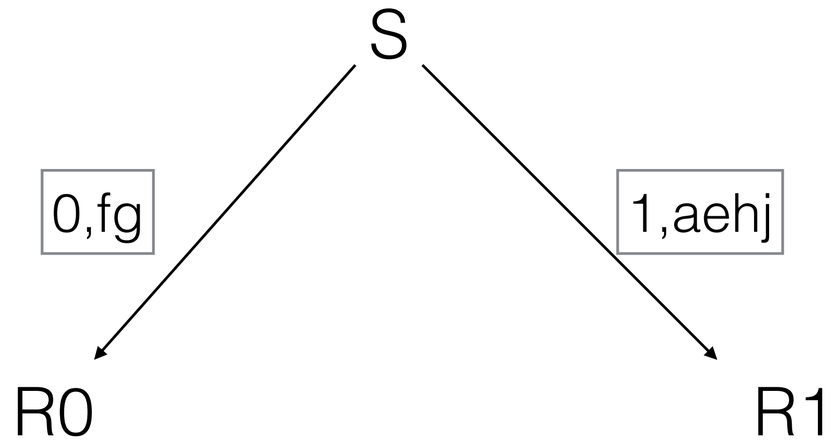


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S sends cheater's `check strings'

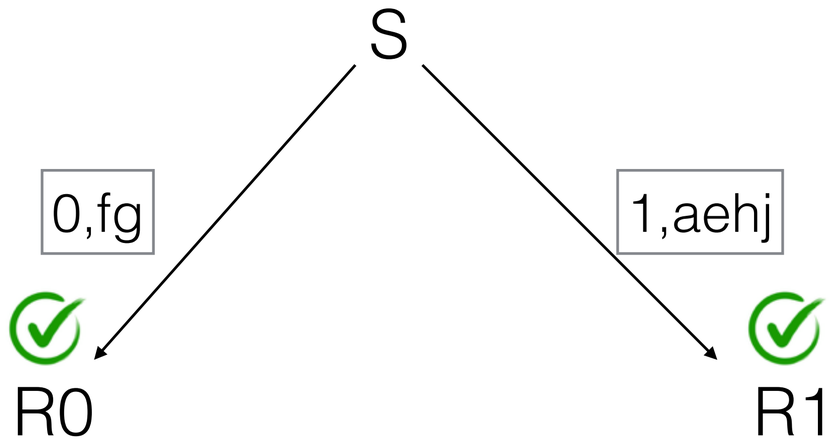


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

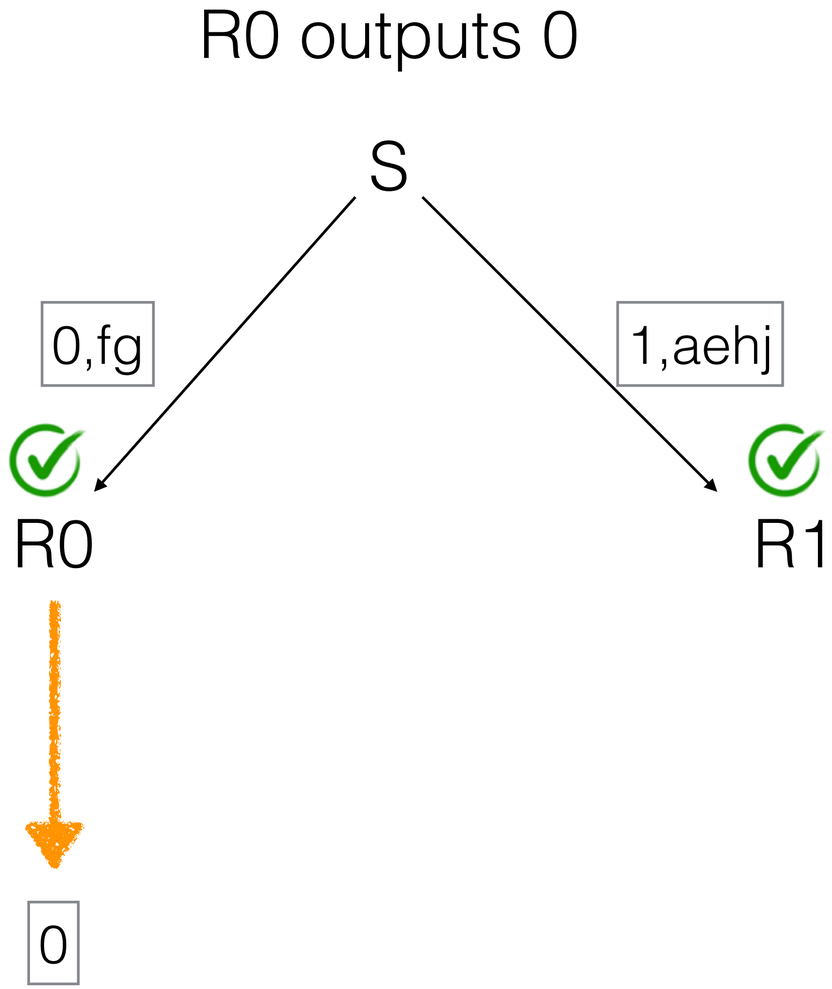
R0 and R1 check their check strings



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

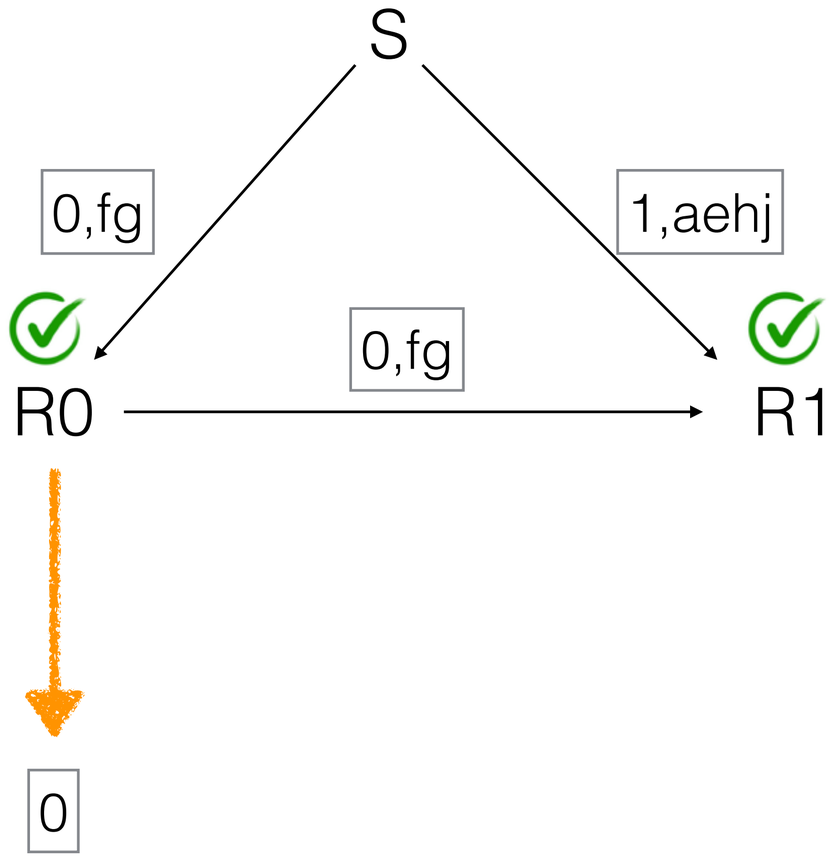


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

R0 sends data and secondary check string to R1

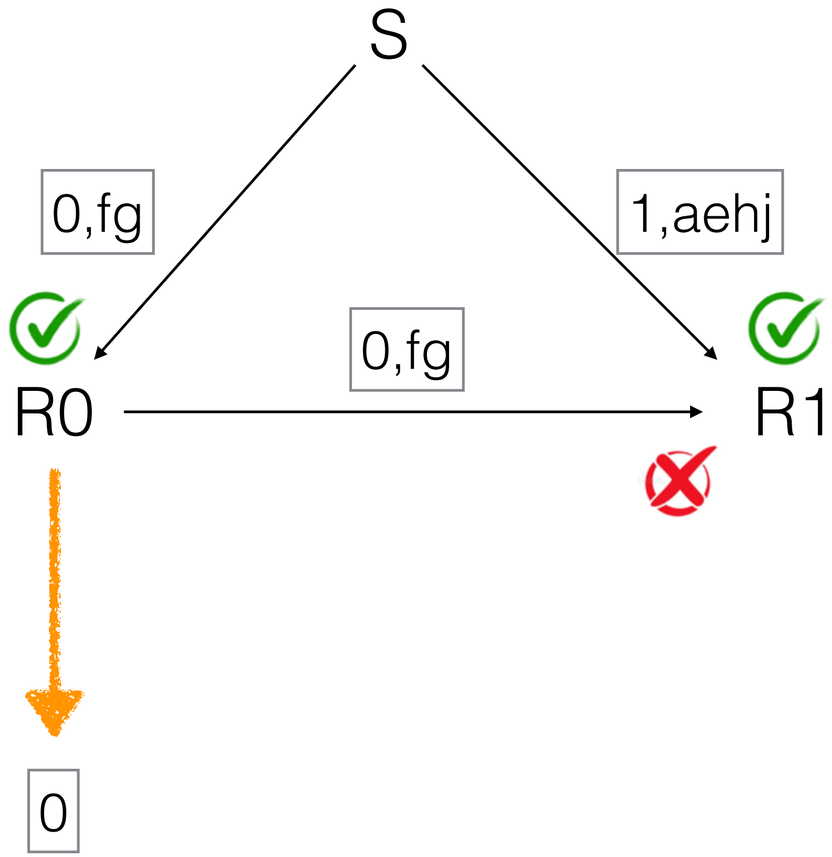


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

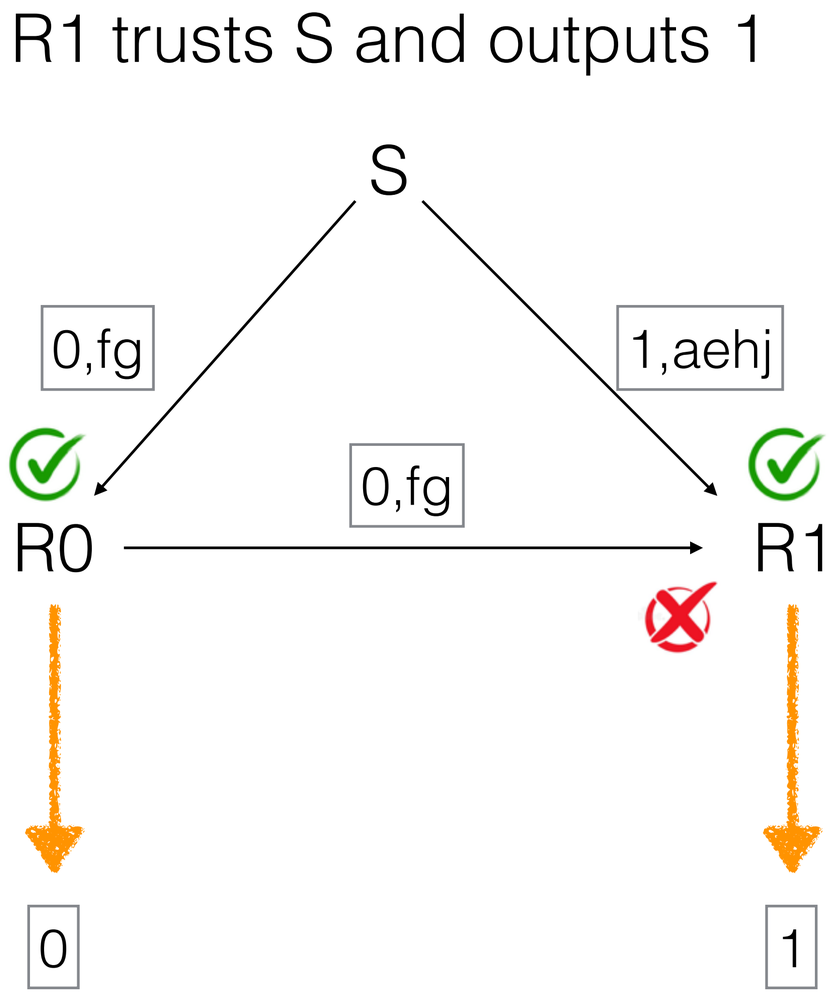
R1 checks R0's message for its internal consistency



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

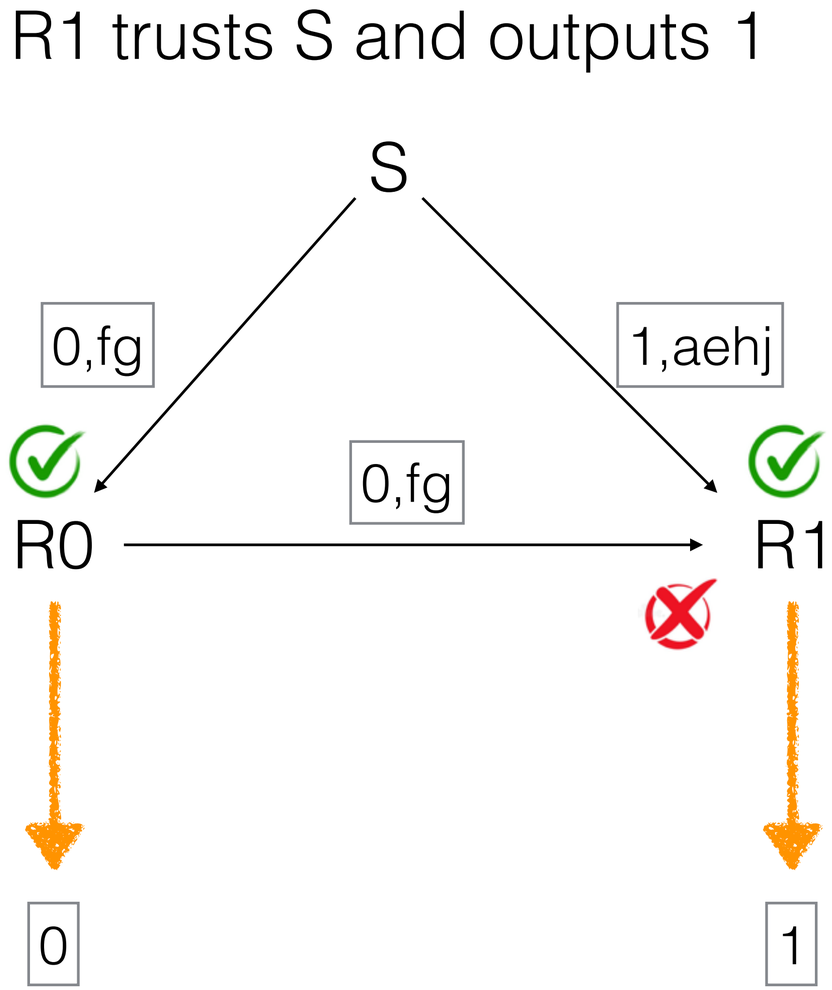
How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

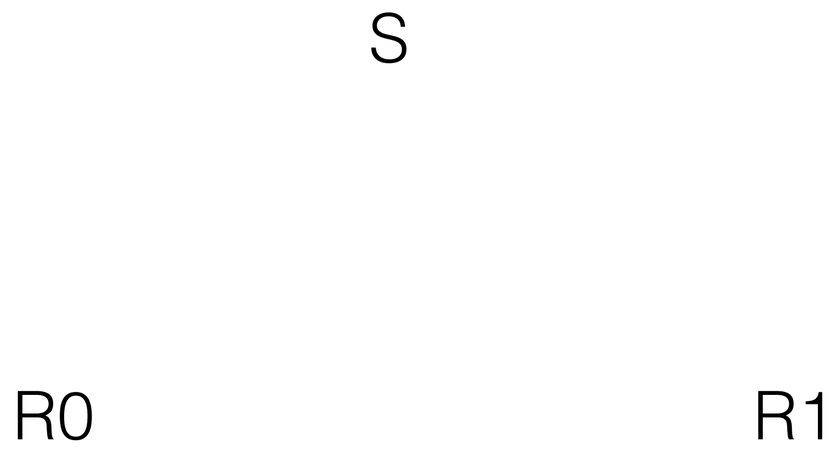
How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?



FAILURE, with exponentially small probability.
This is acceptable.

	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

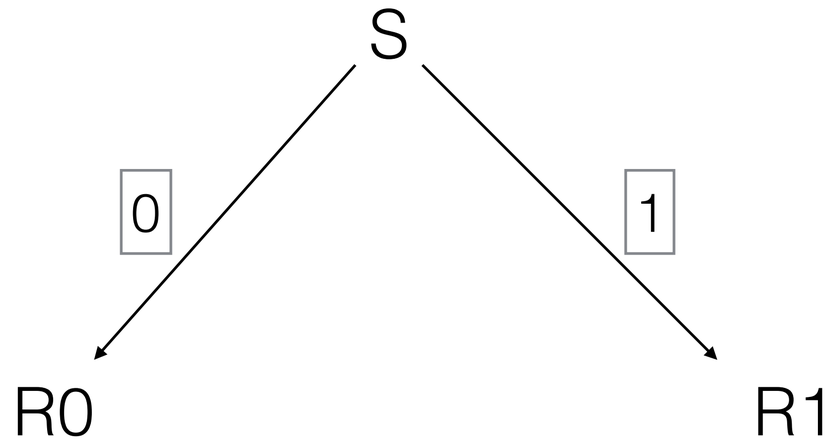


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S wants to mislead receivers (by gambling)

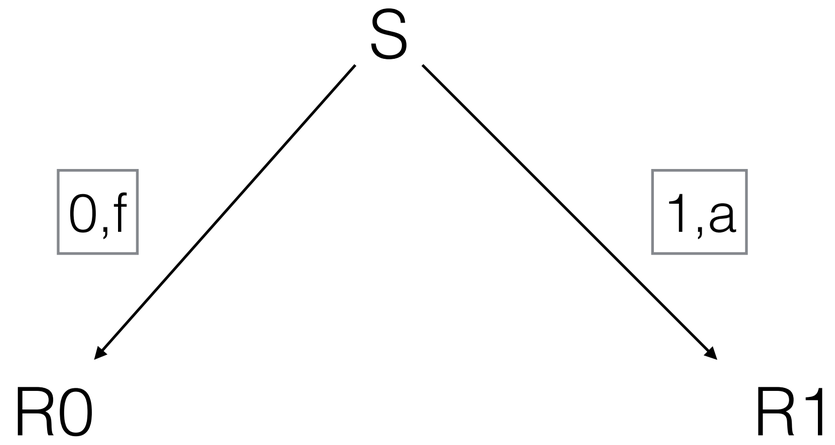


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

S sends very short cheater's `check strings'

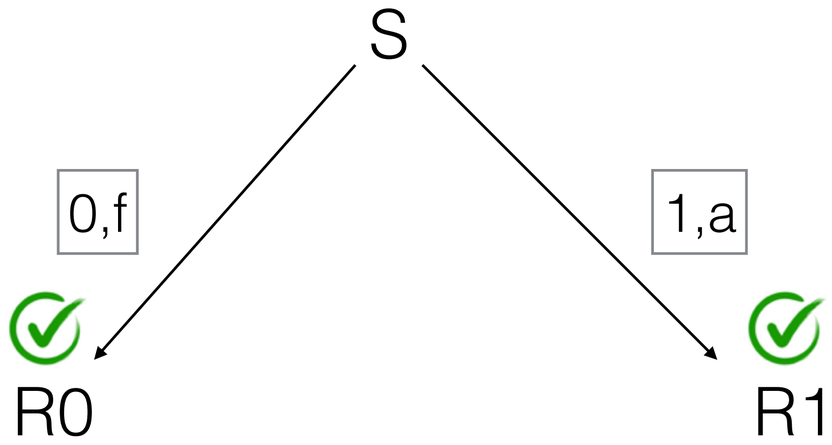


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

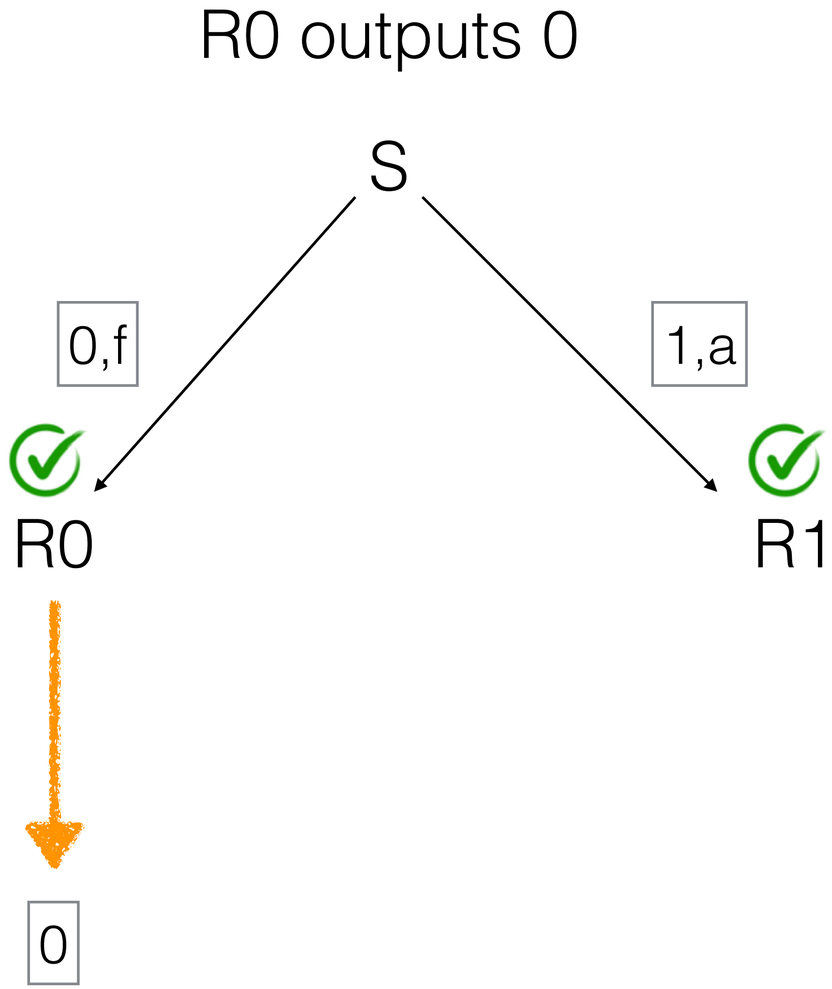
R0 and R1 check their check strings



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

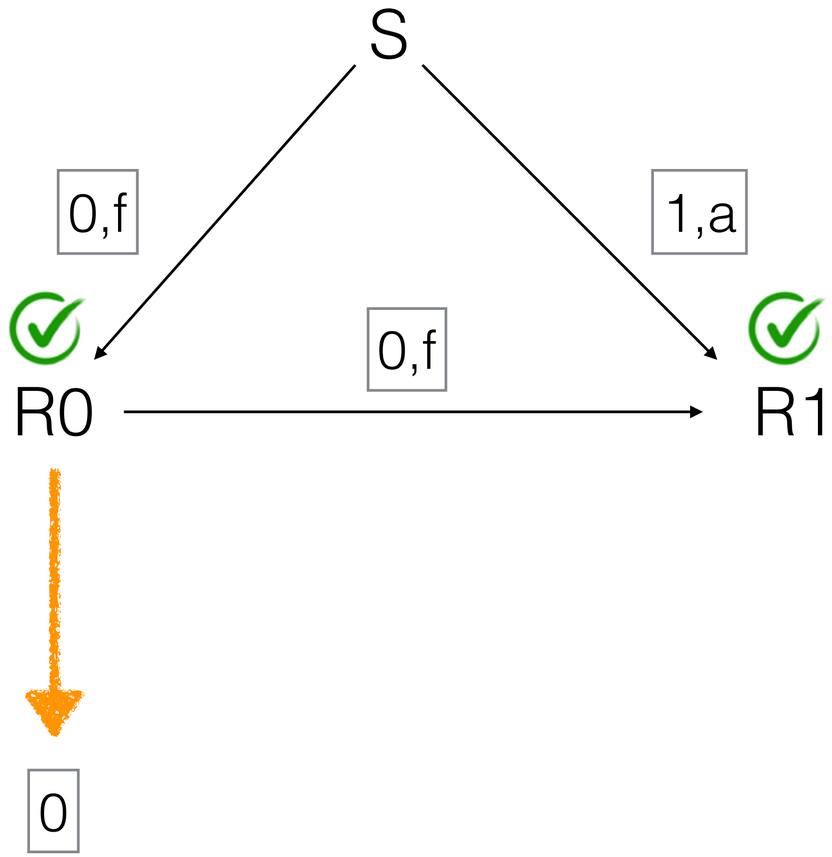


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

R0 sends data and secondary check string to R1

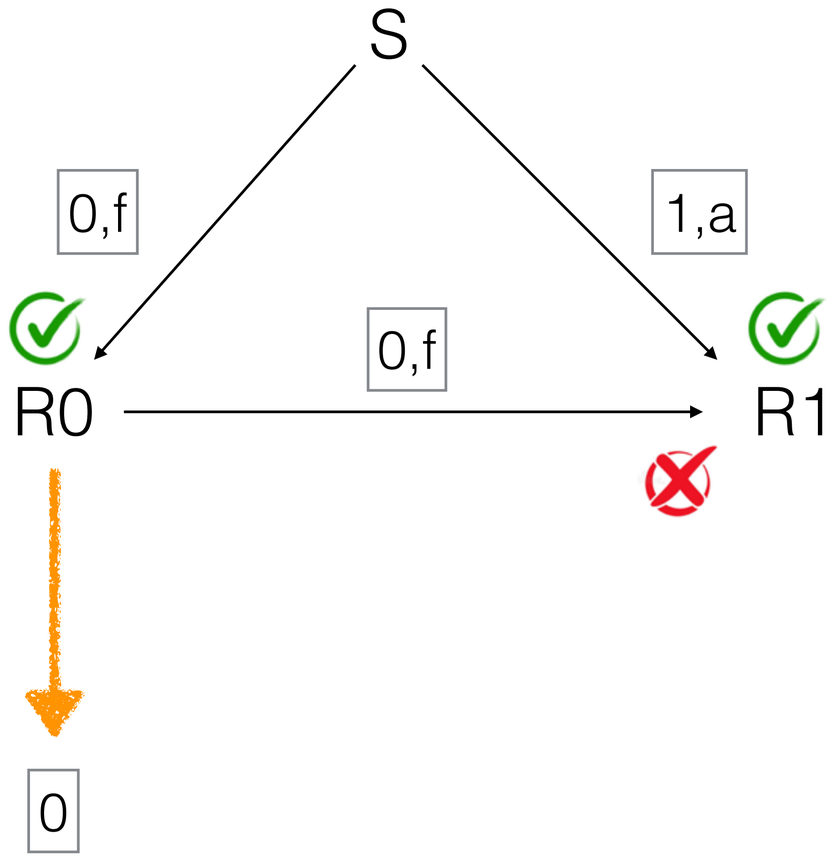


	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?

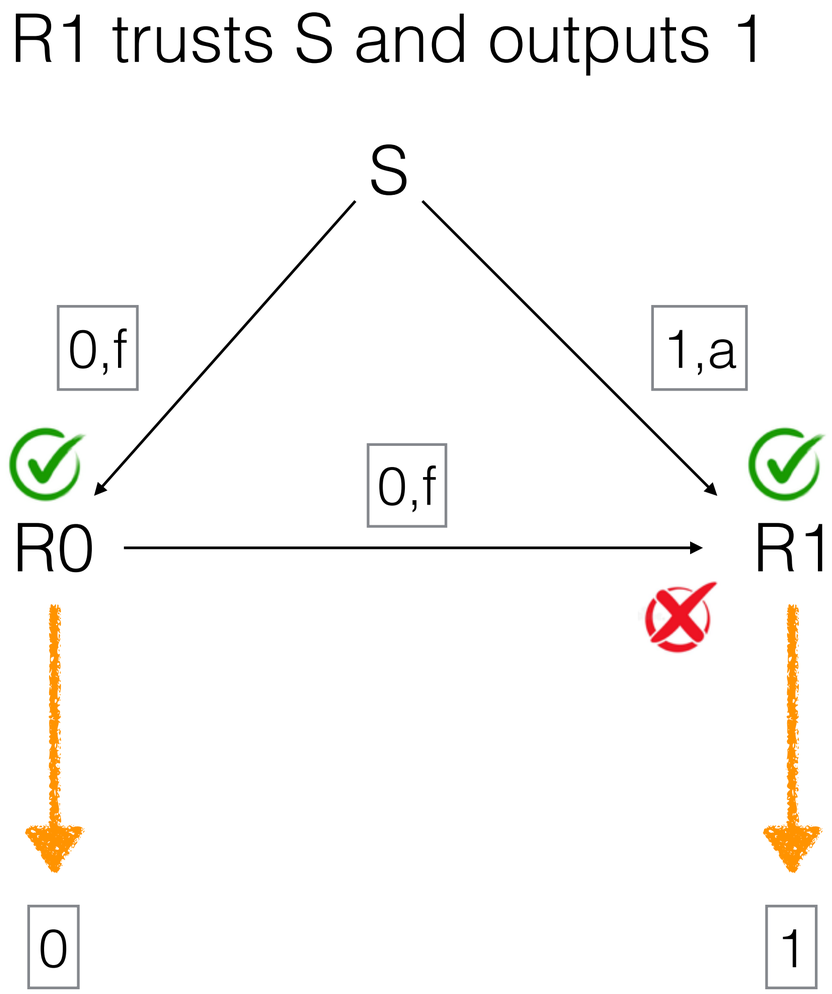
R1 checks R0's message for its internal consistency



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

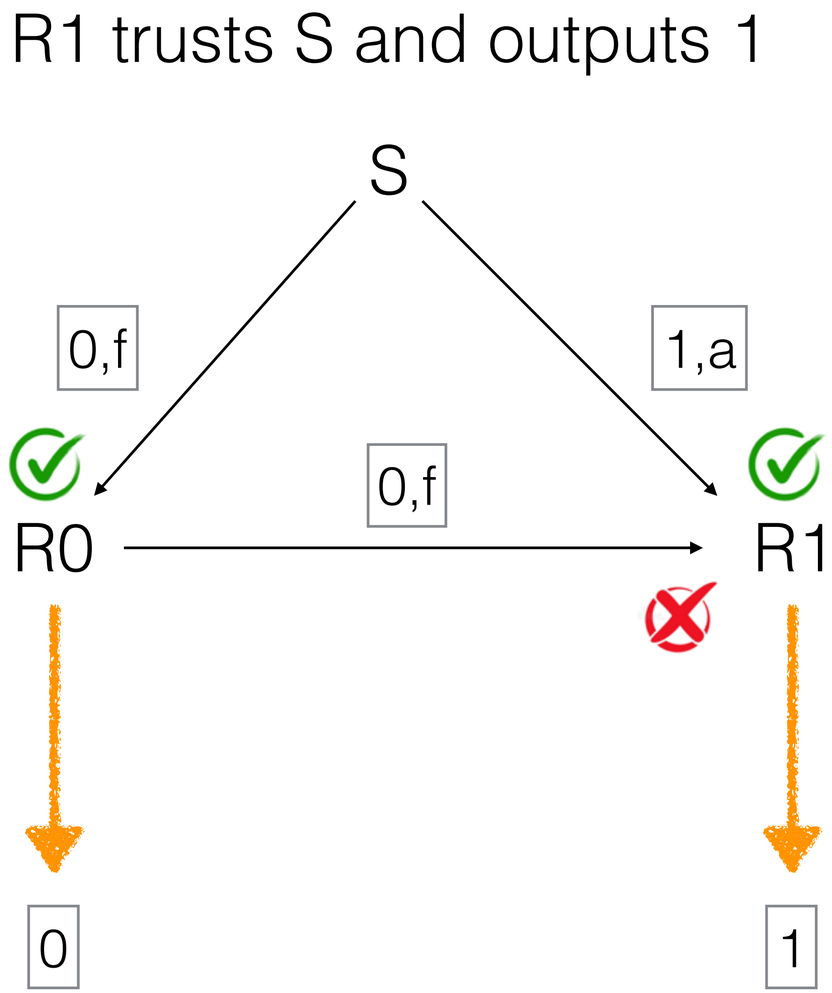
How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

`S faulty' configuration => lucky cheater beats protocol

How does the Fitzi-Cabello-Gaertner Weak Broadcast protocol work?



	S	R0	R1
a	1	1	1
b	2	1	0
c	0	0	0
d	2	1	0
e	1	1	1
f	2	0	1
g	2	0	1
h	1	1	1
i	0	0	0
j	1	1	1
k	0	0	0
m	0	0	0

FAILURE, with probability 1/2
can suppressed by a **minimum string length** parameter

Minimum string length parameter #1: m_0

length of check string should be at least $m_0 N$

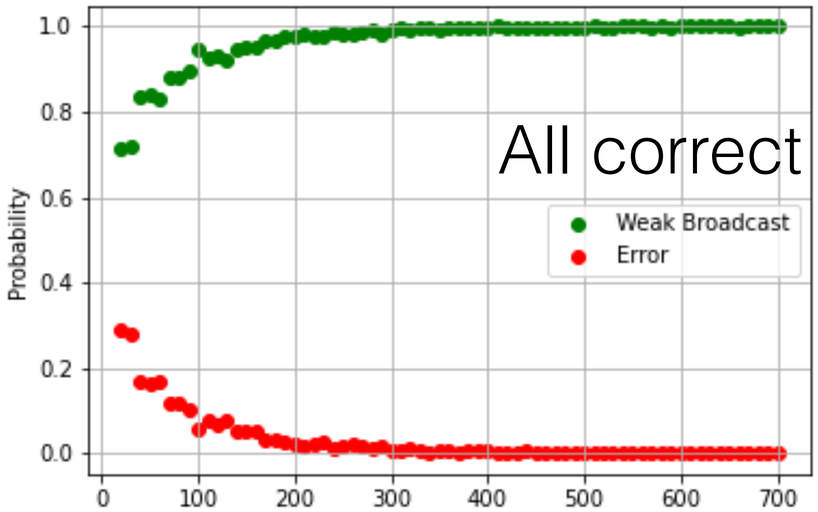
(m_0 should be below but close to 1/3 = 0.33…)

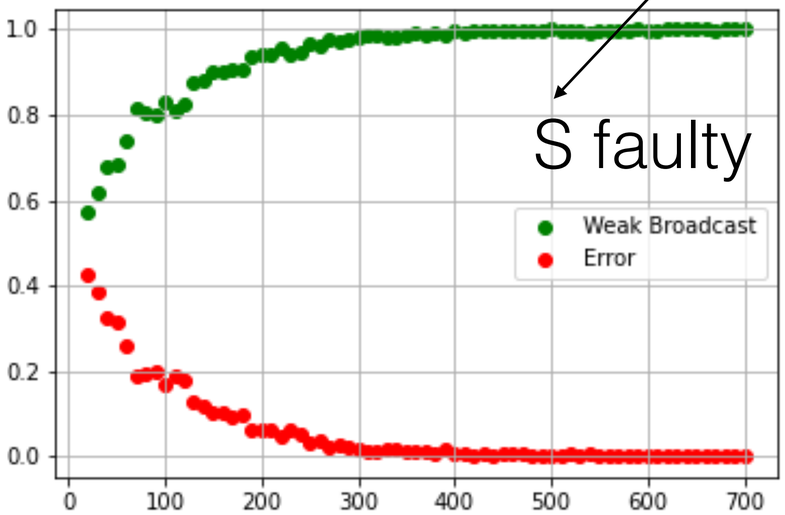
Minimum string length parameter #2: λ

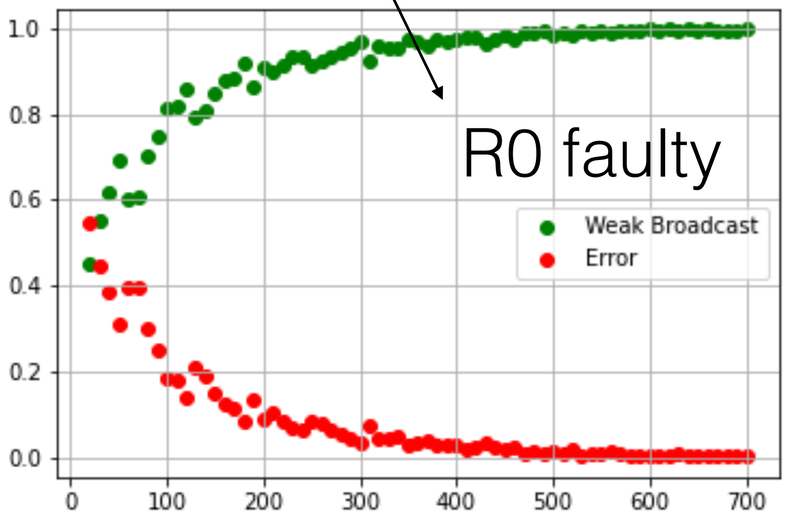
(λ should be below but close to 1)

How many random triplets are needed for 95% success probability?

We try to answer by devising **`smart' adversary strategies**
and Monte-Carlo simulating them







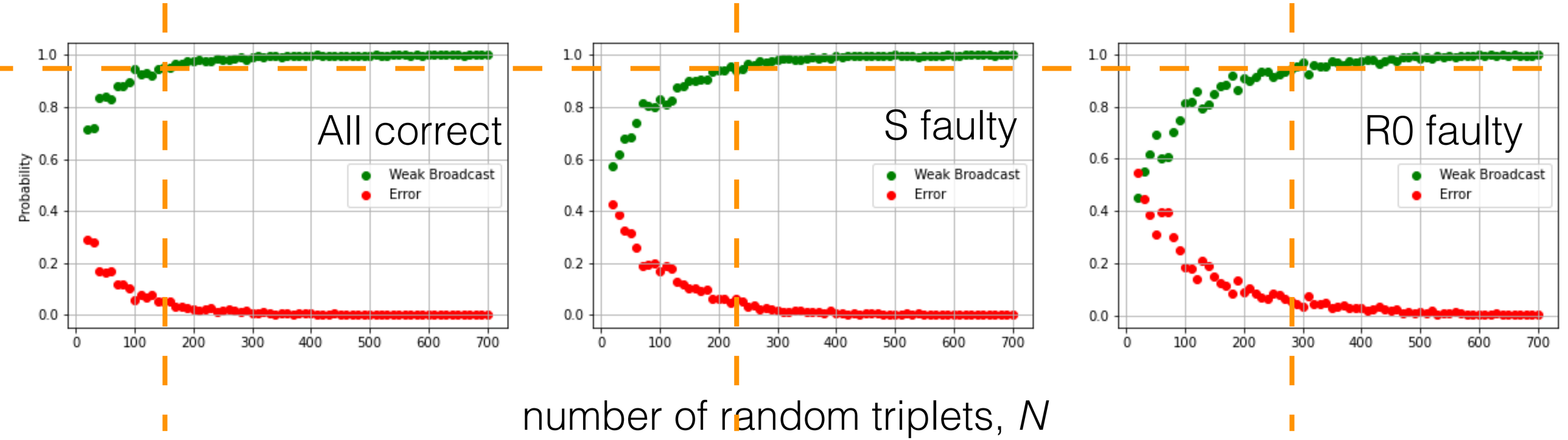
number of random triplets, *N*

$m_0 = 0.27,\ \lambda = 0.94, 1000$ samples

How many random triplets are needed for 95% success probability?

95%

$N_{\min} = 290$

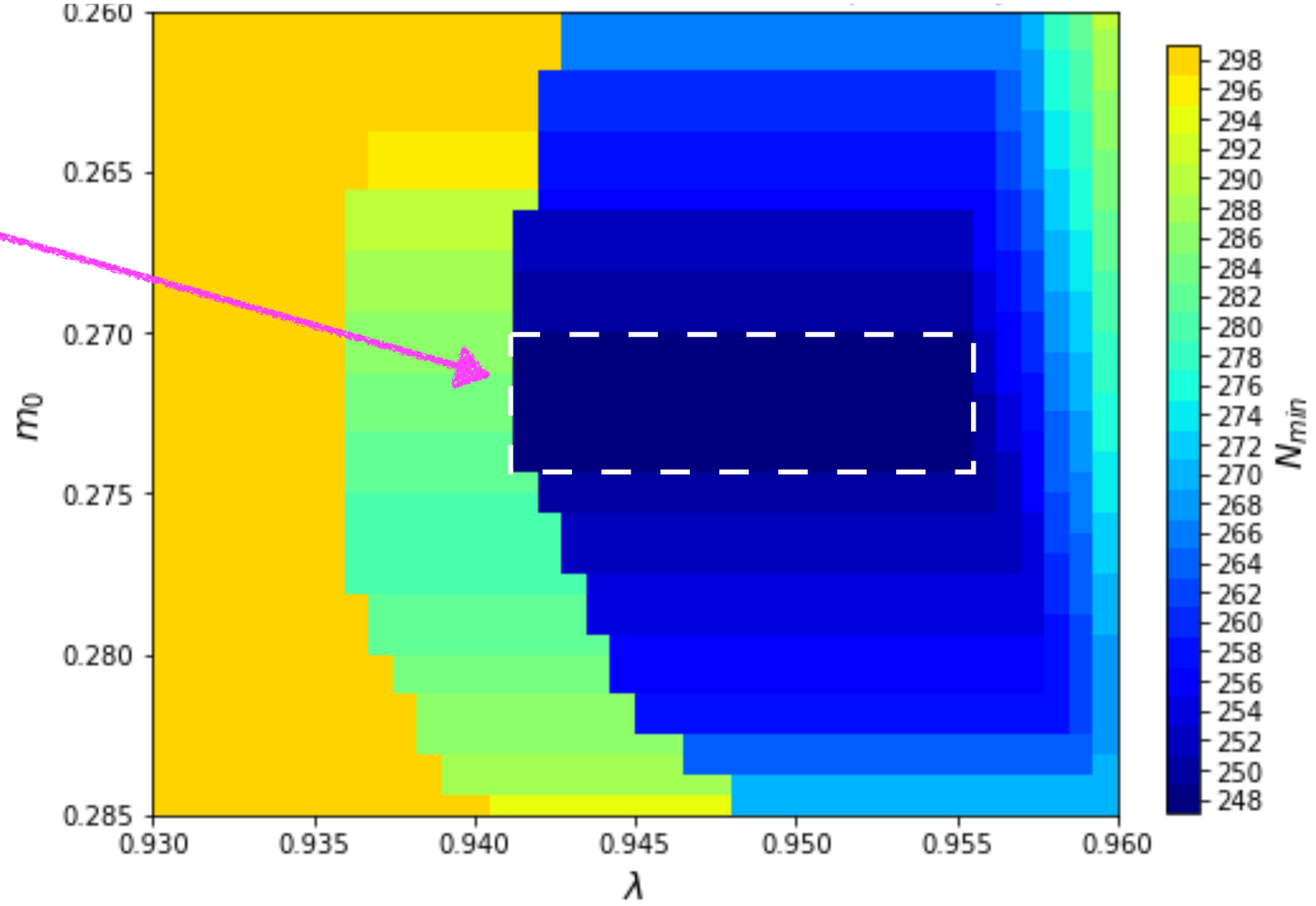


$m_0 = 0.27,\ \lambda = 0.94, 1000$ samples

What are the optimal parameter values?

minimum number of random triplets to achieve Weak Broadcast with 95% probability

sweet rectangle $N_{\min} = 248$



e.g., $\lambda = 0.95$ and $m_0 = 0.272$ is an optimal choice

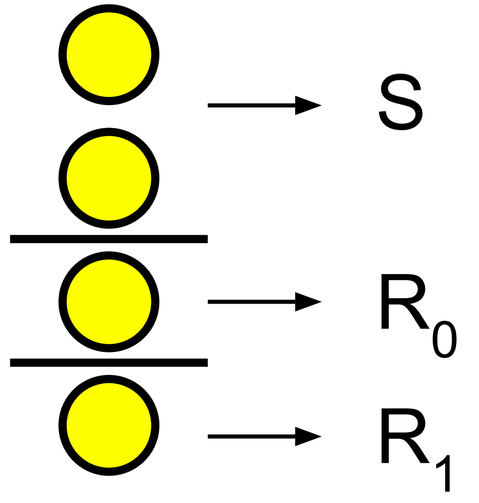
Qubit Approach to the Fitzi-Cabello-Gaertner Weak Broadcast protocol

PHYSICAL REVIEW A **68**, 012304 (2003)

Solving the liar detection problem using the four-qubit singlet state

Adán Cabello*
Departamento de Física Aplicada II, Universidad de Sevilla, 41012 Sevilla, Spain
(Received 18 February 2003; published 7 July 2003)

$$|\mathcal{S}_4^{(2)}\rangle = \frac{1}{2\sqrt{3}}(2|0011\rangle - |0101\rangle - |0110\rangle - |1001\rangle - |1010\rangle + 2|1100\rangle). \quad (2)$$



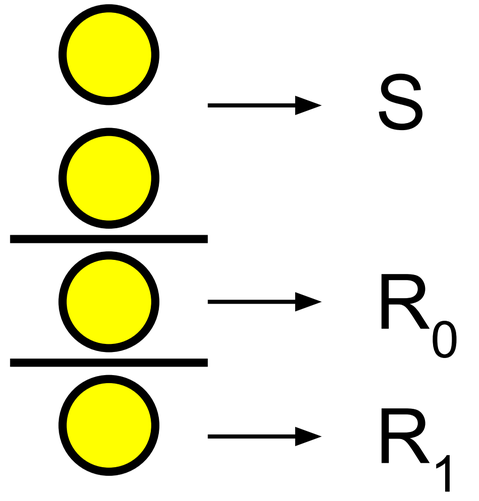
`**four-qubit singlet state**' or the `**Cabello state**'

Cabello state generates the random triplets for Weak Broadcast

probability	S	R0	R1
1/3	1,1	0	0
1/3	0,0	1	1
1/12	1,0	1	0
1/12	0,1	1	0
1/12	1,0	0	1
1/12	0,1	0	1

probability	S	R0	R1
1/3	0	0	0
1/3	1	1	1
1/6	2	1	0
1/6	2	0	1

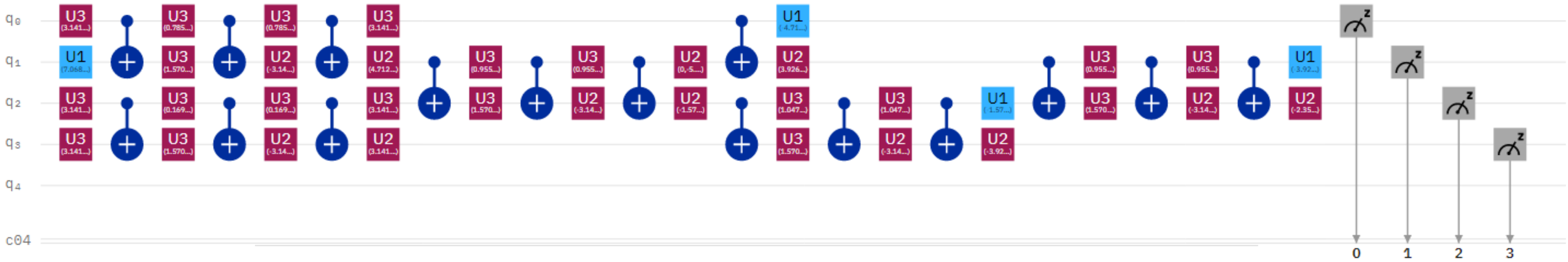
$$|\mathcal{S}_4^{(2)}\rangle = \frac{1}{2\sqrt{3}}(2|0011\rangle - |0101\rangle - |0110\rangle - |1001\rangle - |1010\rangle + 2|1100\rangle). \quad (2)$$



`**four-qubit singlet state**' or the `**Cabello state**'

How to create the Cabello states on a quantum computer?

Gard et al., npj QI 2020



Calibration data

Last calibrated: 43 minutes ago

Map view | Graph view | Table view

Qubit:

Frequency (GHz)

Avg 4.911

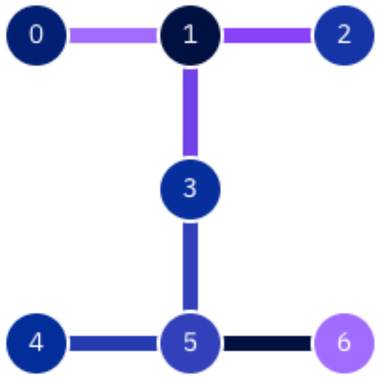
min 4.76 max 5.177

Connection:

CNOT error

Avg 9.741e-3

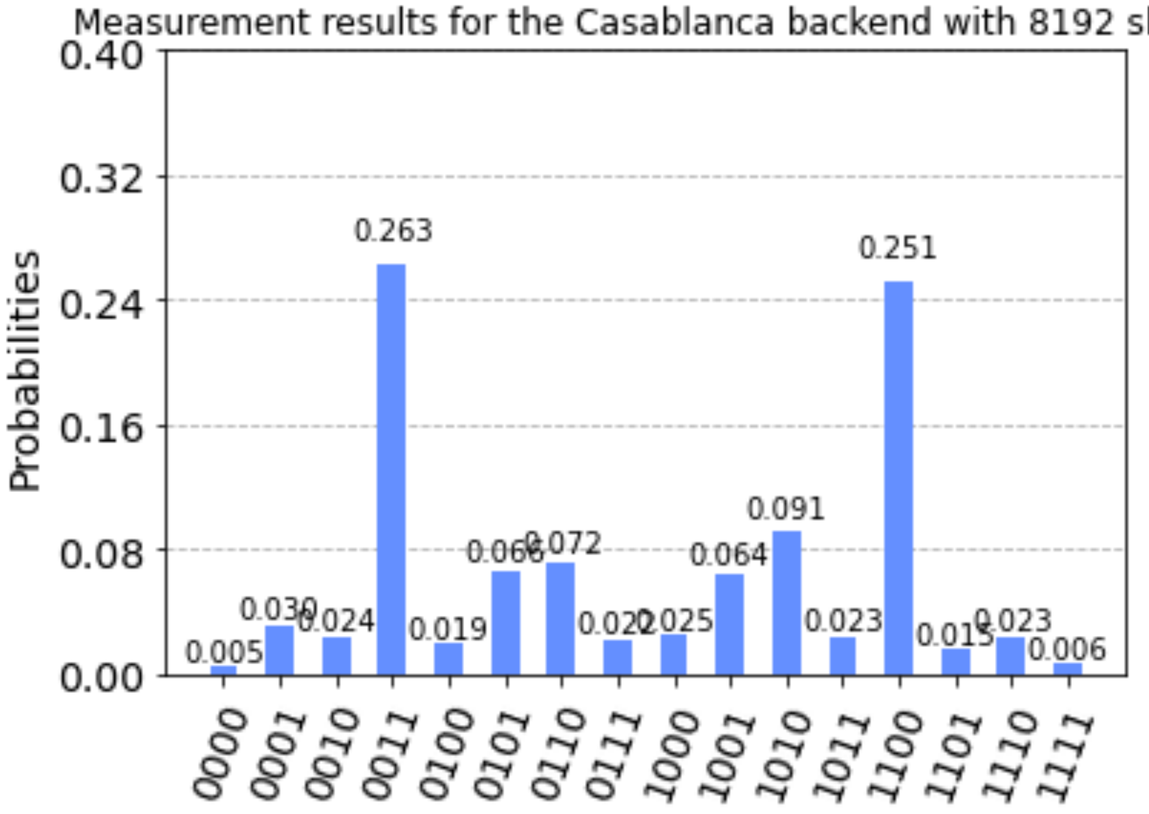
min 6.070e-3 max 1.269e-2



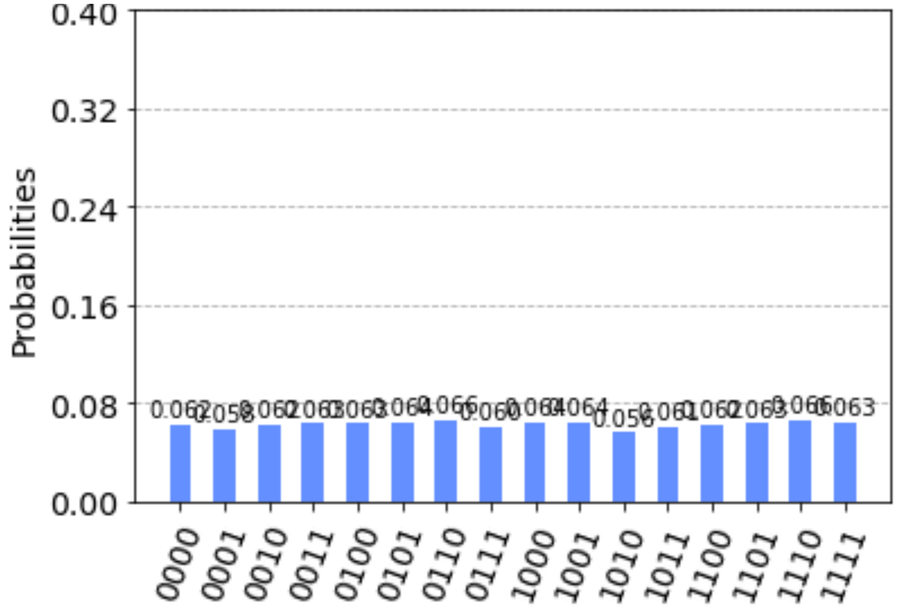
quantum computers as quantum network simulators: Pathumsoot et al PRL 2020

How precisely can we prepare the Cabello states on IBM?

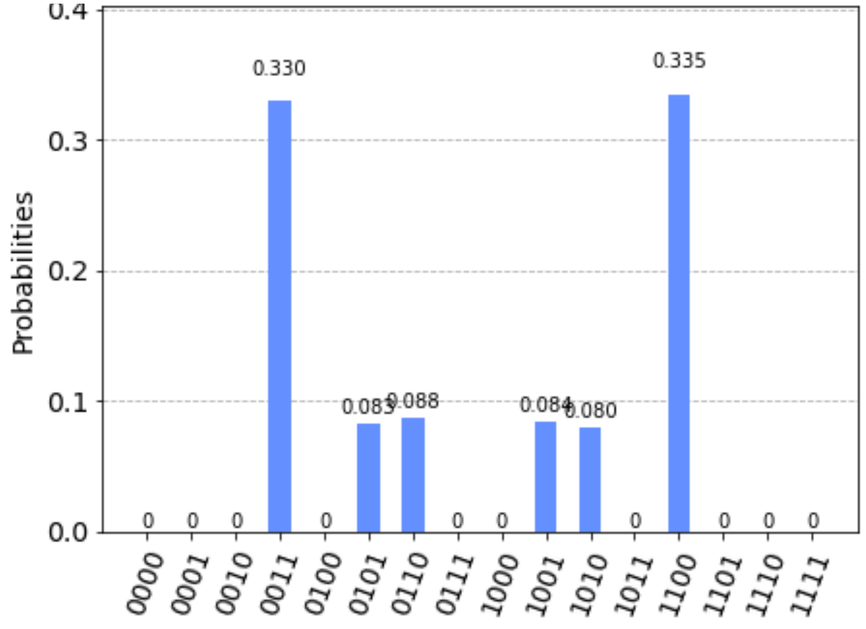
Classical Fidelity: $\mathrm{CF}(p,q) = \sum_{j=1}^{16} \sqrt{p_j q_j}$



CF ~ 0.57

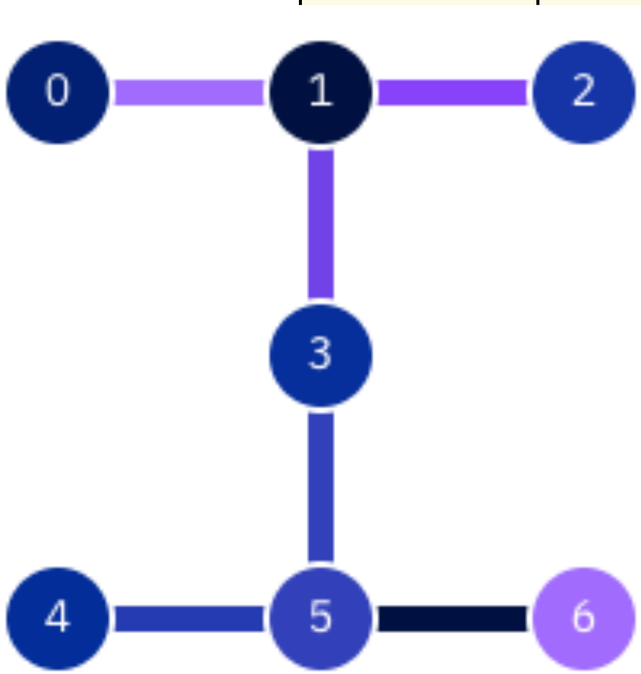


CF ~ 0.9



How precisely can we prepare the Cabello states on IBM?

$$\mathrm{CF}(p,q) = \sum_{j=1}^{16} \sqrt{p_j q_j}$$

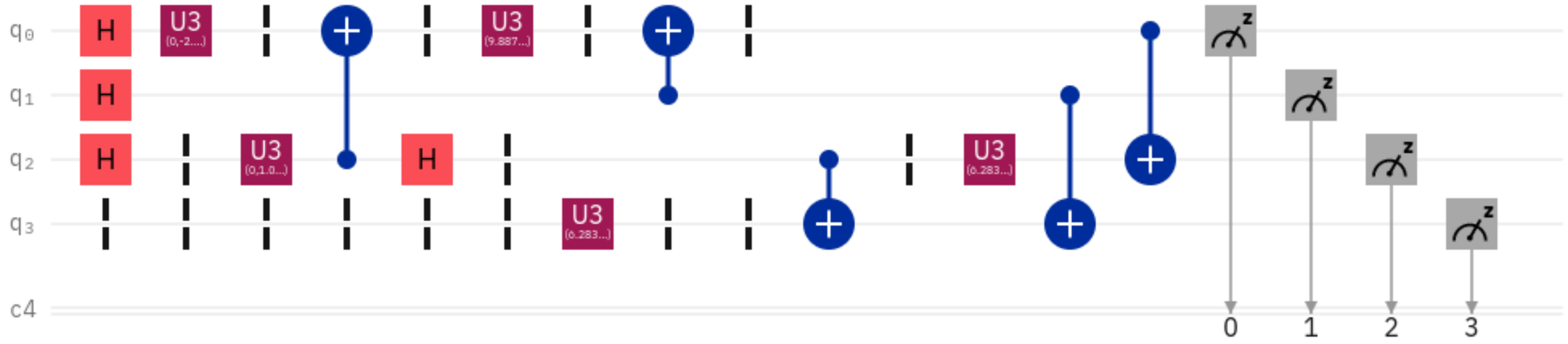


Backend	#qubits	used qubits	shots	CF	CF mitigated	data (.txt)	date
Casablanca	7	0-1-3-5	8192	0.8752	0.9035	casablanca1	2021.02.17.
		5-3-1-0	8192	0.8823	0.9119	casablanca5	2021.02.20
		2-1-3-5	8192	0.8972	0.9227	casablanca2	2021.02.18.
		5-3-1-2	8192	0.8605	0.8811	casablanca6	2021.02.20
		1-3-5-4	8192	0.8559	0.8884	casablanca3	2021.02.18.
		4-5-3-1	8192	0.8642	0.912	casablanca7	2021.02.20
		1-3-5-6	8192	0.8672	0.8977	casablanca4	2021.02.18.
		6-5-3-1	8192	0.8910	0.9243	casablanca8	2021.02.20

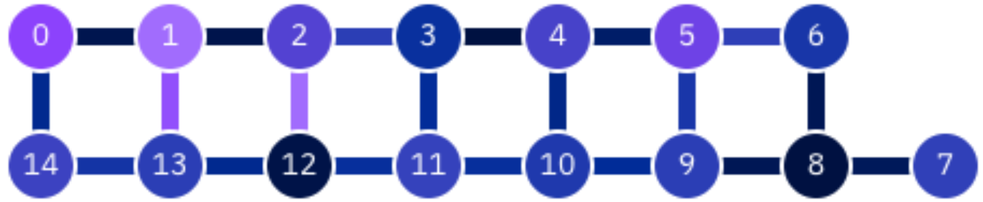
Note: random garbage gives CF (classical fidelity) ~ 0.57

Searching for a state-preparation circuit with less CNOTs

(1) Sample a discrete set of random circuits.
(2) Pick the circuits providing bitstring distribute similar to Cabello's.
(3) Further optimize (gradient descent) of the circuits to reach unit fidelity with Cabello



ibmq_16_melbourne

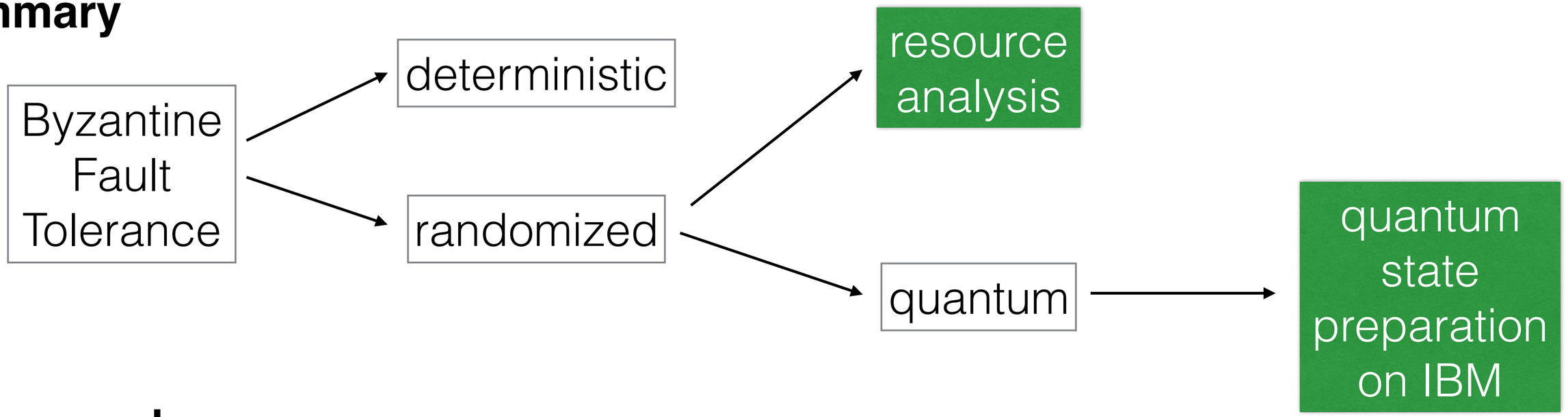


The params for the U3 gates

[0, -2.2689280275926214, 1.53588974175501,
0, 1.0733774899765132, 1.6057029118347823,
9.887290212547788, 6.283185307179586, 0,
6.283185307179586, 6.283185307179586, 0,
6.283185307179586, 6.283185307179586, 1.5707963267948966]

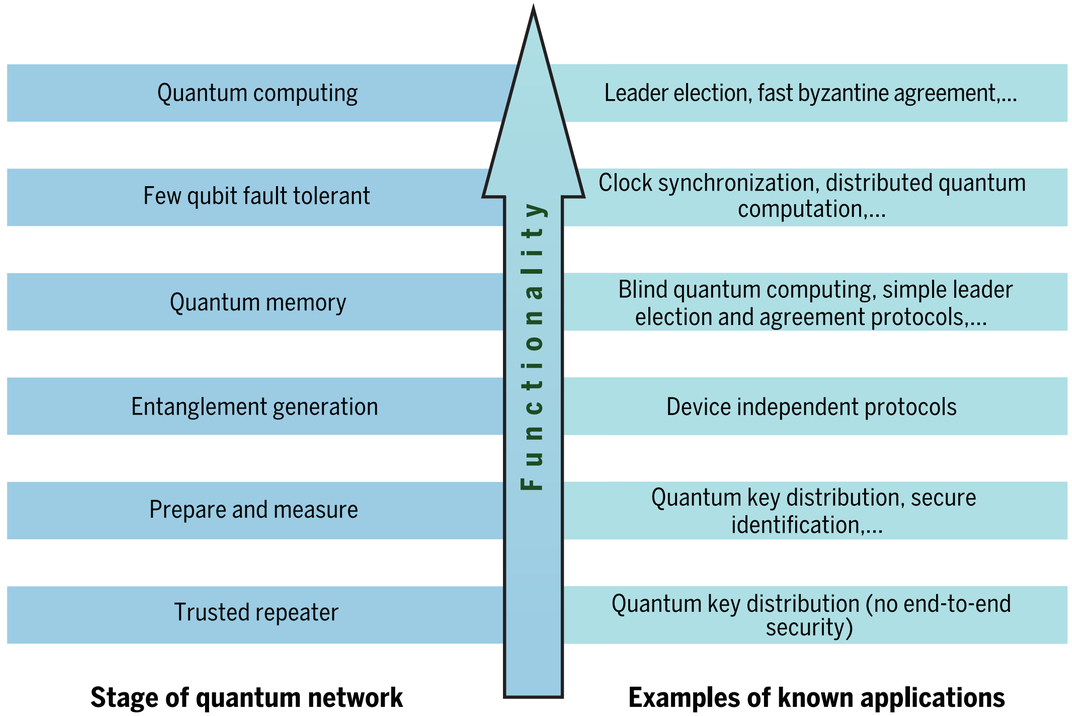
similar fidelities as before

Summary



Future work

- Model physical imperfections of a quantum network (e.g., with NetSquid)
- Make protocols robust against physical imperfections
- Resource analysis of the extension Weak Broadcast(3,1) => Broadcast
- Identify technological opportunities at earlier stages of quantum network development
- …



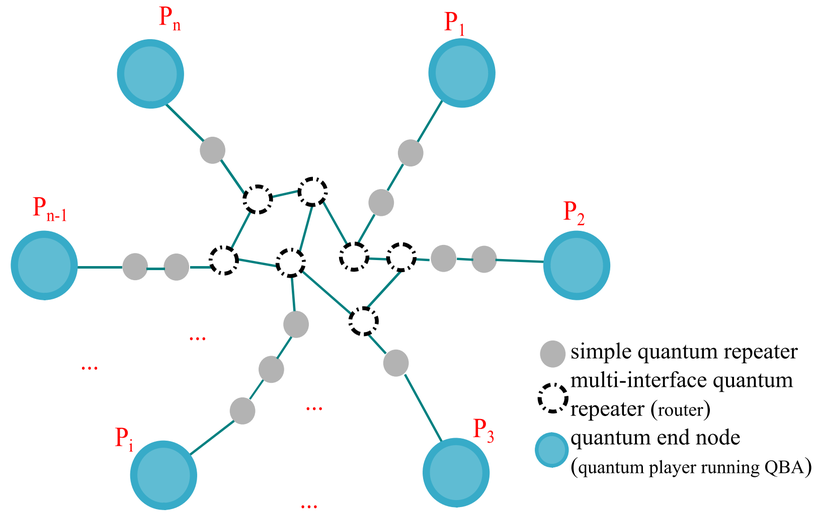


Figure 1. Required elements in quantum repeater networks (QRN) for running scalable quantum distributed applications. Each quantum end-node (the big circles around the network-green in online) may distribute the required state by teleportation of the state to the node via a direct point to point communication link. In general, scalable, long distance distribution of Bell pairs between end-nodes requires quantum repeaters with two-interfaces (small gray circles), or multi-interface quantum routers (dotted circles).

Distribute & Test

or

Verification

phase

- Distribute Cabello states.
- Pick random Cabello states as Test States.
- Measure Test States in random basis.
- Exchange measurement data.
- Verify correlations to prove succesful state distribution.

Advantage of quantum

- Verifiable correlated-random-triplet generator
- Secure against eavesdropping on the correlated random triplet