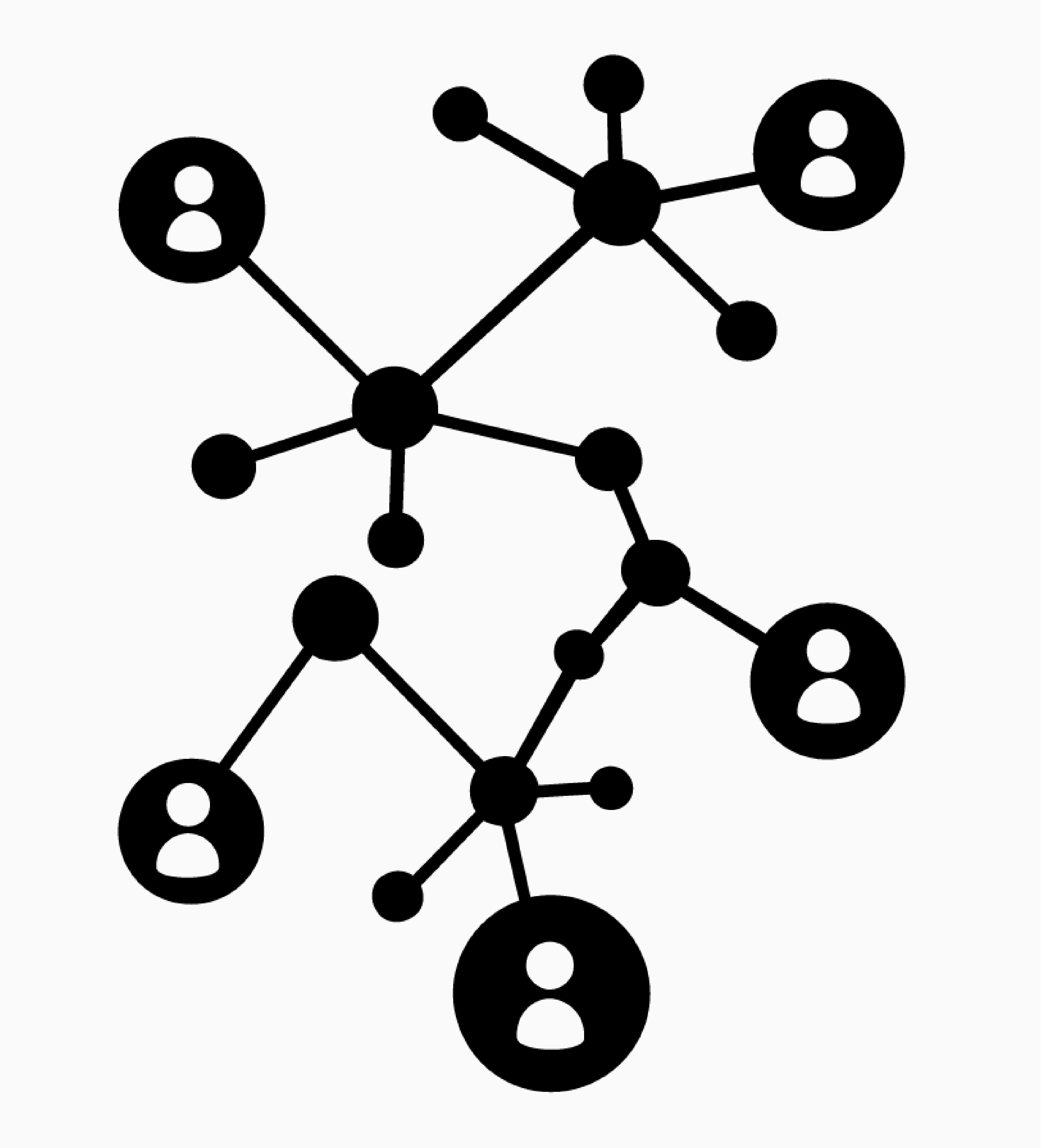
Graphene: Efficient Interactive Set Reconciliation Applied to Blockchain Propagation

19210240055

袁和昕

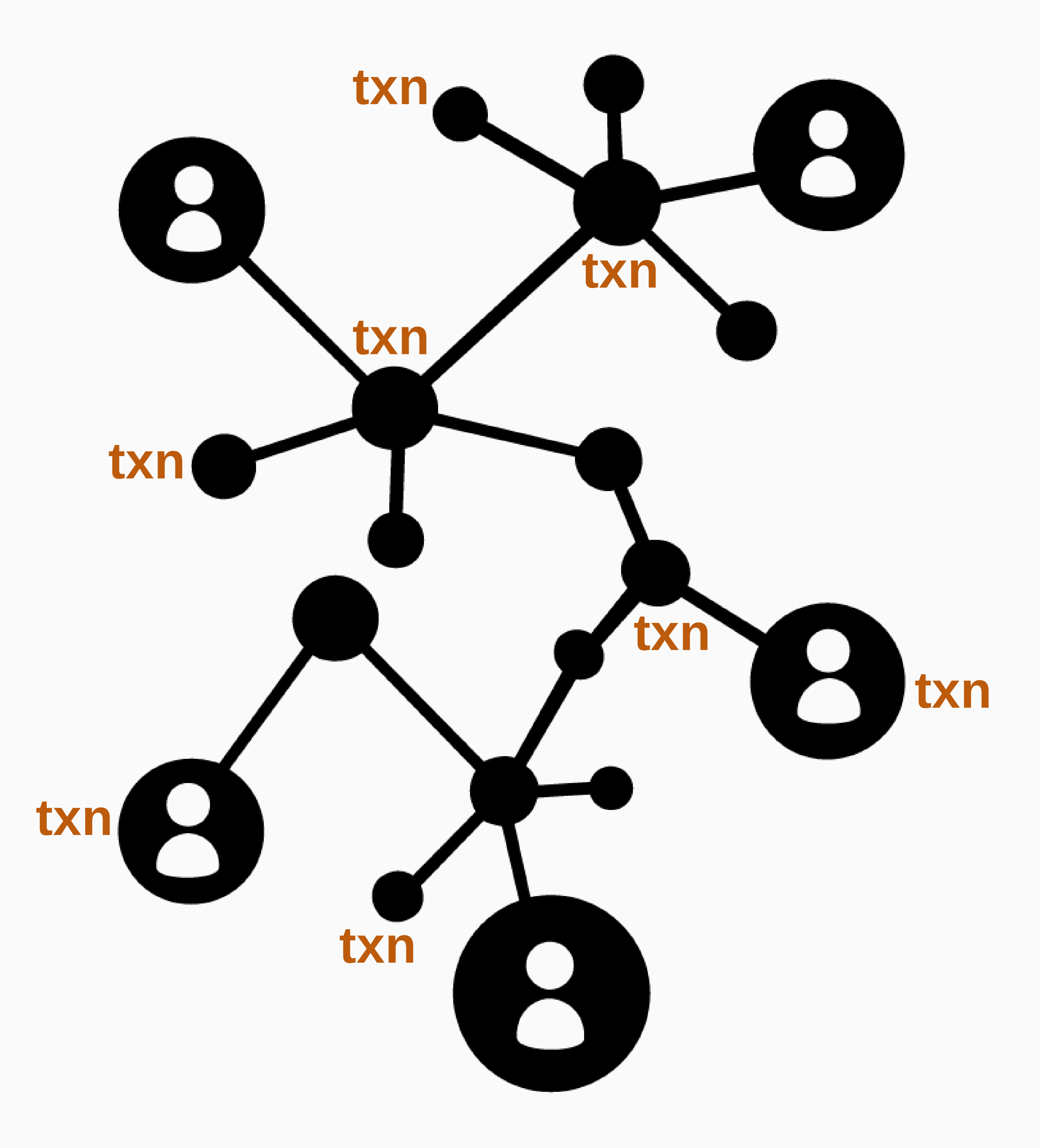
Background

- P2p distributed systems
- End-points need to talk to each other

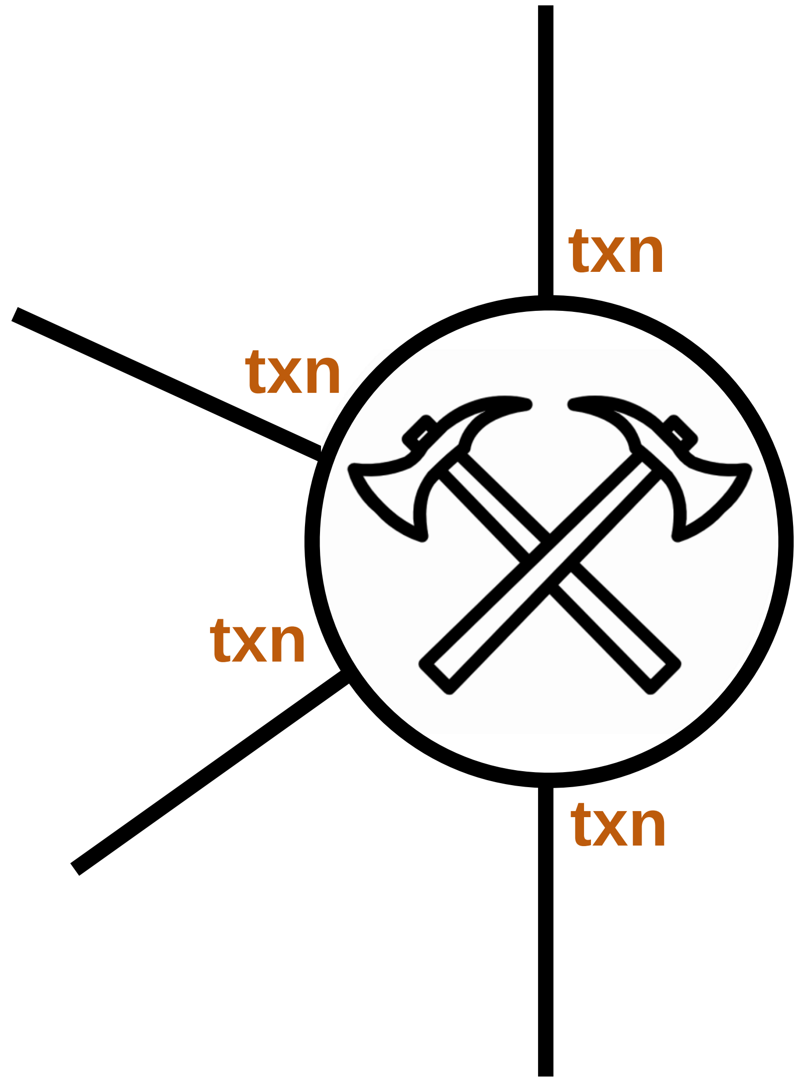


Background: Txns

- Transactions (txns) are transfers of money
- Unvalidated txns are broadcast

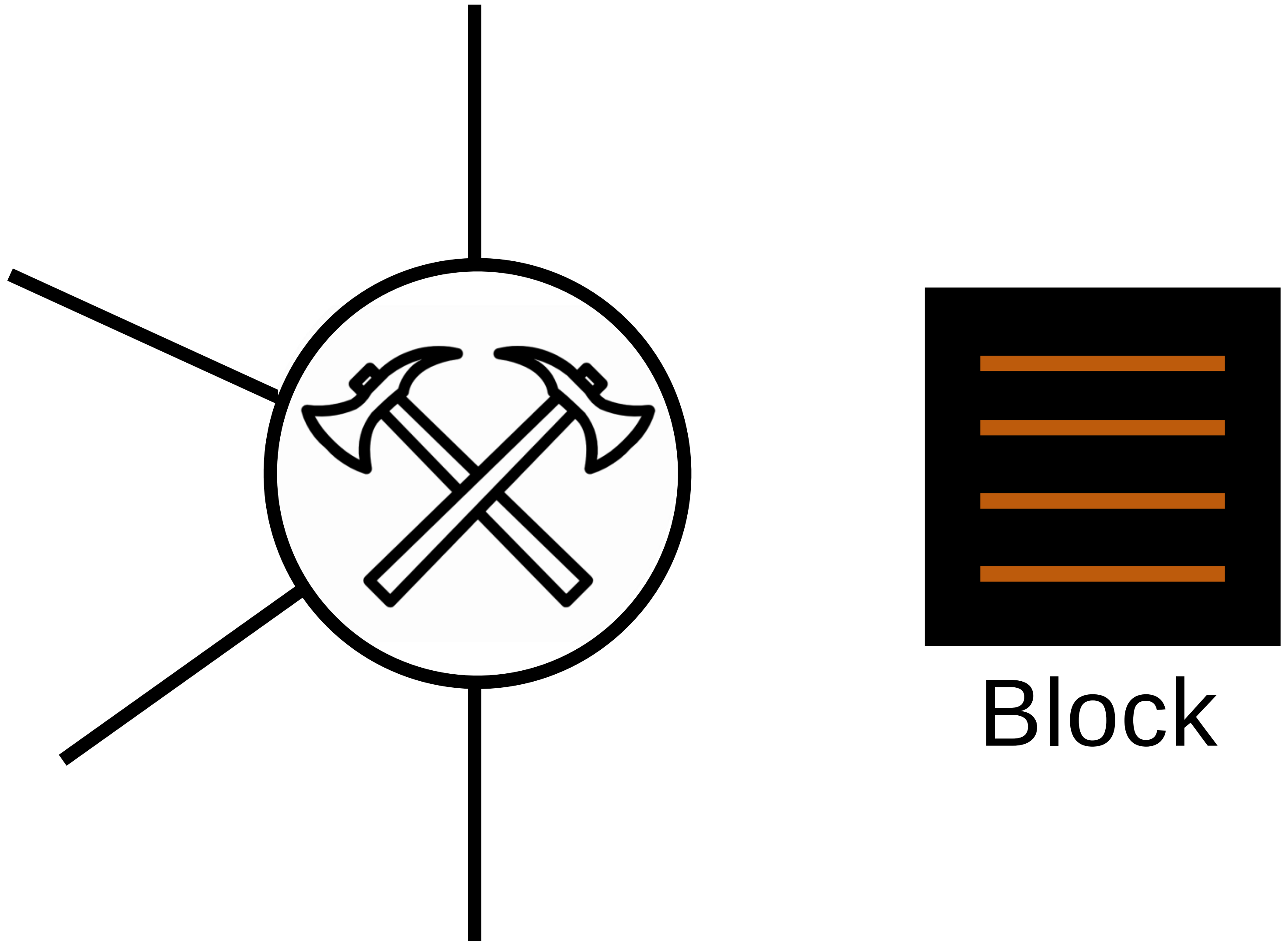


Background: Blocks



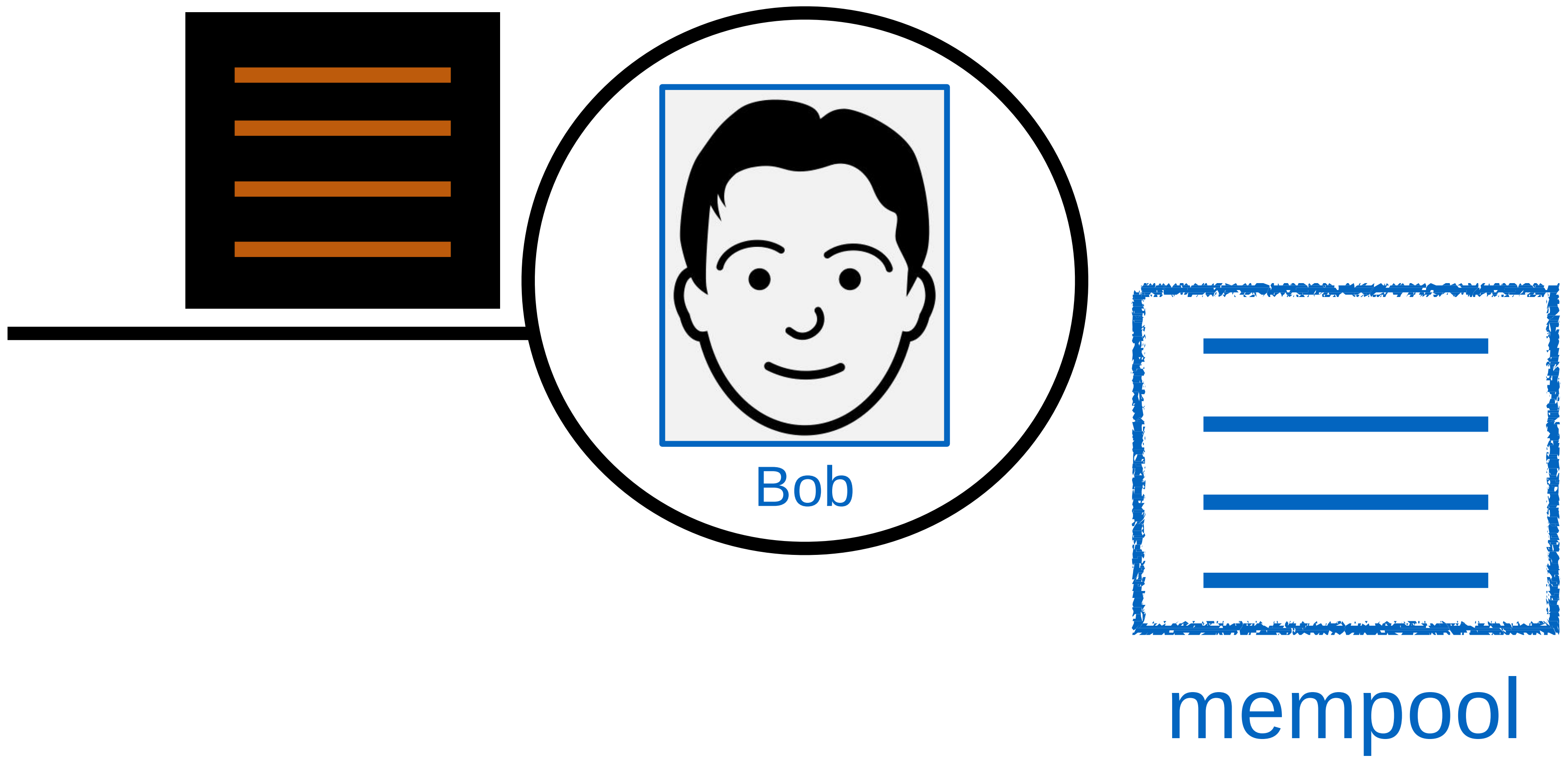
- Transactions (txns) are transfers of money
- Unvalidated txns are broadcast

Background: Blocks



- Transactions (txns) are transfers of money
- Unvalidated txns are broadcast
- Blocks are comprised of txns

Background: Mempool

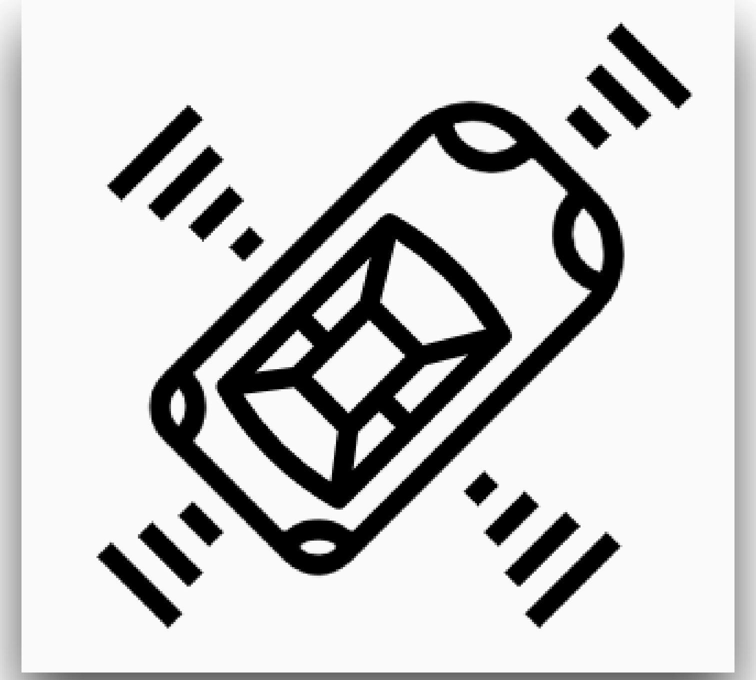


- Blocks are comprised of txns
- Each peer in the network has a pool of unvalidated txns, called the **mempool**
- Peers clear out txns from their mempool

Motivation

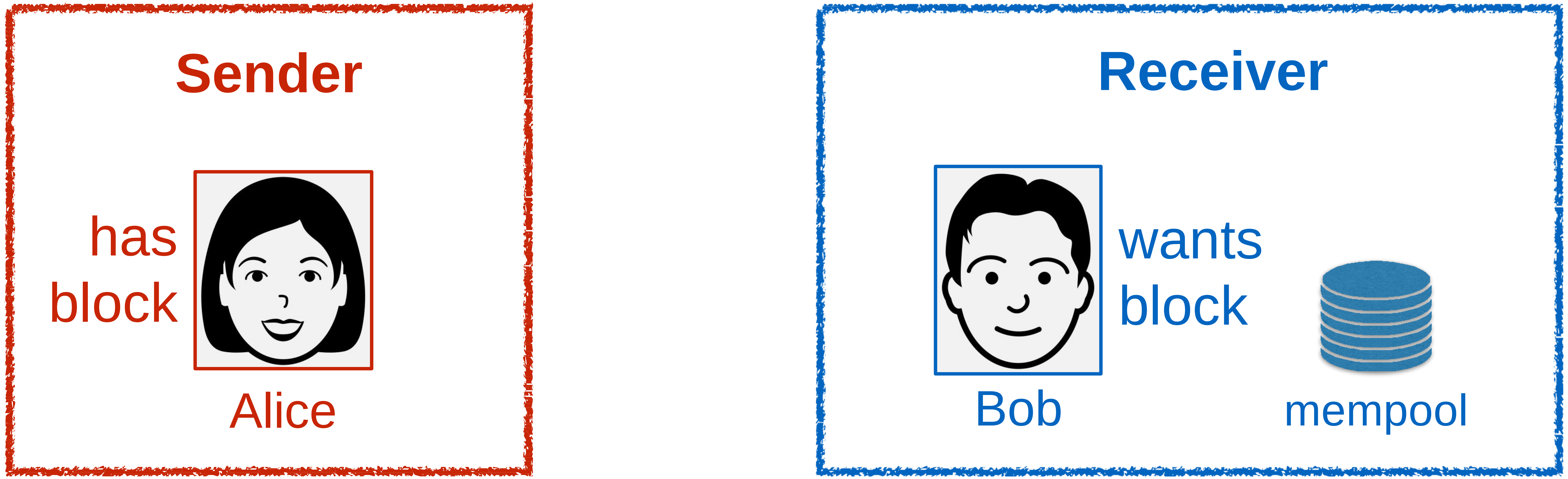
- Many distributed systems require synchronization of records among processes

- Blockchains are just the latest example
  - Replicas in distributed databases
  - Distributed sensors
  - Security certifications



- Must solve **set reconciliation**

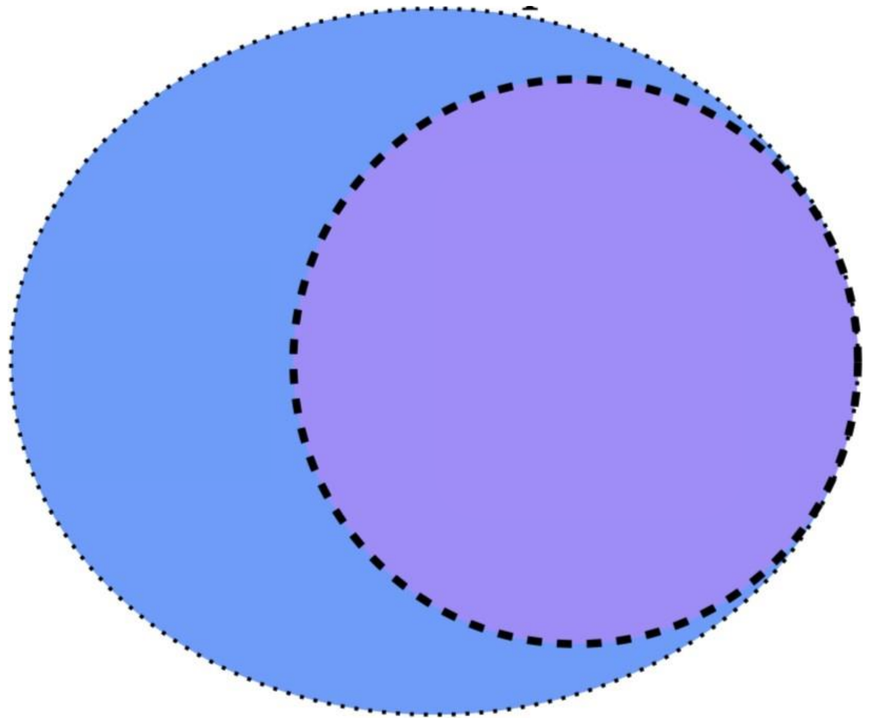
Setup

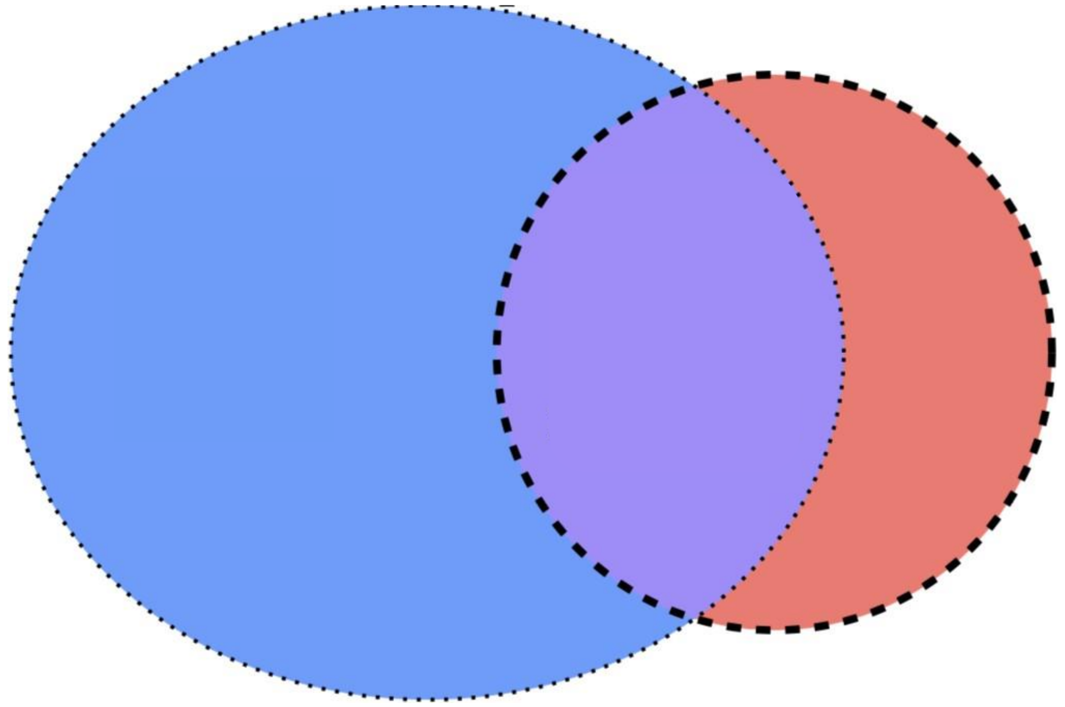


- **Goal:** Send as little data as possible over the wire

Problem Definition

- Given a block of txns from Alice, and a set of txns at Bob, determine:





- The subset of Bob's txns that are in the block
- The subset of txns that Bob is missing

Contributions

- **A new protocol** that solves which elements in a set M stored by a receiver are members of a subset $N \subseteq M$
- **Extension of our protocol** where some of the elements of N are missing
- **Efficient search algorithm for parameterizing an IBLT**
- Evaluation using **open-source deployment** in the real-world, mathematical analysis, and **simulation**

Bloom Filters

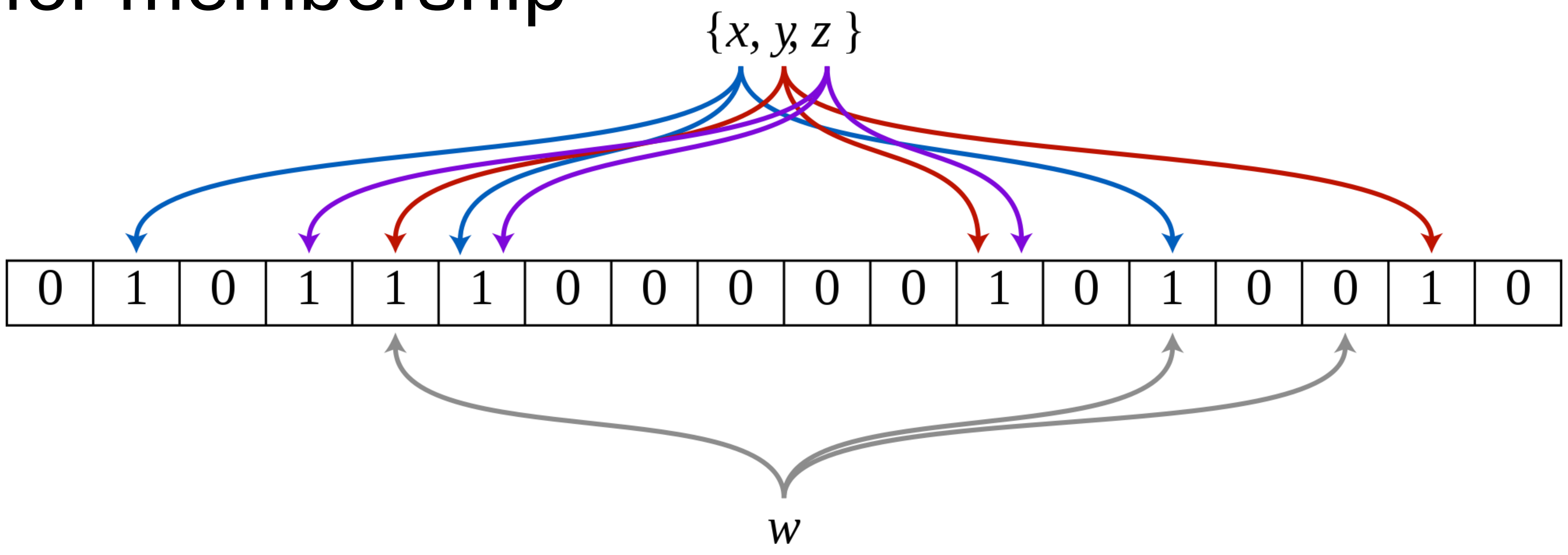
- Bloom filters represent a set of n items
- binary array T (initial value is 0), $\frac{-n log_2 f}{ln^2 2}$ bits
- k hash functions h_1, ..., h_k
  - $\forall x \in S, 1 \leq i \leq k, T[h_i(x)] = 1, \forall x \in S, 1 \leq i \leq k$ 1
  - $T[h_{i\,(x)}] == 1, 1 \leq i \leq k$ 2

Bloom Filters

- The False Positive Rate (FPR) is tunable
  - More bits will lower the FPR

- Number of FPs we will observe approximately follow a binomial distribution with two parameters:
  - *n:* number of items to test for membership
  - *p*: probability of failure

If **FPR =** $\frac{1}{m-n}$,then we expect 1 transaction from mempool to falsely appear to be in the m – n txs



Invertible Bloom Lookup Tables

- IBLTs are a generalization of Bloom Filters
  - Instead of a bit, cells include a count and actual content
- IBLTs I
  - j items, $c = j\tau$ rows, $k + 1$ hash functions, k sub-tables of size c/k

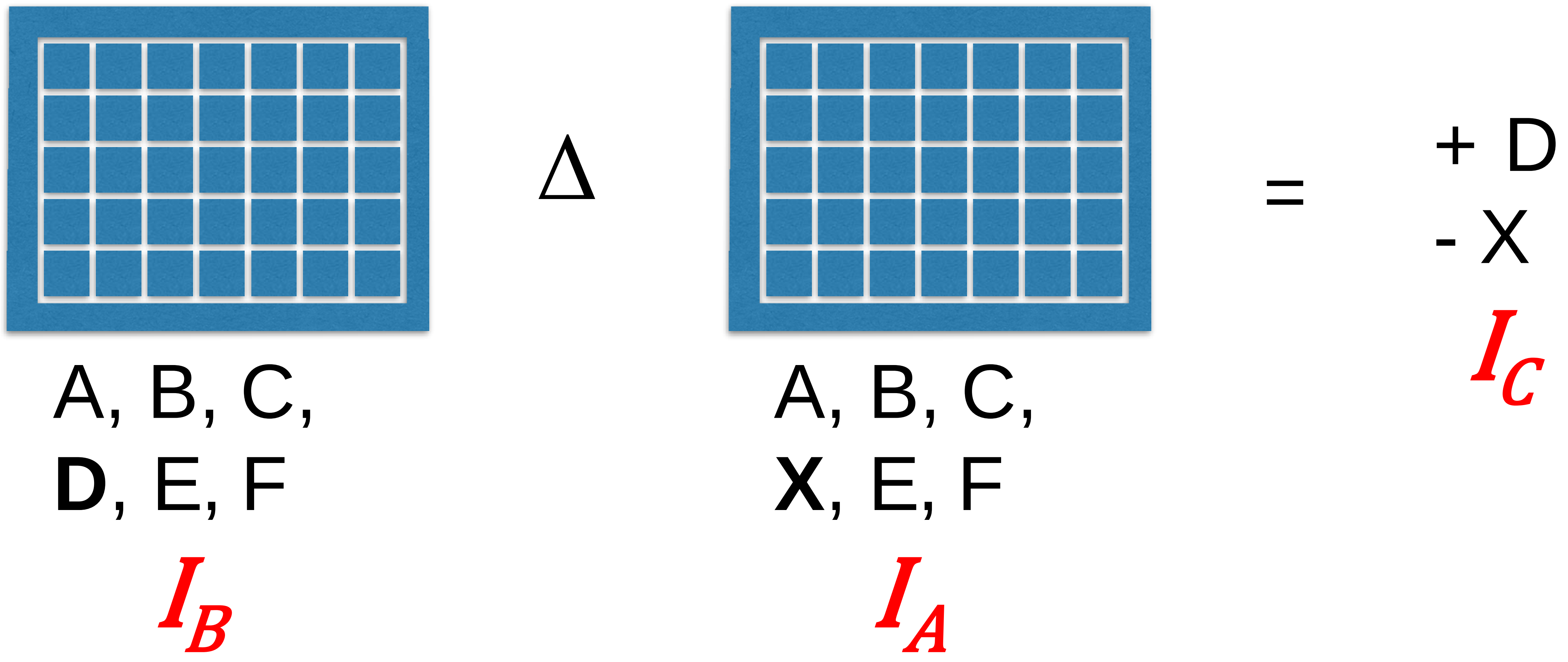
row	count	keySum	value	checkSum
0	1	0x6170706c65	0b0011	100f7a
1	1	0x6170706c65	0b0011	100f7a
…	…	…	…	…

Invertible Bloom Lookup Tables

- We can separate the key-value pairs through the table rows where the **count** is 1 and recalculate the insertion position to delete the key-value pairs from the table.
  By gradually deleting the table row where the **count** is 1, the original set S can be recovered from the IBLT table I.

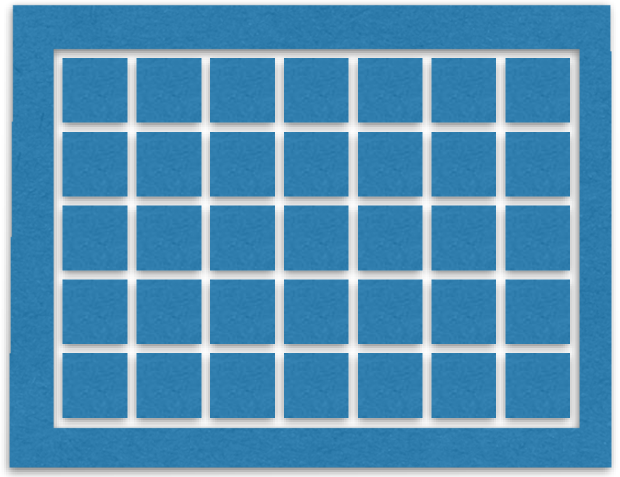
row	count	keySum	Value	checkSum
0	1	0x6170706c65	0b0011	100f7a
1	1	0x6170706c65	0b0011	100f7a
…	…	…	…	…

Invertible Bloom Lookup Tables



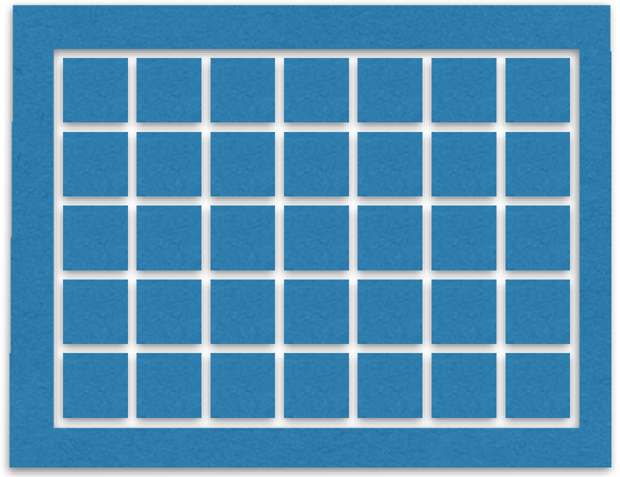
- IBLTs support **subtraction**
  - IBLTs must be **the same size** for subtraction
  - Subtraction recovers **symmetric difference**, $(S_A\text{-}S_B) \cup (S_B\text{-}S_A)$
- If subtraction recovers the entire symmetric difference, then we say that the **subtraction decoded**

Invertible Bloom Lookup Tables



A, B, C,
D, E, F
I_B

Δ



A, B, C,
X, E, F
I_A

=

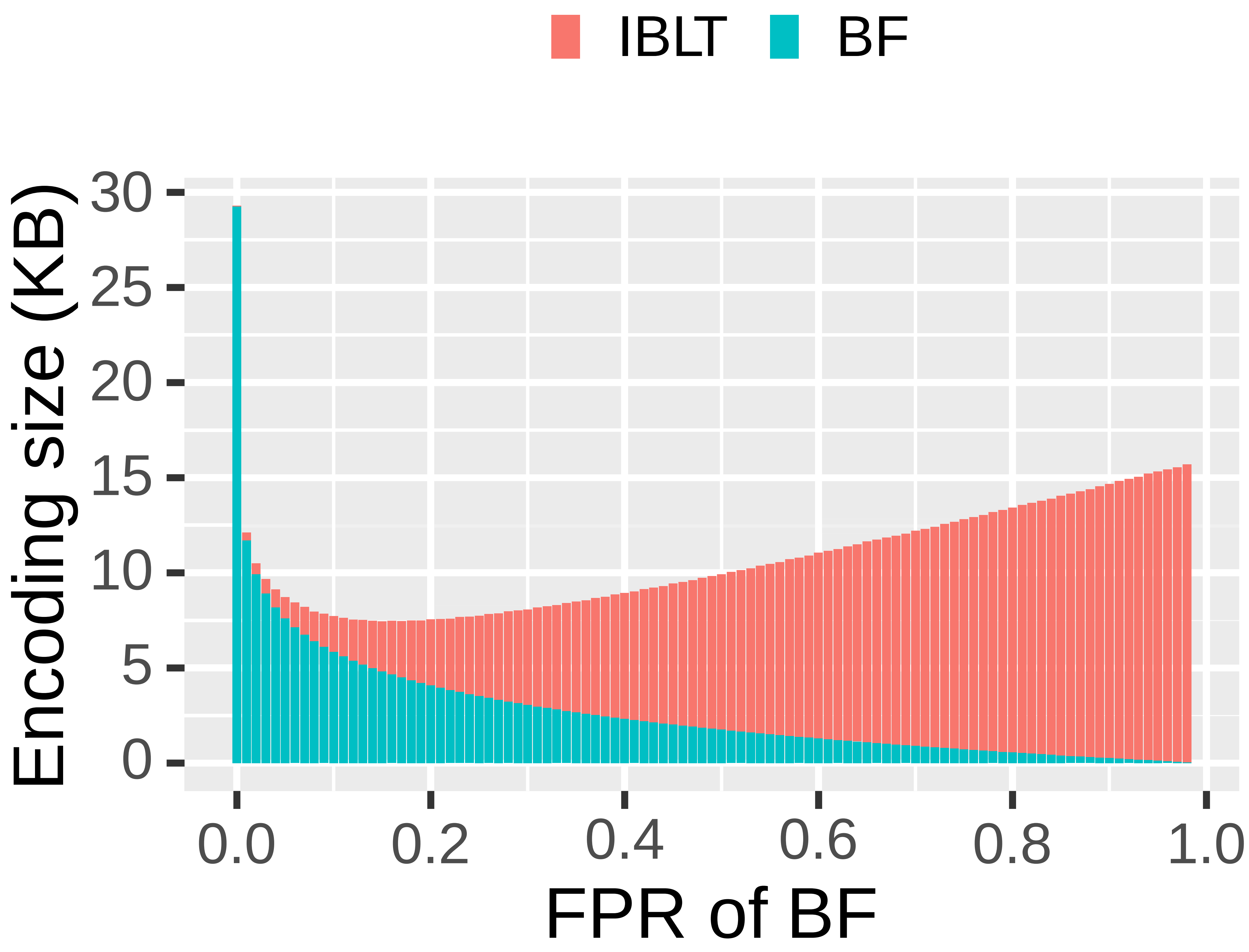
\+ D
\- X
I_C

row	count	keySum	Value	checkSum
0	1	0x6170706c65	0b0011	100f7a
1	-1	0x4300754343	044326	100e1b
…	…	…	…	…

Graphene

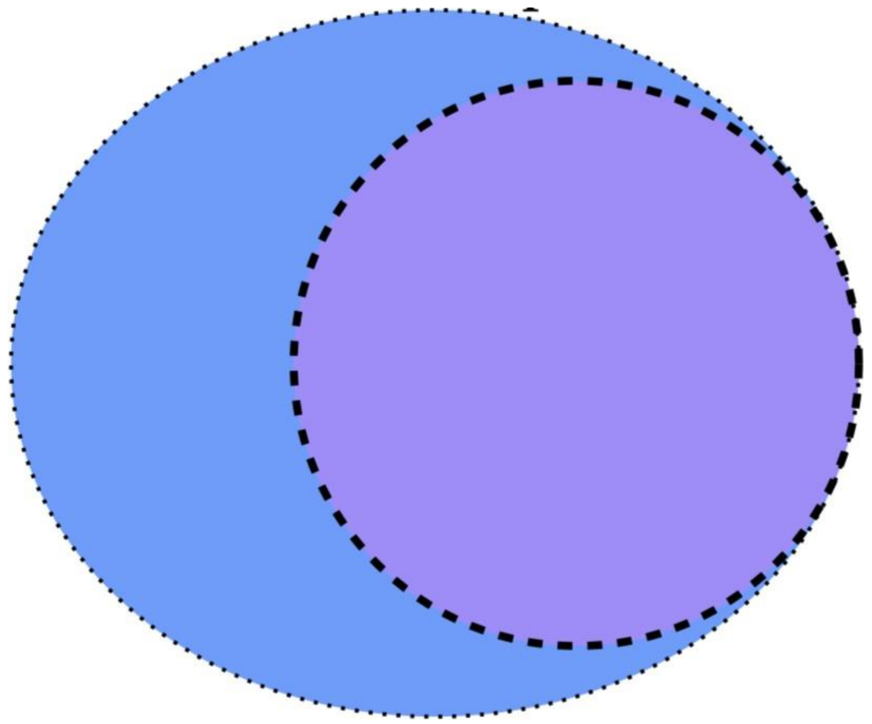
- There is something to optimize!

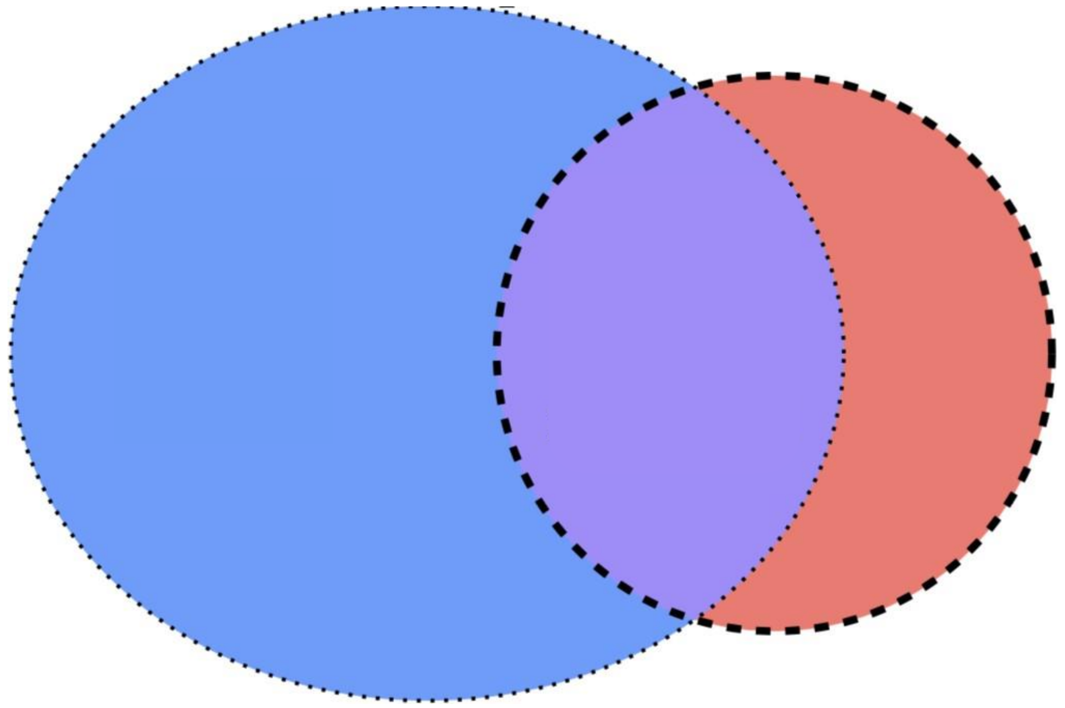
- The figure shows that we don’t need a low FPR for Bloom filter
  - The IBLT can help us recover from the mistakes made by the Boom filter



Graphene

- A occurs in X with probability at least β.
- Given a block of txns from Alice（blue）

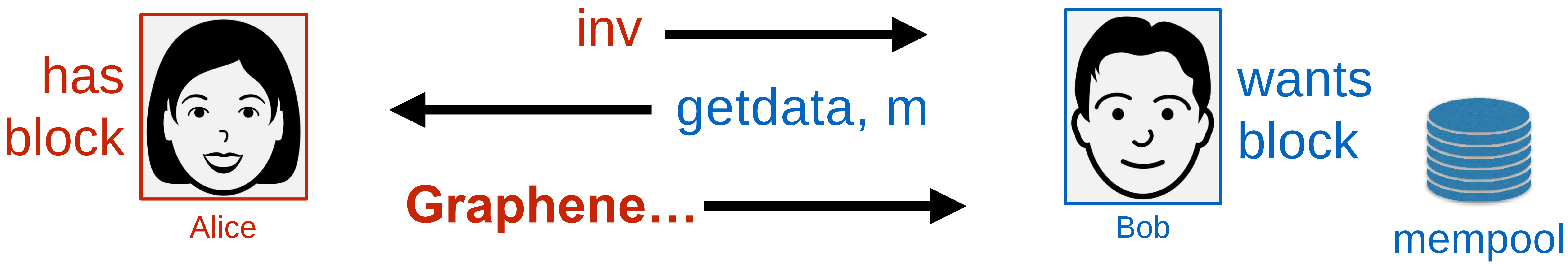




- Protocol 1: The subset of Bob's txns that are in the block（purple）
- Protocol 2: The subset of txns that Bob is missing（red）

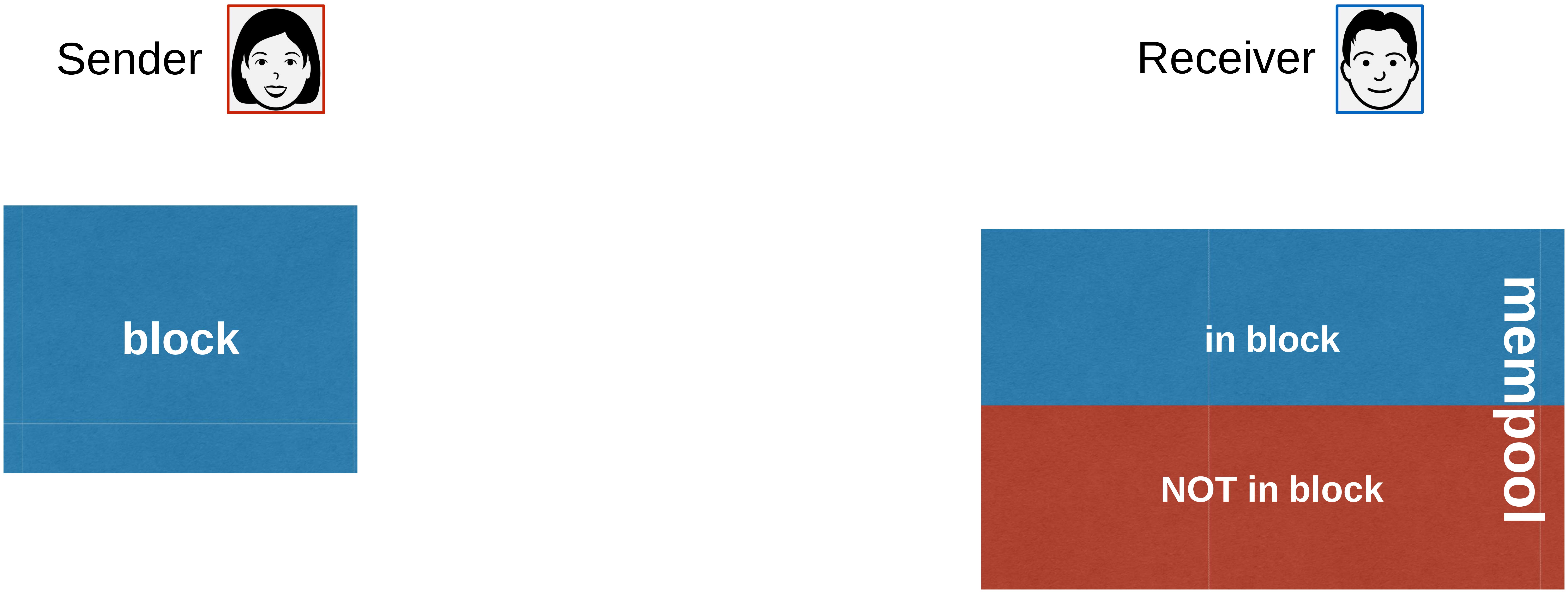
Compact Blocks

- inv： block headers
- getdata： requests the block if needed
- Alice’s block： n txs
  Bob’s mempool： m txs



Graphene: Protocol 1

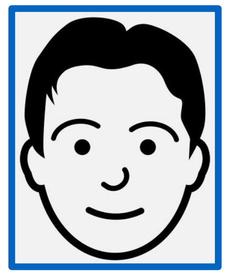
- The block is a subset of the mempool



Graphene: Protocol 1

- The block is a subset of the mempool

Sender 

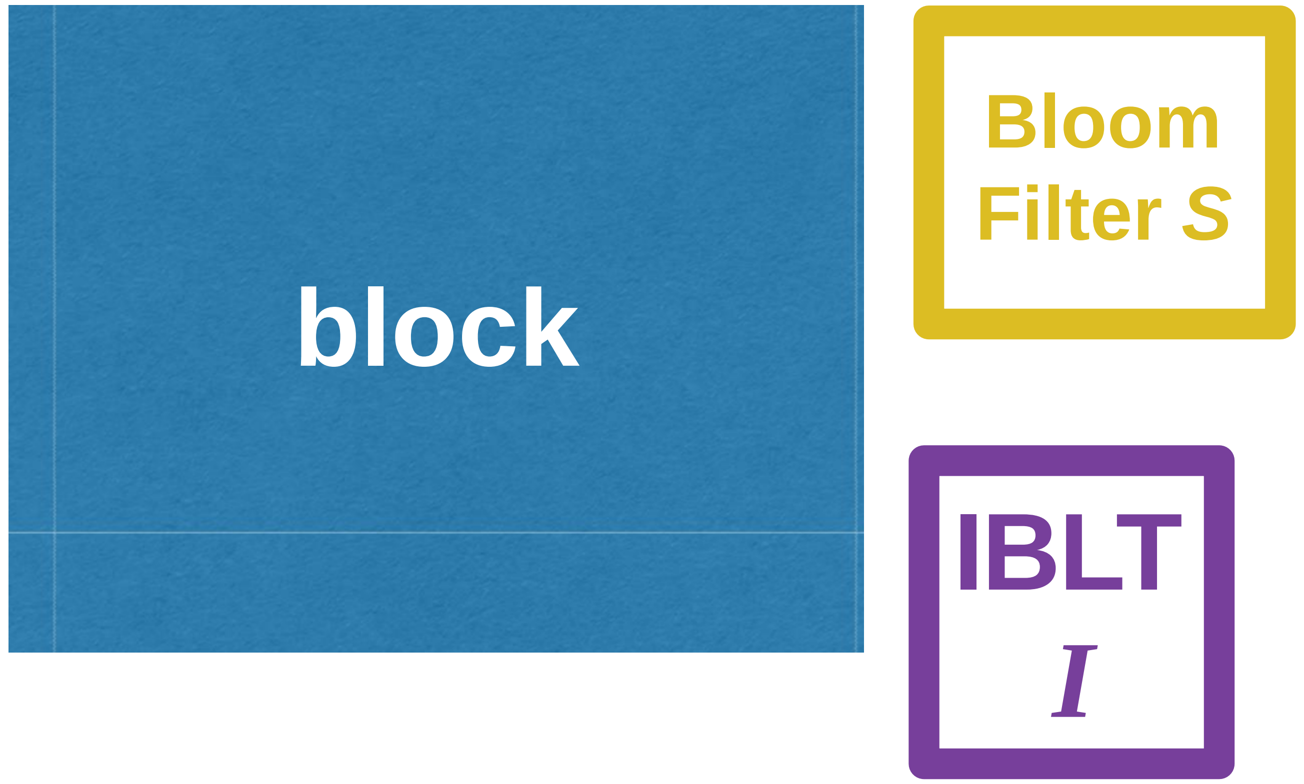
Receiver 

block

Graphene: Protocol 1

- The block is a subset of the mempool

Sender 



$$f_S = \frac{a}{m - n}$$

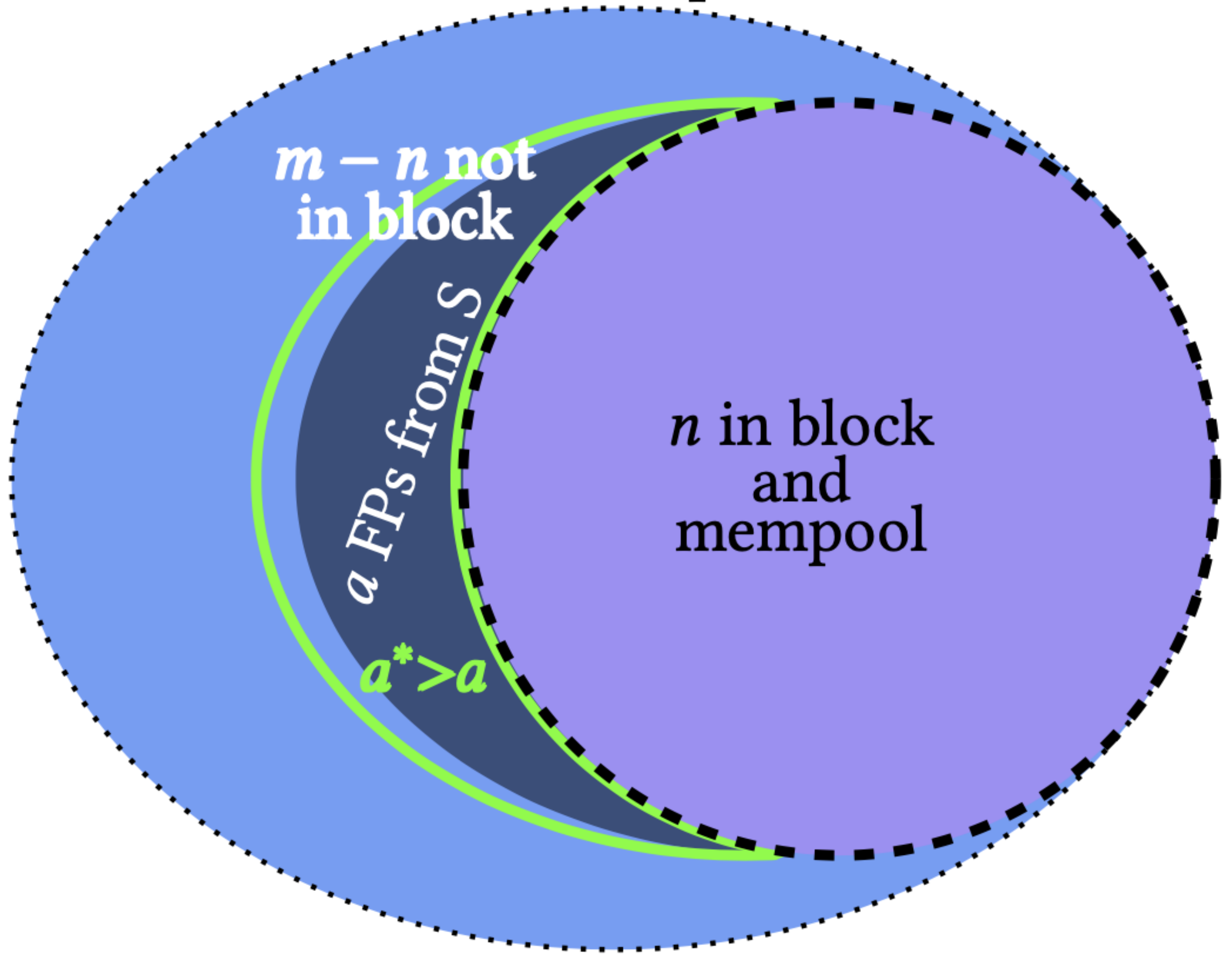


Figure 4: [Protocol 1] Passing m mempool transactions through S results in a FPs (in dark blue). A green outline illustrates $a^* > a$ with β-assurance, ensuring IBLT I decodes.

Graphene: Protocol 1

- The block is a subset of the mempool

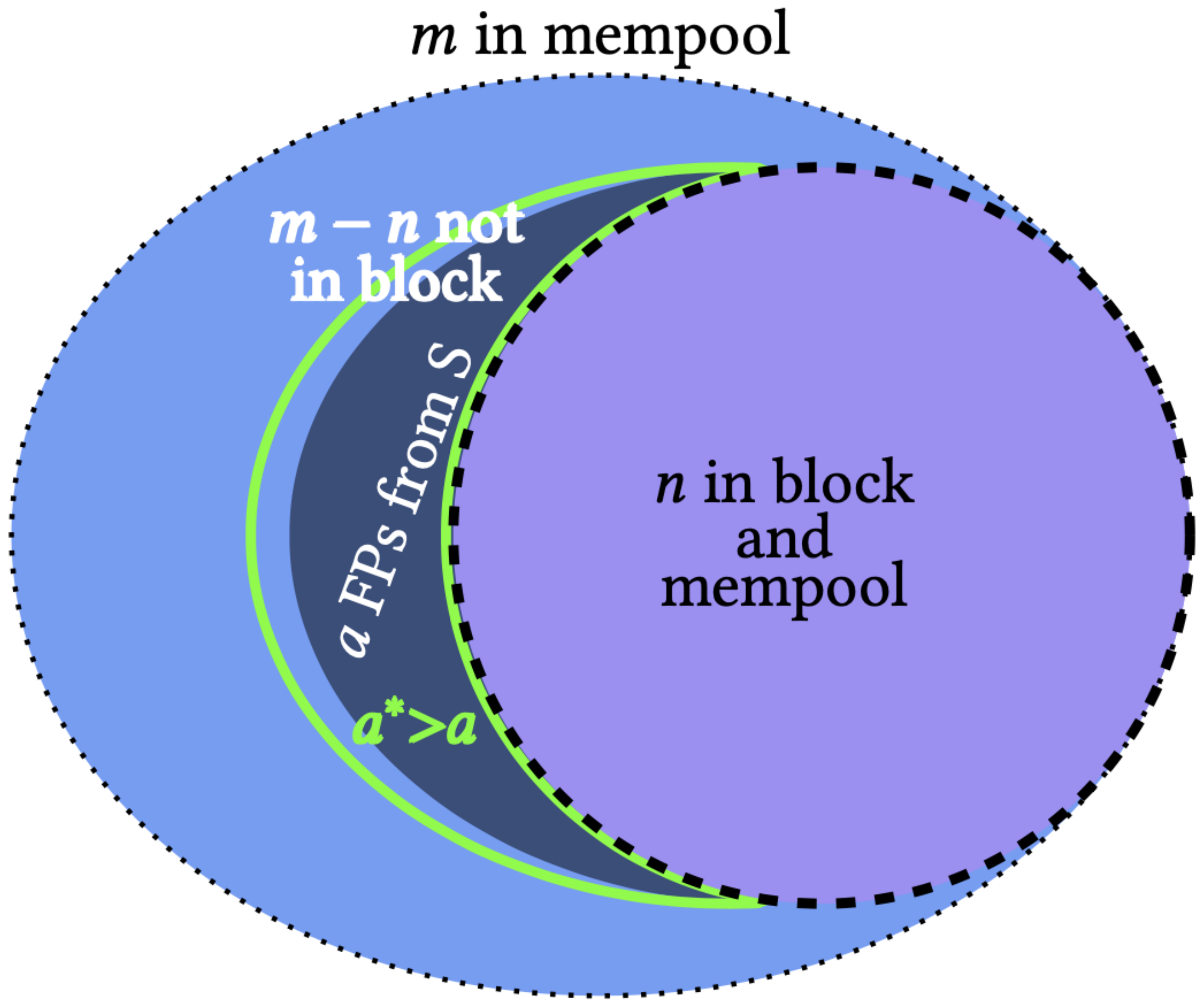
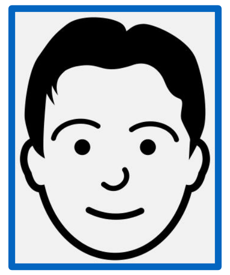
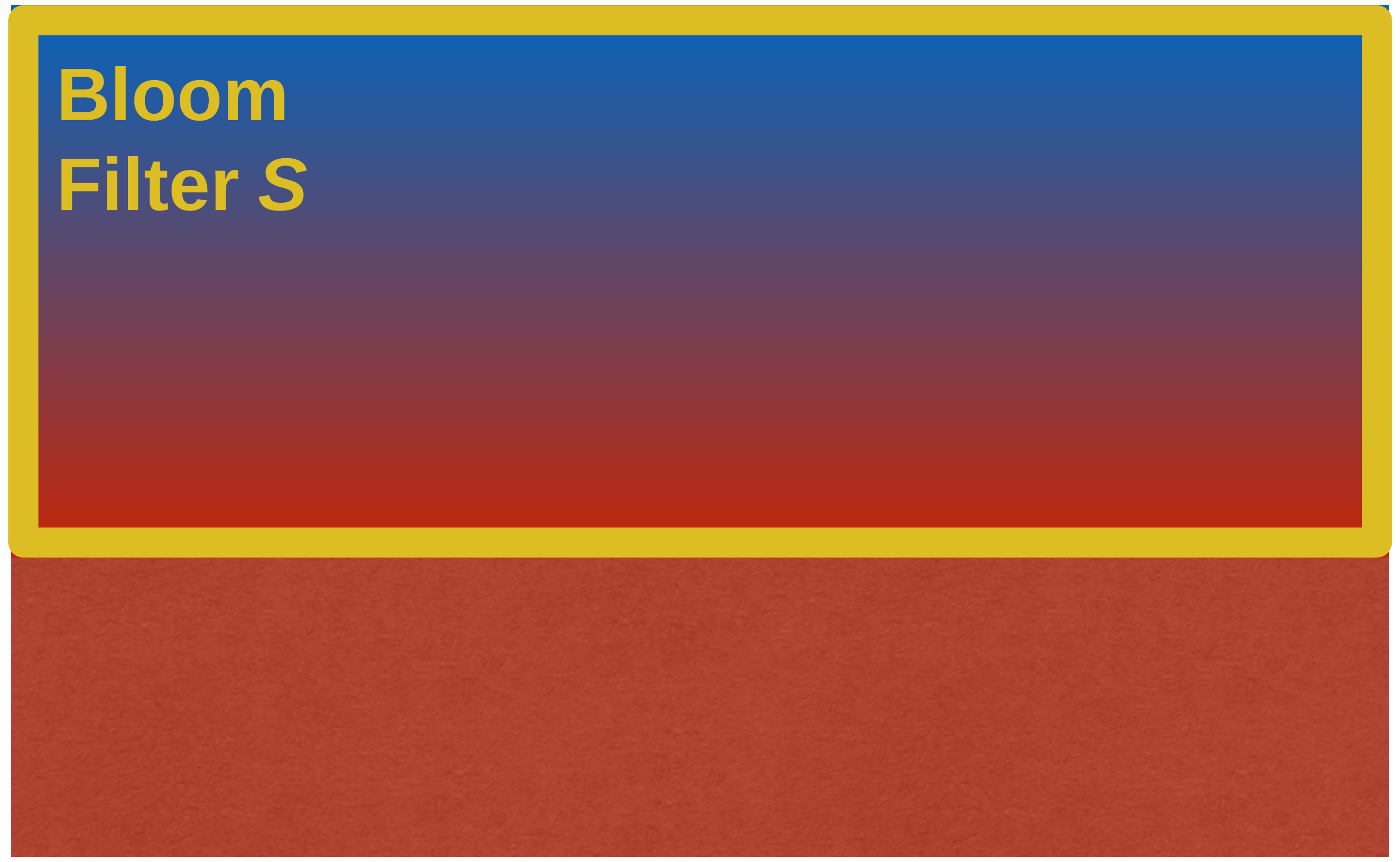


Figure 4: [Protocol 1] Passing m mempool transactions through S results in a FPs (in dark blue). A green outline illustrates $a^* > a$ with β-assurance, ensuring IBLT I decodes.

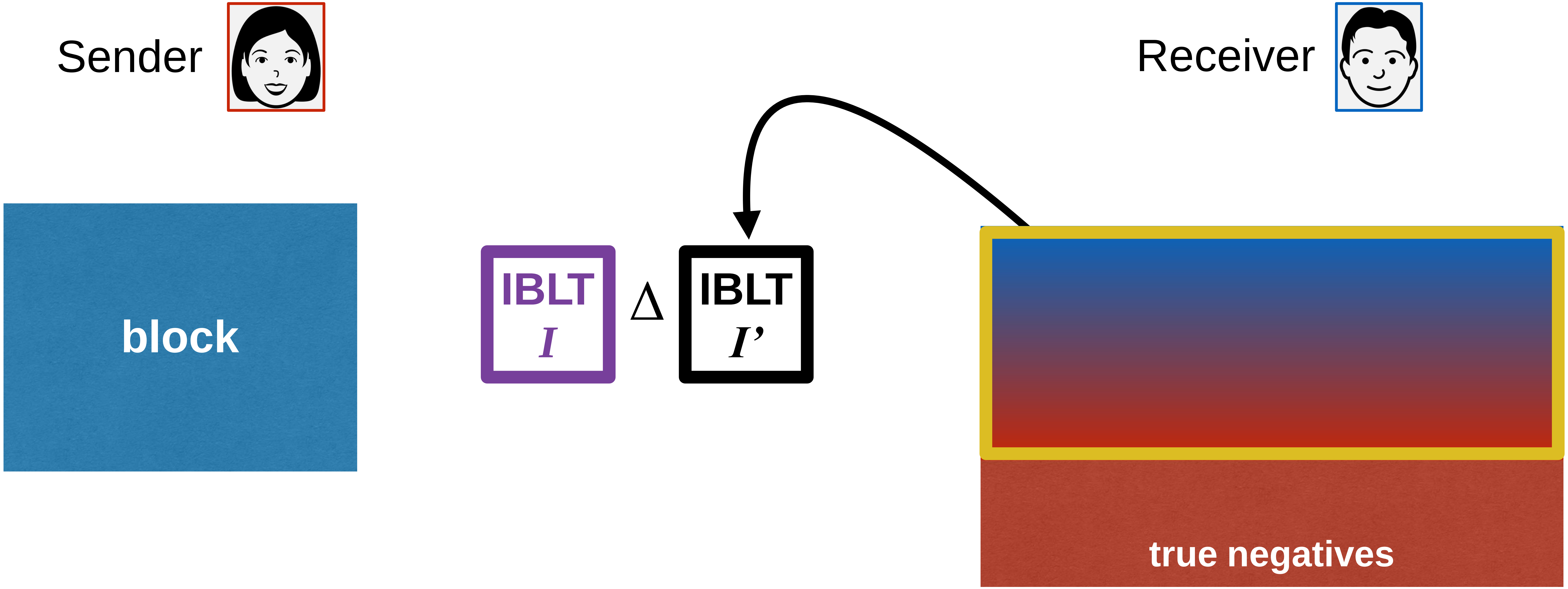
Receiver





Graphene: Protocol 1

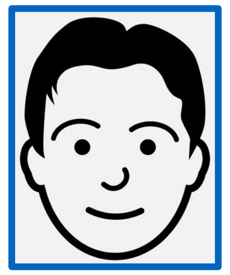
- The block is a subset of the mempool

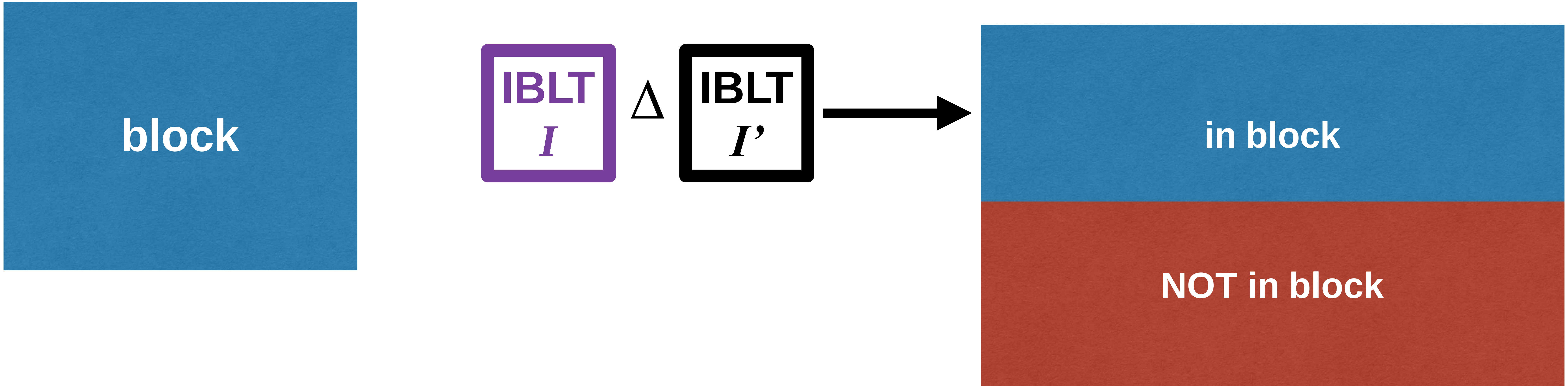


Graphene: Protocol 1

- The block is a subset of the mempool

Sender 

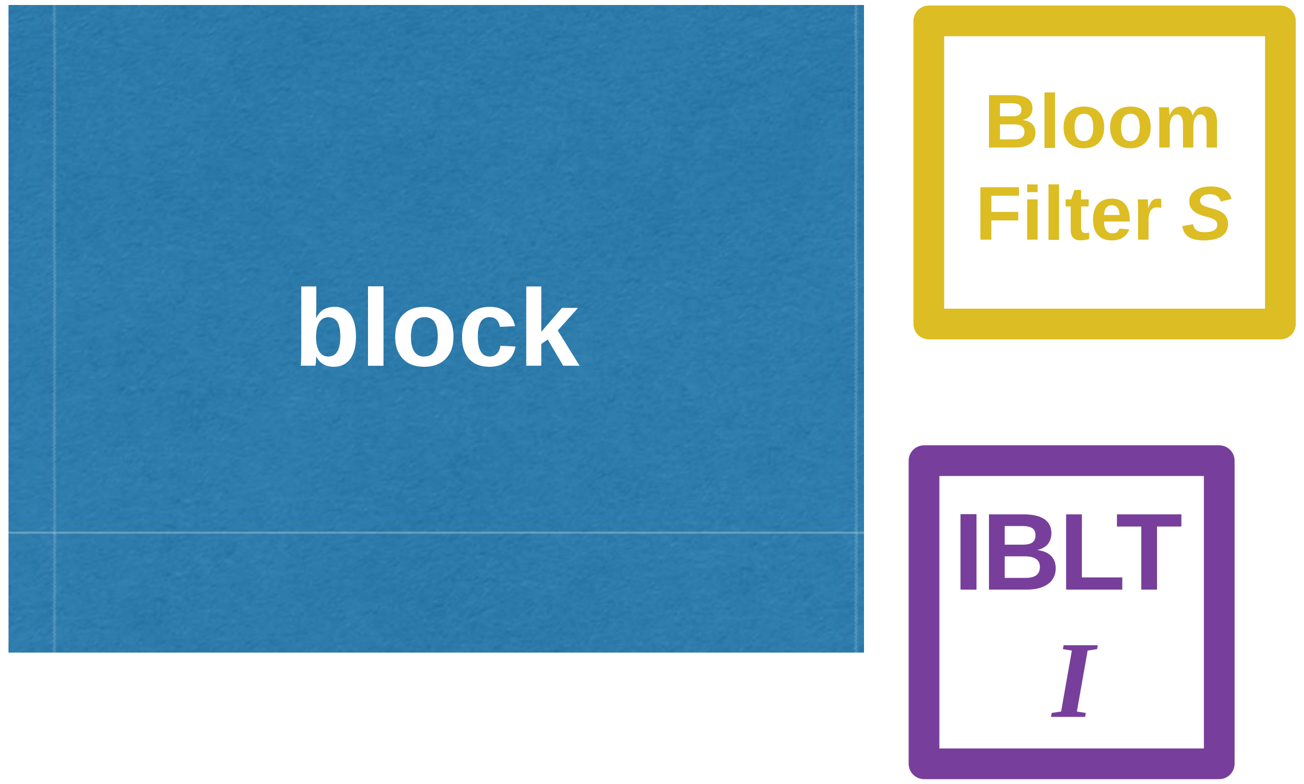
Receiver 



Graphene: Protocol 1

- The block is a subset of the mempool

Sender 



$$f_S = \frac{a}{m-n},\ a^* = (1+\delta)a$$

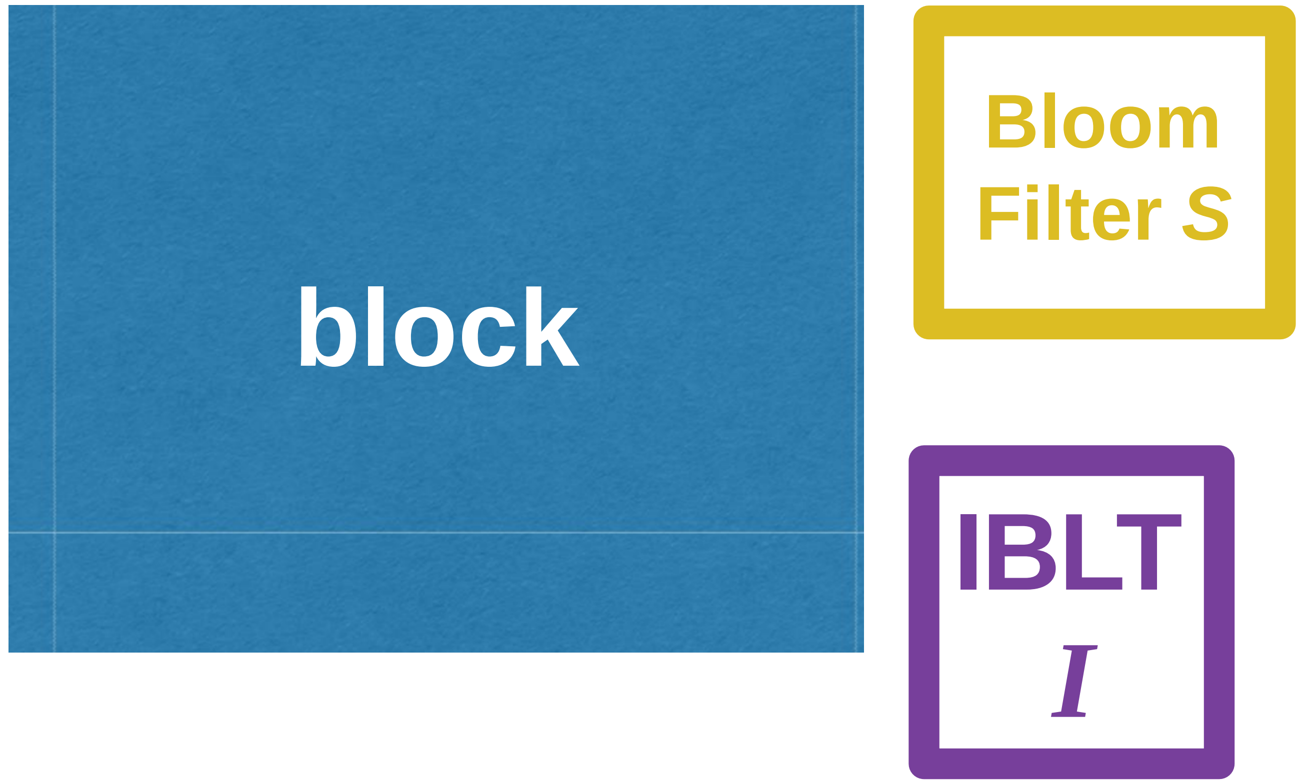
$$T_I = r\tau(1+\delta)a,\ T_{BF} = \frac{-n \ln f_S}{8(\ln 2)^2}$$

$$T = T_{BF} + T_I = \frac{-n \ln(\frac{a}{m-n})}{8(\ln 2)^2} + r\tau(1+\delta)a$$

Graphene: Protocol 1

- The block is a subset of the mempool

Sender 



$$T(a) = T_{BF} + T_I = \frac{-n\ln\left(\frac{a}{m-n}\right)}{8(\ln 2)^2} + r\tau(1+\delta)a$$

$$a = n/(8r\tau\, ln^2 2),\ \delta = 0$$

$$accurate\ only\ for\ a > 100$$

Graphene: Protocol 1

- **THEOREM 1: Derivation of a$*$**

Let m be the size of a mempool that contains all n transactions from a block. If a is the number of false positives that result from passing the mempool through Bloom filter S with FPR f_S , then $a^* \geq a$ with probability β when

$$a^* = (1+\delta)a,$$

$$\textit{where } \delta = \frac{1}{2}\left(s + \sqrt{s^2 + 8s}\right) \textit{ and } s = \frac{-\ln(1-\beta)}{a}$$

Graphene: Protocol 1

- **THEOREM 1: Derivation of a∗, PROOF**

LEMMA 1: *Let A be the sum of i independent Bernoulli trials $A_1, \ldots, A_i$, with mean μ = E[A]. Then for δ > 0*

$$Pr[A \geq (1+\delta)\mu] \leq Exp\left(-\frac{\delta^2}{2+\delta}\mu\right)$$

Starting from the well-known Chernoff bound

$$\begin{aligned} Pr[A \geq (1+\delta)\mu] &\leq \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu} \\ &= \mathrm{Exp}(\mu(\delta - (1+\delta)ln(1+\delta))) \\ &\leq \mathrm{Exp}\left(\mu\left(\delta - (1+\delta)\left(\frac{2\delta}{2+\delta}\right)\right)\right) \\ &= \mathrm{Exp}\left(\frac{-\delta^2}{2+\delta}\mu\right) \end{aligned}$$

$$ln(1+x) \geq \frac{x}{1+x/2} = \frac{2x}{2+x} \text{ for } x > 0$$

Graphene: Protocol 1

- **THEOREM 1: Derivation of a∗, PROOF**

There are m − n potential false positives that pass through S. They are a set $A_1, \ldots, A_{m-n}$ of independent Bernoulli trials such that $Pr[A_i = 1] = f_S$. Let $\Sigma_{i-1}^{m-n} A_i = A$ and $\mu = E[A] = f_S(m-n) = a$. From Lemma 1, we have

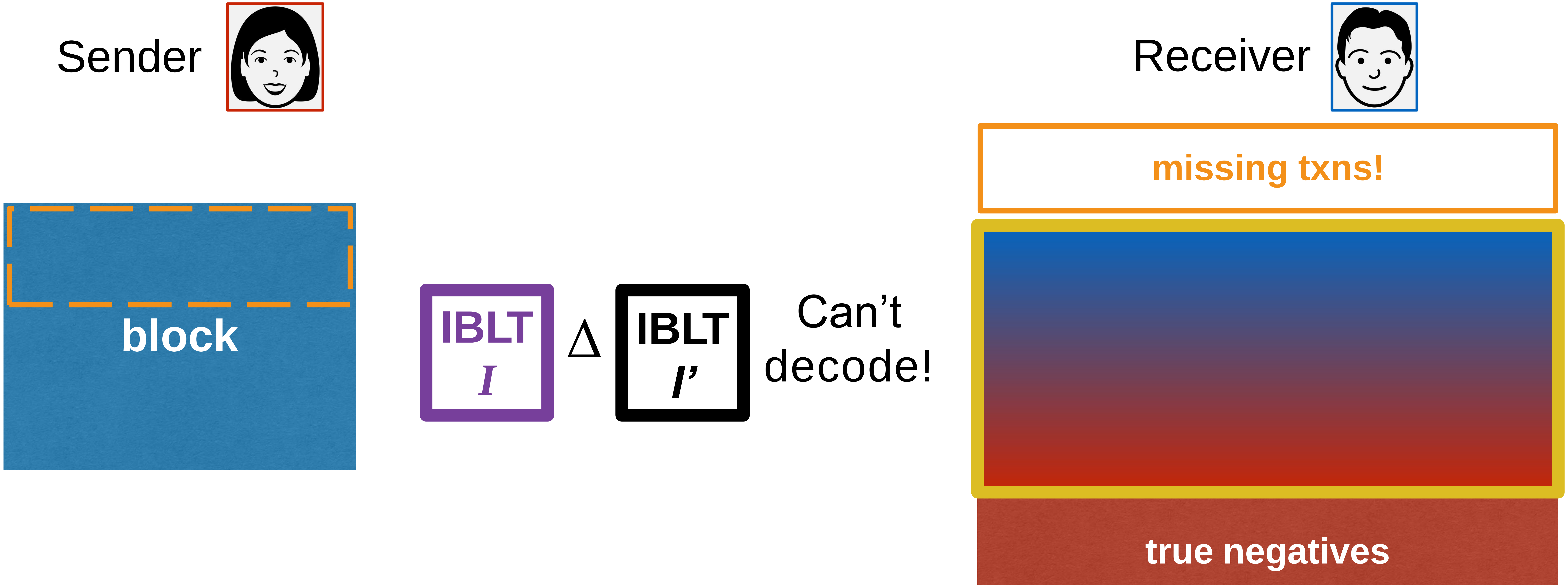
$$Pr[A \geq (1+\delta)\mu] \leq \mathrm{Exp}\left(-\frac{\delta^2}{2+\delta}\mu\right)$$

$$\beta = 1 - \mathrm{Exp}\left(-\frac{\delta^2}{2+\delta}a\right)$$

$$\delta = \frac{1}{2}\left(s + \sqrt{s^2 + 8s}\right), \text{ where } s = \frac{-\ln(1-\beta)}{a}$$

Graphene: Protocol 2

- The block is *not* a subset of the mempool



Graphene: Protocol 2

- The block is *not* a subset of the mempool

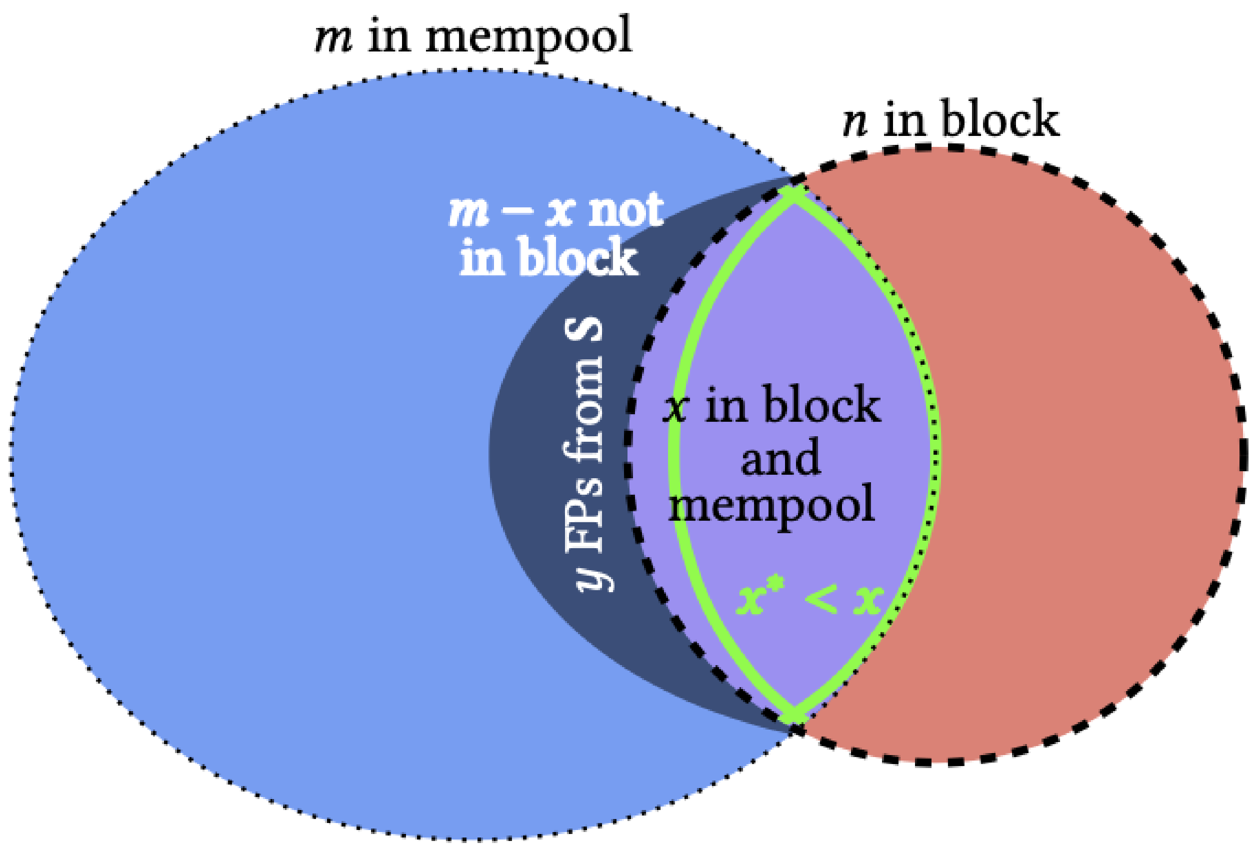
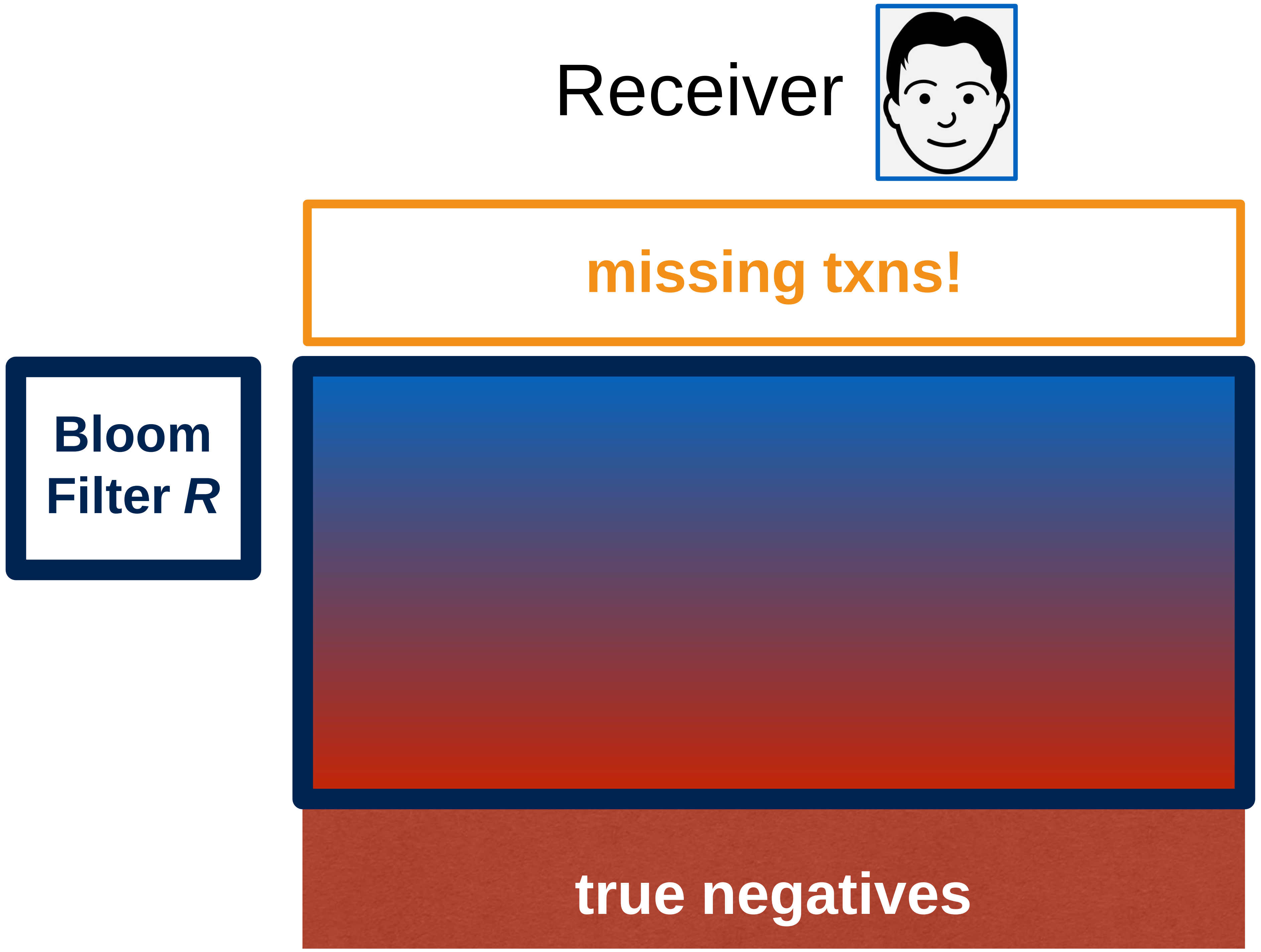


Figure 5: [Protocol 2] Passing m transactions through S results in z positives, obscuring a count of x TPs (purple) and y FPs (in dark blue). From z, we derive $x^* < x$ with β-assurance (in green).



Graphene: Protocol 2

- The block is *not* a subset of the mempool

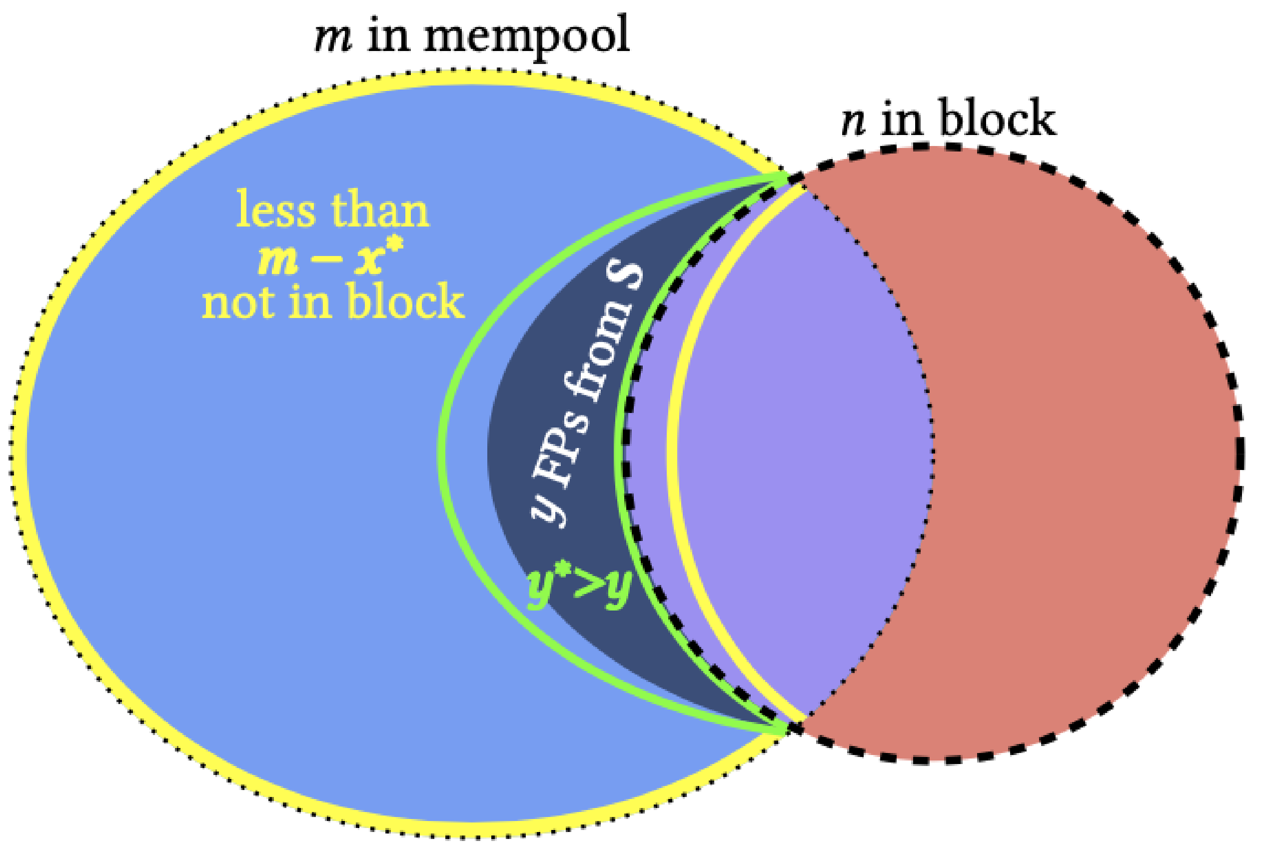
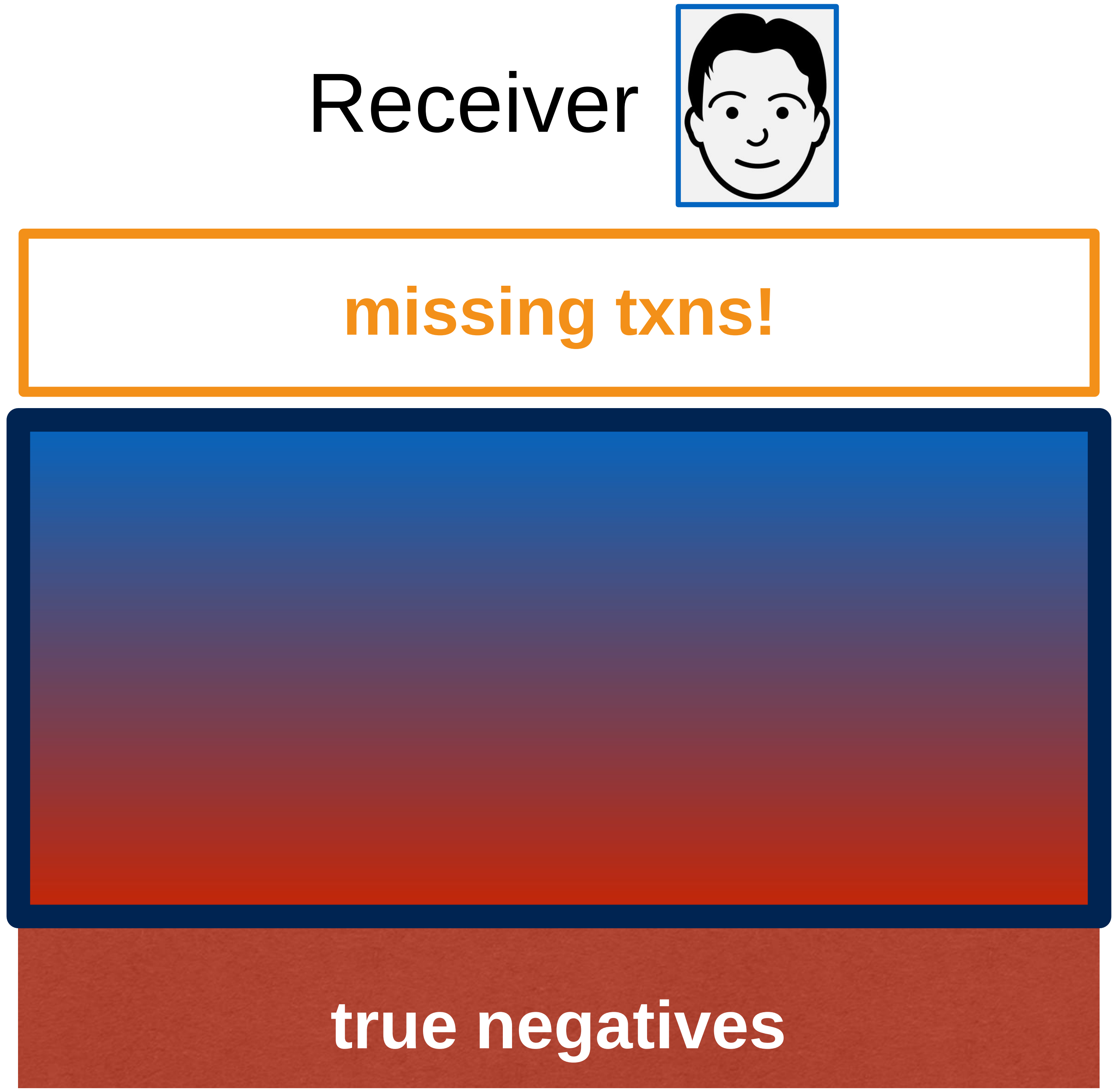


Figure 6: [Protocol 2] From our bound $m - x^* > m - x$ with β-assurance (in yellow), we can derive a bound for the false positives from S as $y^* > y$ with β-assurance outlined in green.

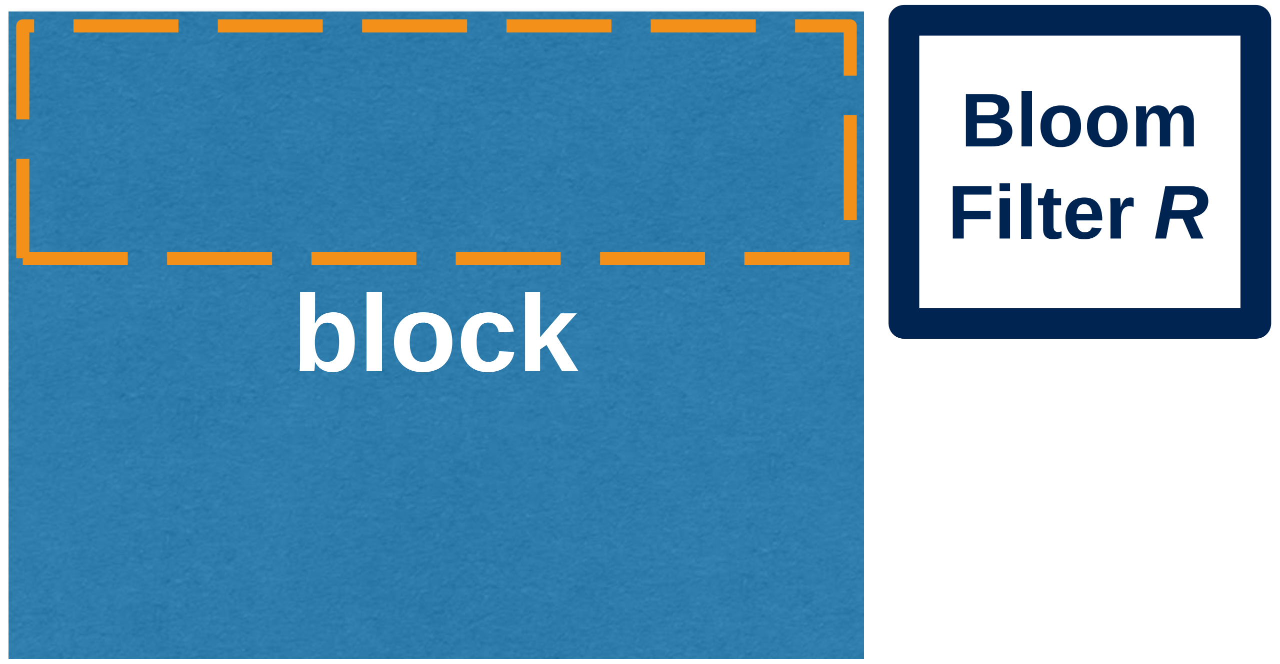
Bloom Filter *R*

$$f_R = \frac{b}{n - x}$$



- The block is *not* a subset of the mempool

Sender



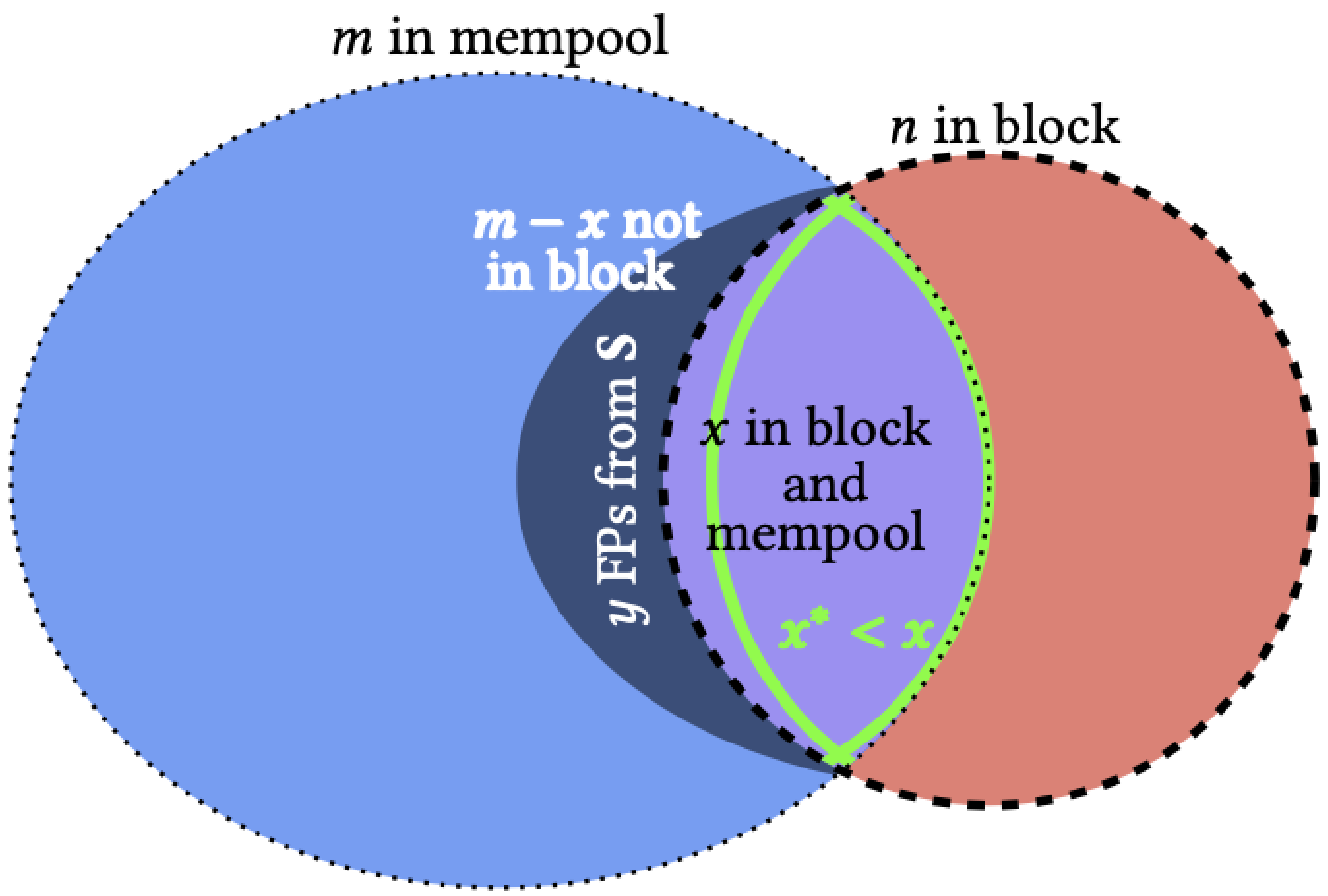
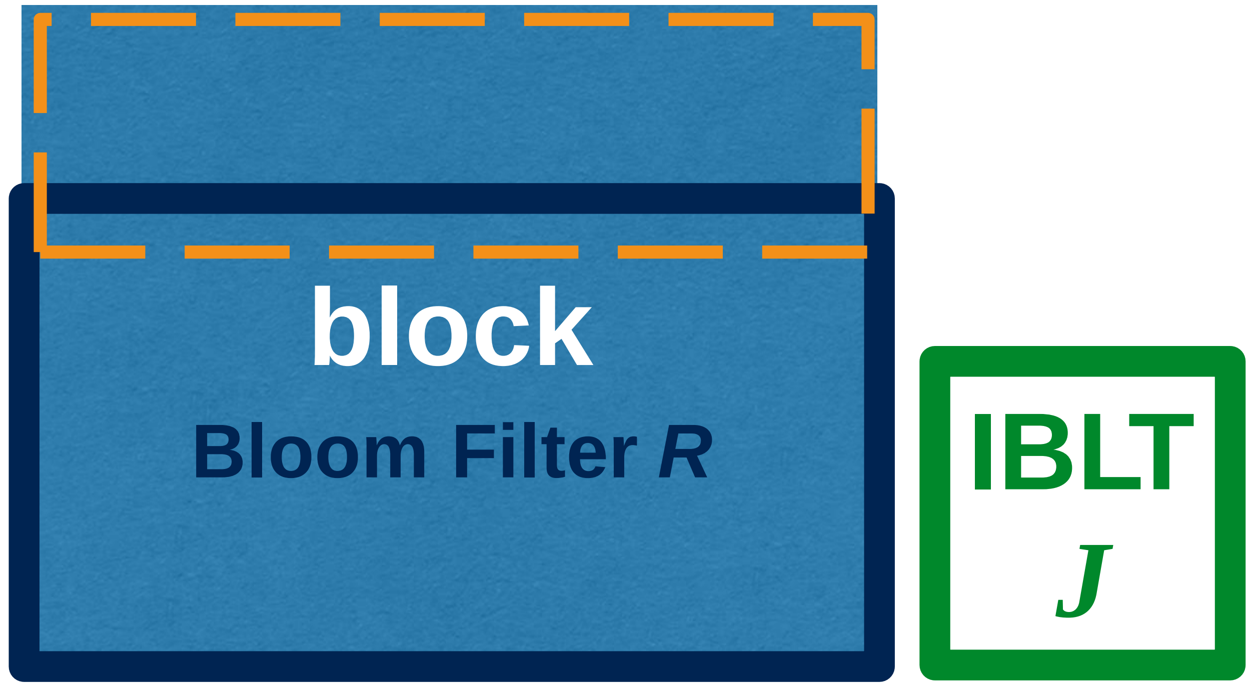


Figure 5: [Protocol 2] Passing m transactions through S results in z positives, obscuring a count of x TPs (purple) and y FPs (in dark blue). From z, we derive $x^* < x$ with β-assurance (in green).

Graphene: Protocol 2

- The block is *not* a subset of the mempool

Sender 



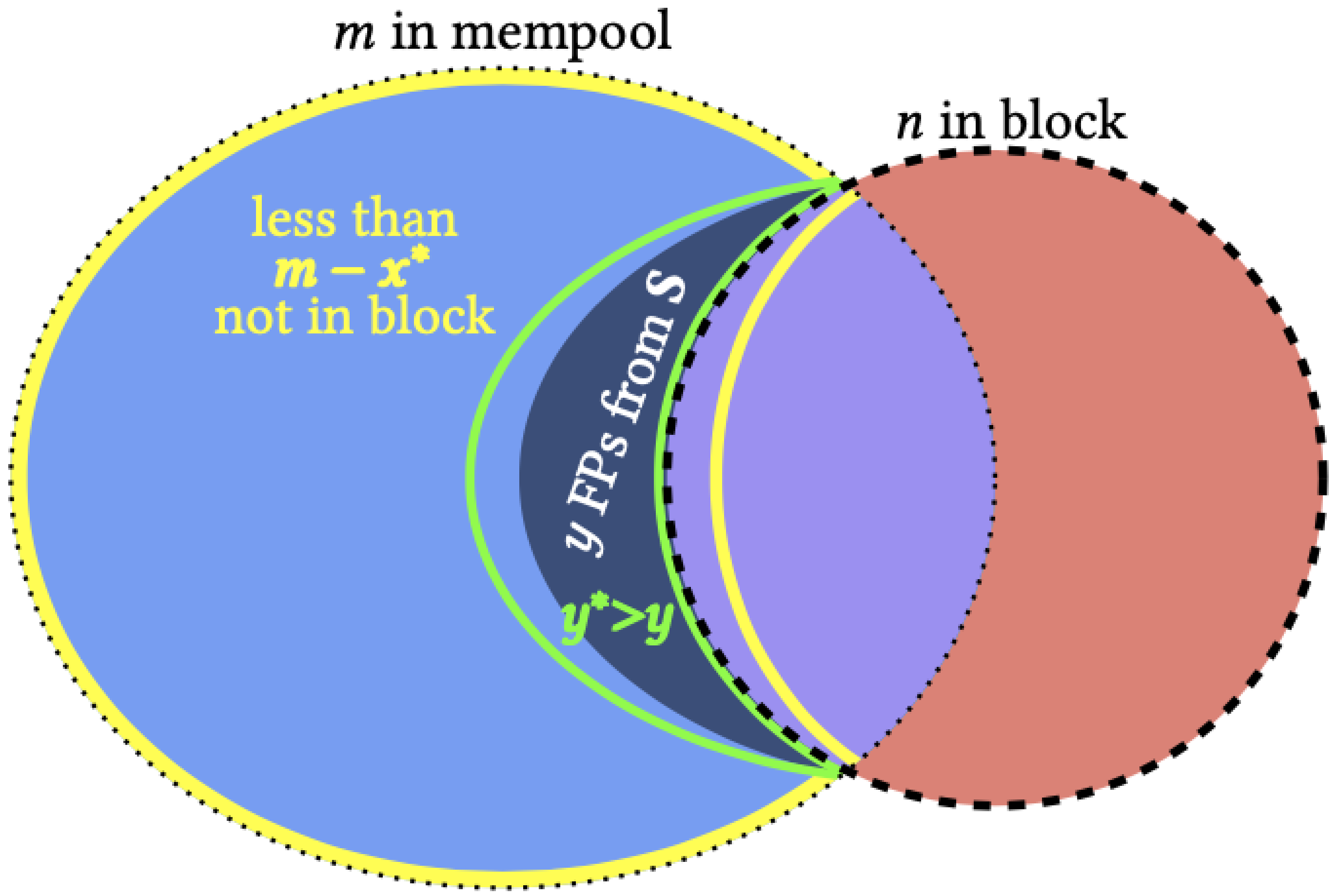
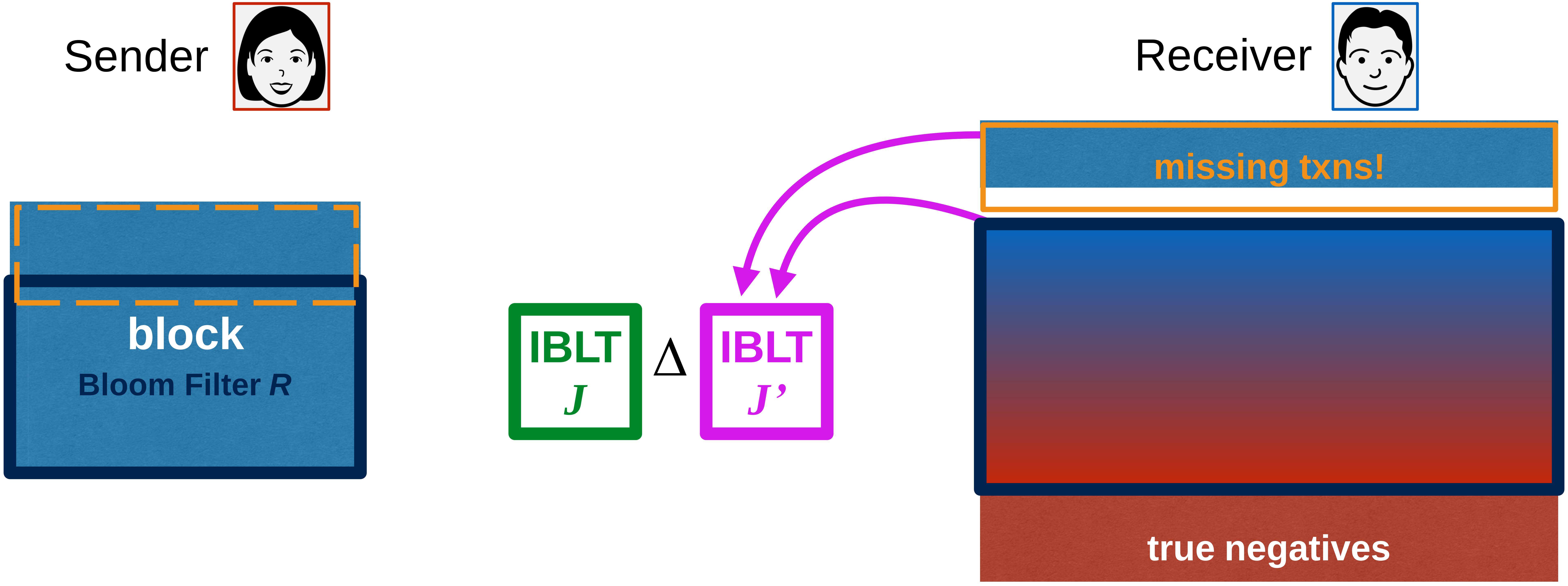


Figure 6: [Protocol 2] From our bound $m - x^* > m - x$ with β-assurance (in yellow), we can derive a bound for the false positives from S as $y^* > y$ with β-assurance outlined in green.

Graphene: Protocol 2

- The block is *not* a subset of the mempool



Graphene: Protocol 2

- The block is *not* a subset of the mempool

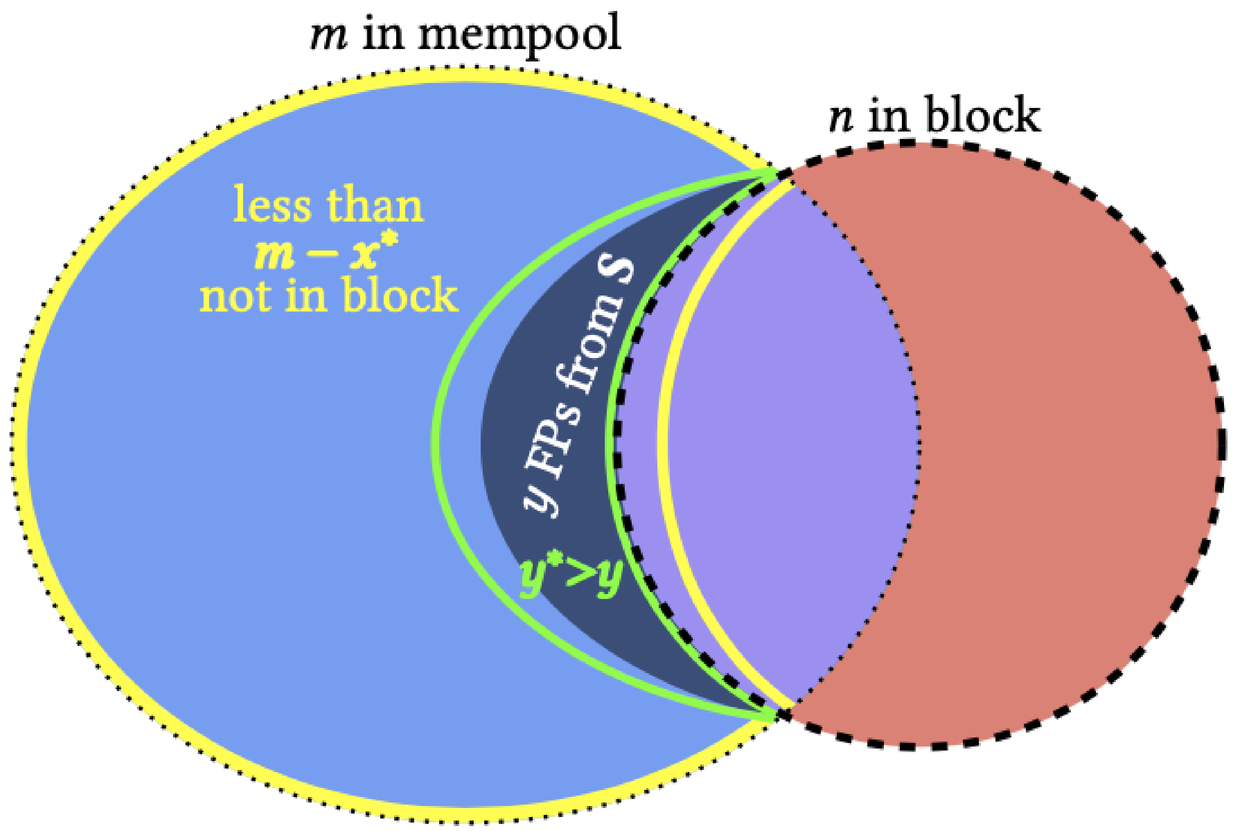
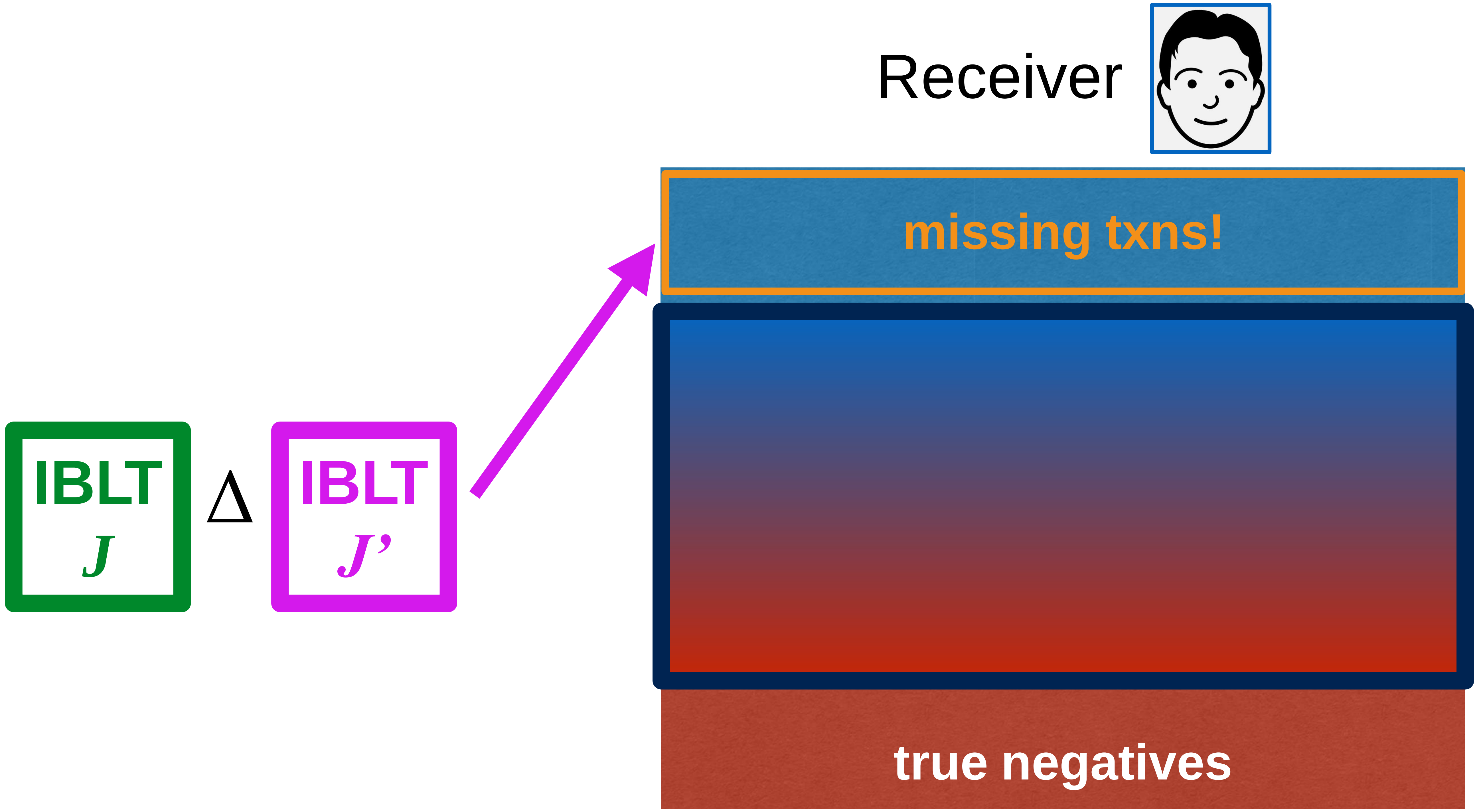


Figure 6: [Protocol 2] From our bound $m - x^* > m - x$ with β-assurance (in yellow), we can derive a bound for the false positives from S as $y^* > y$ with β-assurance outlined in green.



Graphene: Protocol 2

- **Parameterizing b**

We show below that $y^{*} = (1+\delta)y$. Thus, for protocol 2 the total size is

$$T(b) = \frac{z \ln(\frac{b}{n-x^*})}{8 \ln^2 2} + r\tau(1+\delta)b$$

$$b = z/(8r\tau \ln^2 2)$$

Graphene: Protocol 2

- **Using z to parameterize R and J**

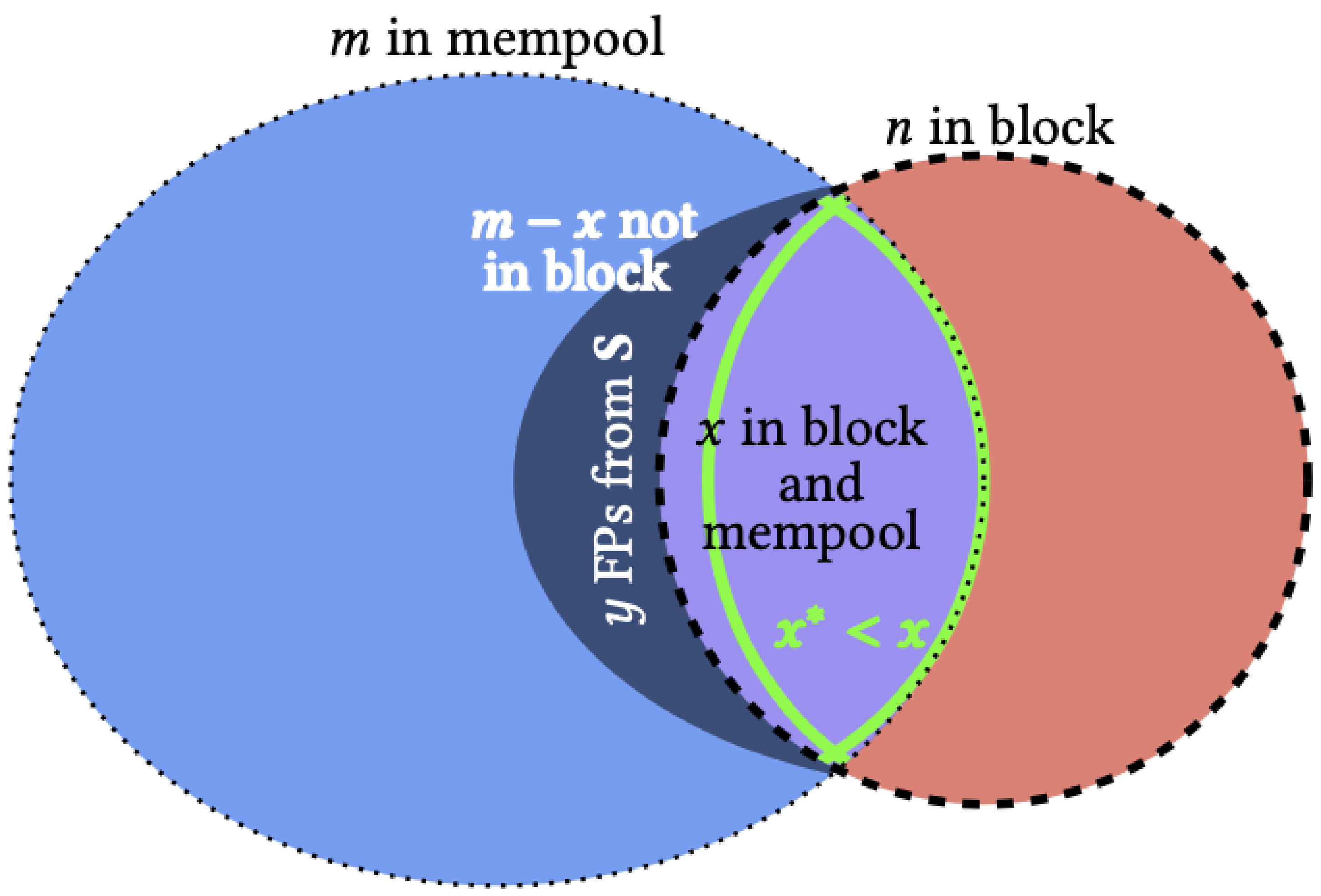


Figure 5: [Protocol 2] Passing m transactions through S results in z positives, obscuring a count of x TPs (purple) and y FPs (in dark blue). From z, we derive $x^* < x$ with β-assurance (in green).

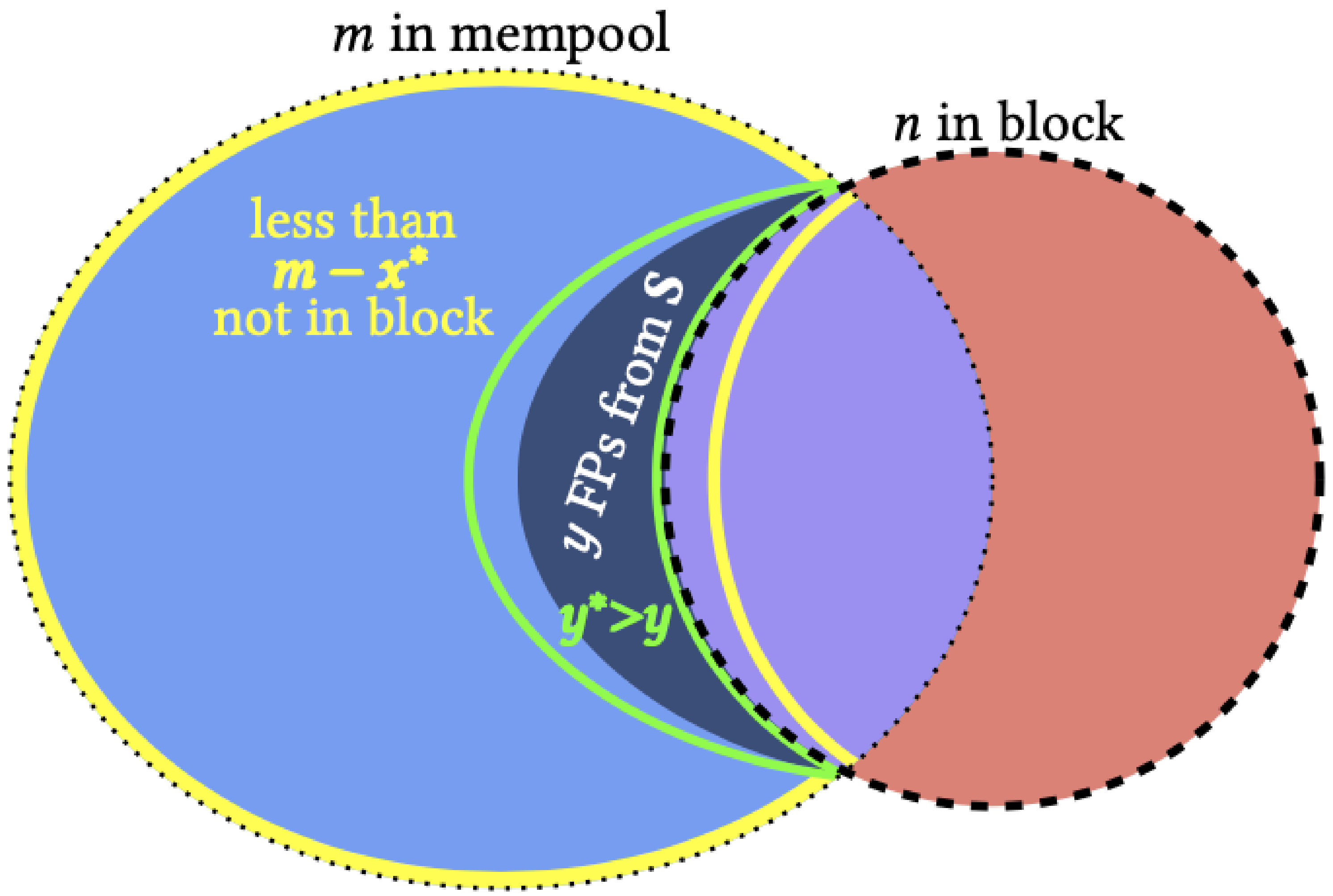


Figure 6: [Protocol 2] From our bound $m - x^* > m - x$ with β-assurance (in yellow), we can derive a bound for the false positives from S as $y^* > y$ with β-assurance outlined in green.

Graphene: Protocol 2

- **THEOREM 2:**

Let m be the size of a mempool containing $0 \leq x \leq n$ transactions from a block. Let $z = x + y$ be the count of mempool transactions that pass through S with FPR f_S, with true positive count x and false positive count y. Then $x^* \leq x$ with probability β when

$$x^* = \arg\min_{x^*} Pr[x \leq x^*; z, m, f_S] \leq 1 - \beta.$$

$$\textit{where } Pr[x \leq k; z, m, f_S] \leq \sum_{i=0}^{k} \left(\frac{e^{\delta_k}}{(1+\delta_k)^{1+\delta_k}} \right)^{(m-k)f_S}$$

$$\textit{and } \delta_k = \frac{z-k}{(m-k)f_S} - 1.$$

Graphene: Protocol 2

- **THEOREM 2 PROOF:**

Let $Y_1, \ldots, Y_{m-x}$ be independent Bernoulli trials representing transactions not in the block that might be false positives such that $\Pr[Yi = 1] = f_S$*.*
Let $\Sigma_{i-1}^{m-n} Y_i = Y$ *and* $y = E[Y]$*.*
For a given value x, we can compute Pr[Y ≥ y], the probability of at least y false positives passing through the sender's Bloom filter. We apply a Chernoff bound:

$$Pr[y; z, x, m] = \quad Pr[Y \geq (1+\delta)\mu] \leq \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}$$

Graphene: Protocol 2

- **THEOREM 2 PROOF:**

where $\delta > 0$, and $\mu = E[Y] = (m-x)f_S$. By setting $(1+\delta)\mu = z - x$ and solving for δ, we have

$$(1+\delta)(m-x)f_S = z - x$$

$$\delta = \frac{z-x}{(m-x)f_S} - 1.$$

Graphene: Protocol 2

- **THEOREM 2 PROOF:**

$y = z - x$

$$Pr[x \leq k; z, m, f_S] = \sum_{i=0}^{k} Pr[y; z, k, m]$$

$$\leq \sum_{i=0}^{k} \left(\frac{e^{\delta_k}}{(1+\delta_k)^{1+\delta_k}} \right)^{(m-k)f_S}$$

where $\delta_k = \frac{z-k}{(m-k)f_S} - 1.$

$$\arg\min_{x^*} Pr[x \leq x^*; z, m, f_S] \leq 1 - \beta$$

Graphene: Protocol 2

- **THEOREM 3:**

Let m be the size of a mempool containing $0 \le x \le n$ transactions from a block. Let $z = x + y$ be the count of mempool transactions that pass through S with FPR f_S, with true positive count x and false positive count y. Then $y^* \ge y$ with probability β when

$$y^* = (1+\delta)(m - x^*) f_S,$$

$$\text{where } \delta = \frac{1}{2}\left(s + \sqrt{s^2 + 8s}\right) \text{ and } s = \frac{-\ln(1-\beta)}{(m - x^*) f_S}$$

Graphene: Protocol 2

- **THEOREM 3 PROOF:**

We find $y^* = z - x^* \geq y$ *by applying Lemma 1 to* $\Sigma_{i=1}^{m-x^*} Y_i = Y$, *the sum of* $m - x^*$ *independent Bernoulli such that might be false positives such that* $\Pr[Yi = 1] = f_S$ *trials and* $\mu = (m - x^*) f S$

$$Pr[Y \geq (1+\delta)\mu] \leq \mathrm{Exp}\left(-\frac{\delta^2}{2+\delta}\mu\right)$$

$$\beta = 1 - \mathrm{Exp}\left(-\frac{\delta^2}{2+\delta}(m - x^*) f_S\right)$$

$$\delta = \frac{1}{2}\left(s + \sqrt{s^2 + 8s}\right), \text{ where } s = \frac{-\ln(1-\beta)}{(m - x^*) f_S}.$$

Graphene: Protocol 2

- **THEOREM 3 PROOF:**

$$y^* = (1 + \delta)(m - x^*) f_S$$

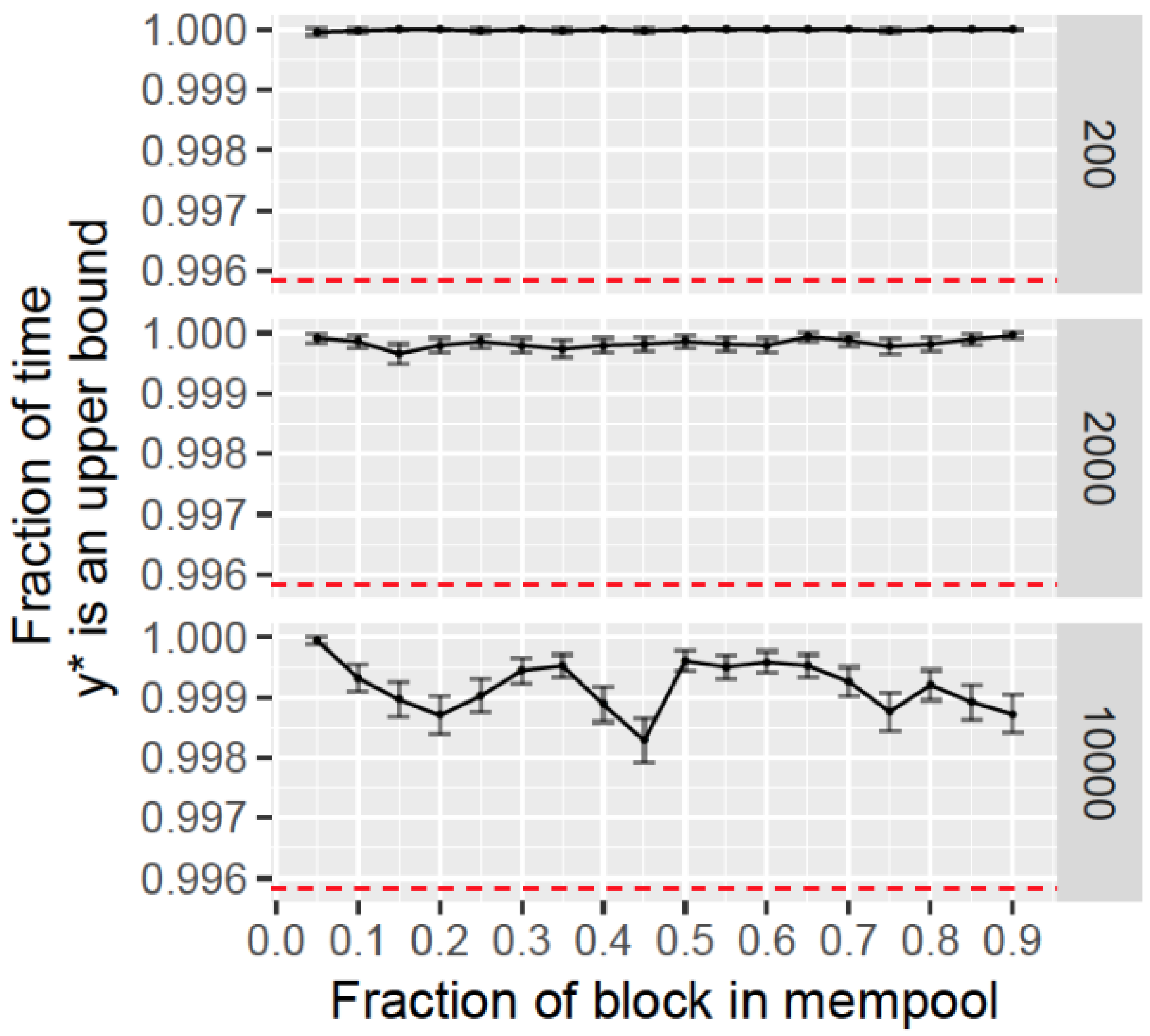
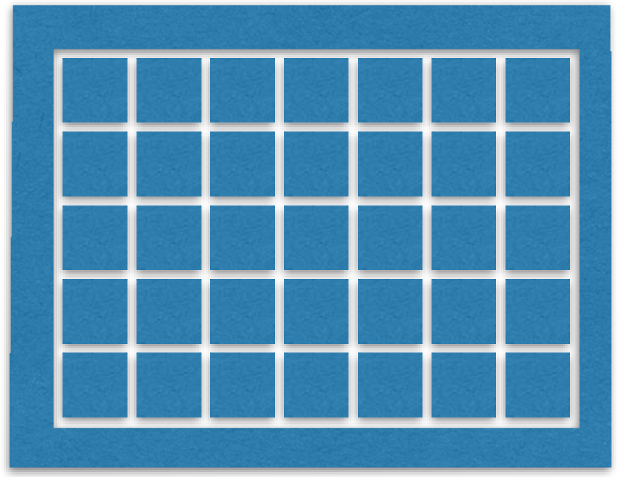
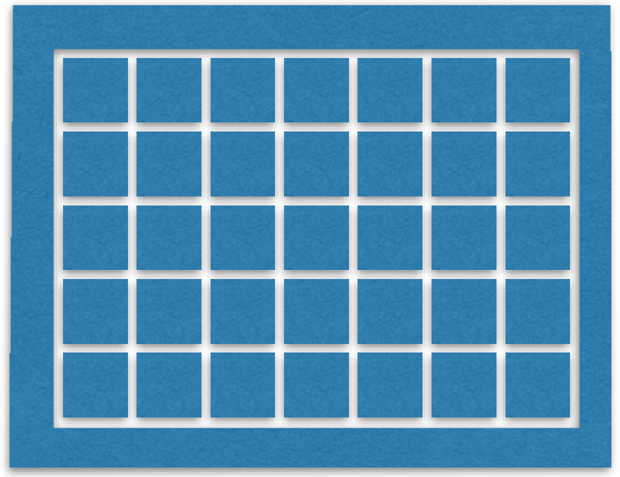


Figure 20: [Simulation, Protocol 2] The fraction of Monte Carlo experiments where $y^* > y$ via Theorem 3 compared to a desired bound of $\beta = 239/240$ (shown as a red dotted line).

ENHANCING IBLT PERFORMANCE



Δ



Let $H = (V, X, k)$ be a k-partite, k-uniform hypergraph, composed of a set of c vertices. Let $V = V_1 \cup \cdots \cup V_k$, where each V_i is a subset of c/k vertices (we enforce that c is divisible by k). X is a set of j hyperedges, each connecting k vertices, one from each of the Vi .

ENHANCING IBLT PERFORMANCE

IBLT

	row	count	value
Hash 1	1	2	j1⊗j2
	2	2	j3⊗j4
	3	1	j5
Hash 2	4	2	j1⊗j2
	5	2	j3⊗j5
	6	1	j4
Hash 3	7	2	j1⊗j2
	8	2	j4⊗j5
	9	1	j3

Hypergraph equivalent

edge	connected vertices
j1	v1,v4,v7
j2	v1,v4,v7
j3	v2,v5,v9
j4	v2,v6,v8
j5	v3,v5,v8

2-core

$V = V1 \cup V2 \cup V3$

$V1=\{v1,v2,v3\}$

$V2=\{v4,v5,v6\}$

$V3=\{v7,v8,v9\}$

– *j* items and hyper-edges
– *c* cells and vertices
– *k* hash functions and vertices connecting each edge

ENHANCING IBLT PERFORMANCE

$$H_{j,p} = \{(V, X, k) \mid E[decode((V, X, k))] \geq p, |X| = j\}$$

$$\underset{(V,X,k)\in\mathcal{H}_{j,p}}{\arg\min} |V|$$

ENHANCING IBLT PERFORMANCE

ALGORITHM 1: IBLT-Param-Search

```
SEARCH(j, k, p):
  c_l = 1
  c_h = c_max
  trials = 0
  success = 0
  L = (1 − p)/5
  WHILE c_l ≠ c_h:
      trials += 1
      c = (c_l + c_h)/2
      IF decode(j, k, c):
          success += 1
      conf=conf_int(success, trials)
      r = success/trials
      IF r − conf ≥ p:
          c_h = c
      IF (r + conf ≤ p):
          c_l = c
      IF (r − conf > p − L) and (r + conf < p + L):
          c_l = c
  RETURN c_h
```

ENHANCING IBLT PERFORMANCE

- Larger but computation is faster
- Create a universal lookup table

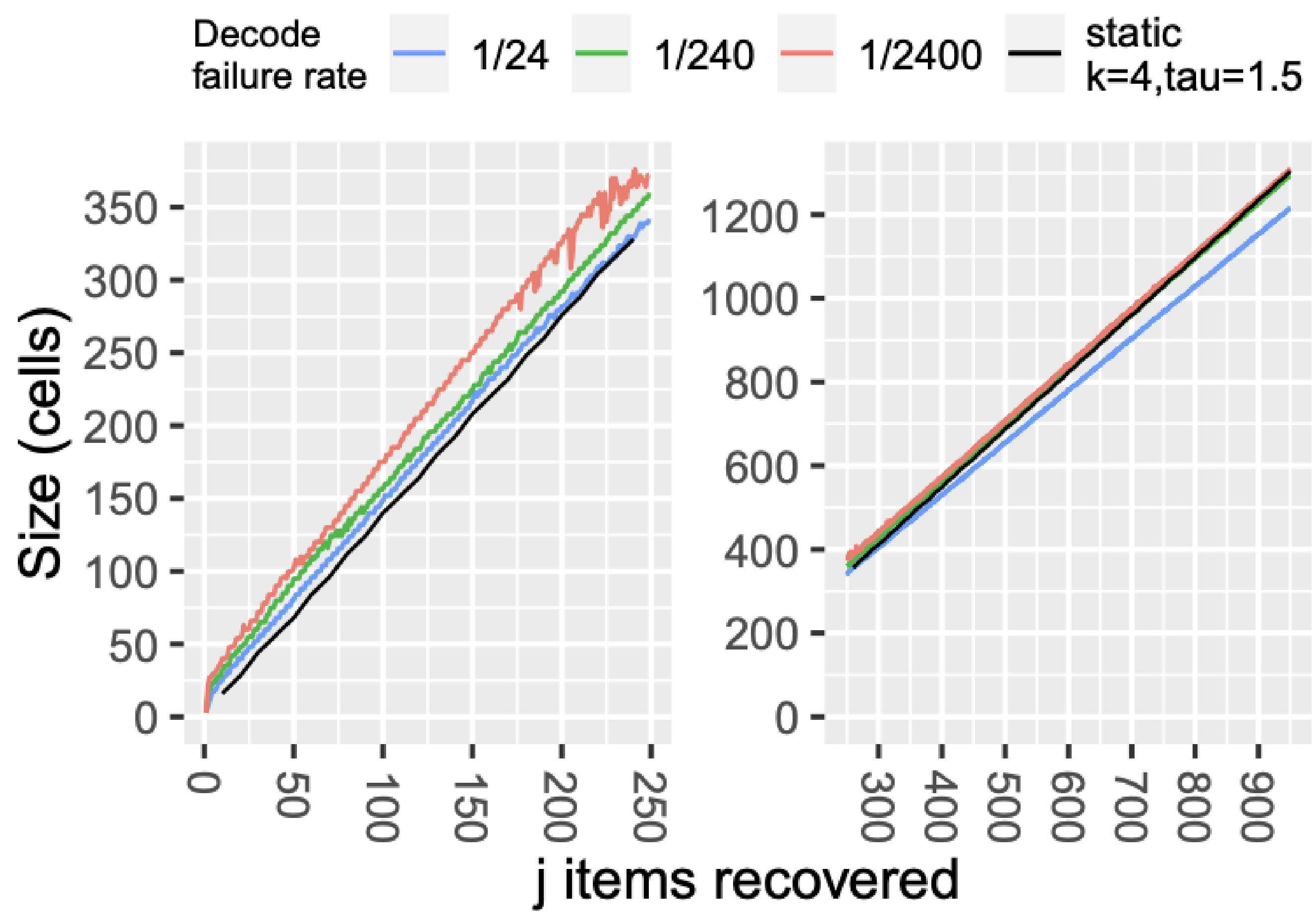


Figure 10: Size of optimal IBLTs (using Alg. 1) given a desired decode rate; with a statically parameterized IBLT ($k = 4$, $\tau = 1.5$) in black. For clarity, the plot is split on the x-axis. Decode rates are shown in Fig. 7.

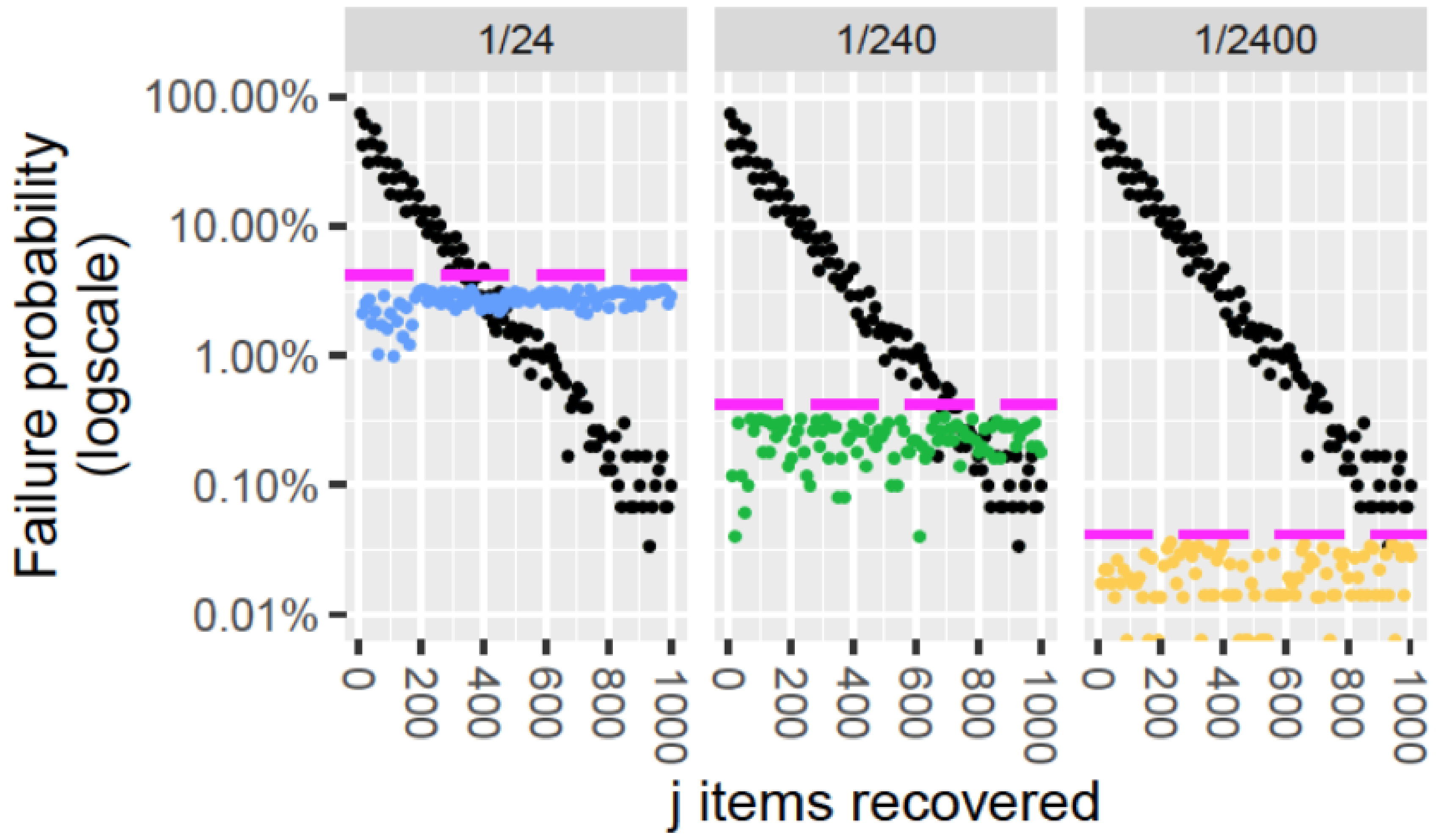


Figure 7: Parameterizing an IBLT statically results in poor decode rates. The black points show the decode failure rate for IBLTs when $k = 4$ and $\tau = 1.5$. The blue, green and yellow points show decode failure rates of optimal IBLTs, which always meet a desired failure rate on each facet (in magenta). Size shown in Fig. 10.

ENHANCING IBLT PERFORMANCE

- Ping-Pong Decoding

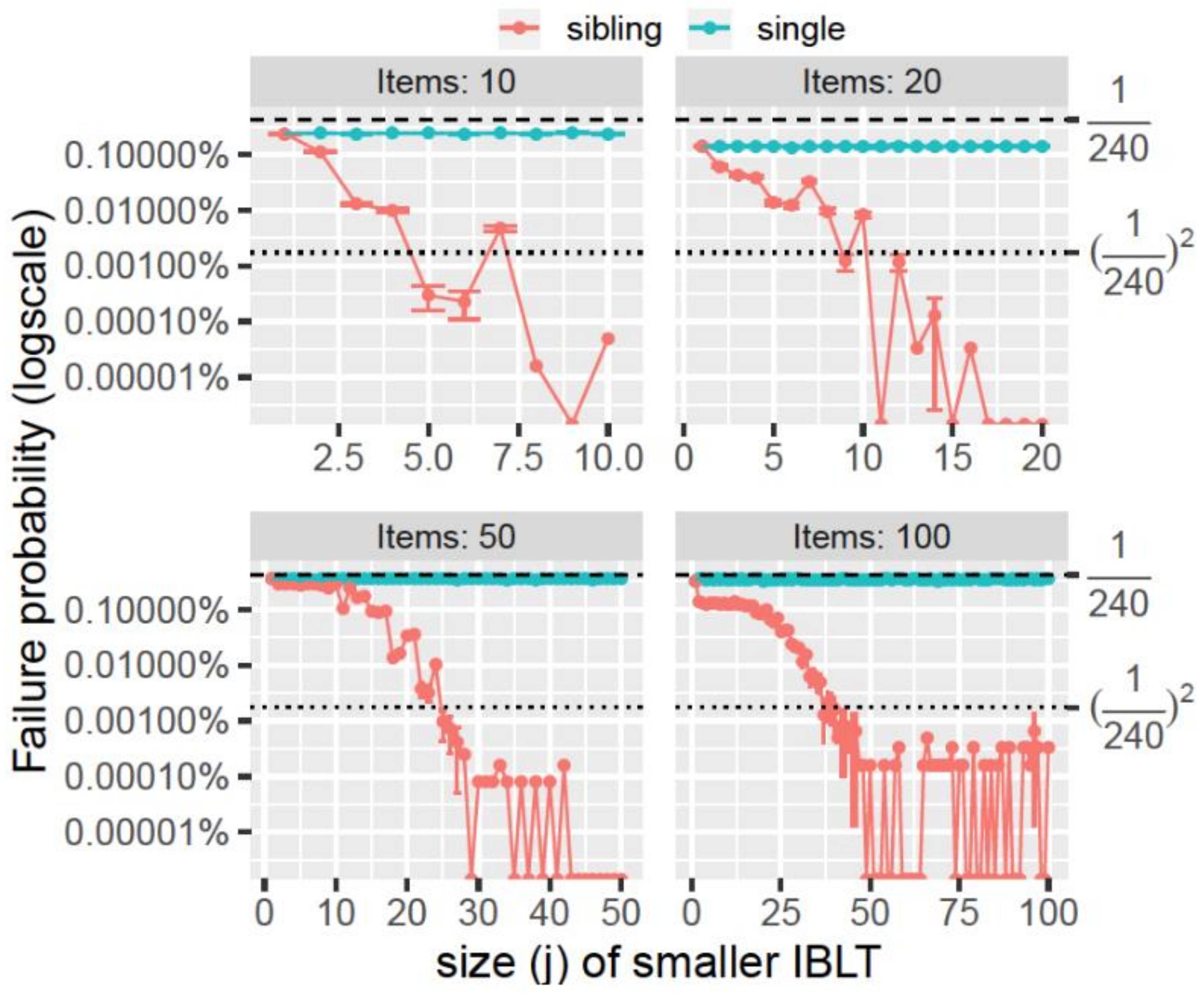
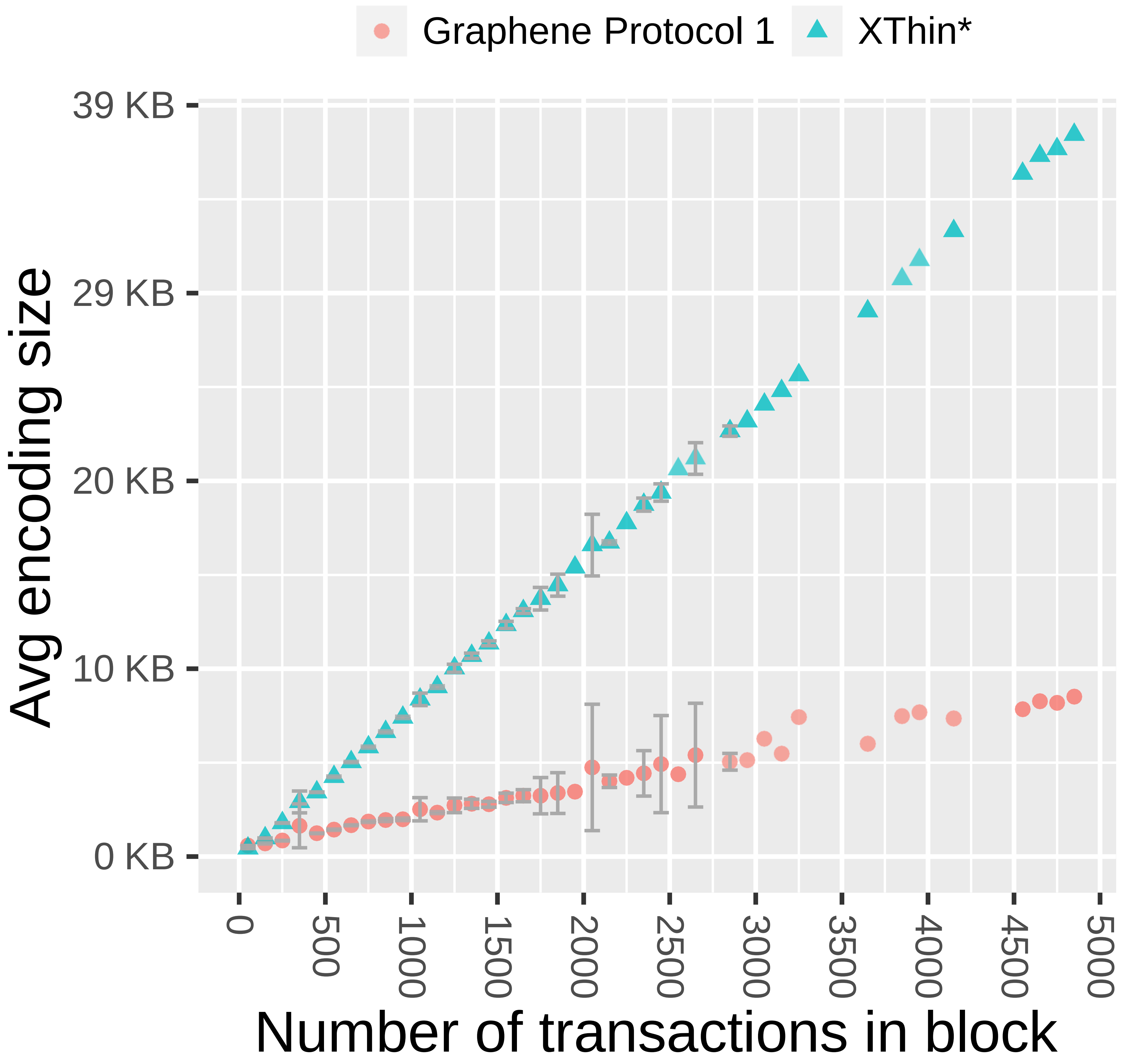


Figure 11: Decode rate of a single IBLT (parameterized for a 1/240 failure rate) versus the improved *ping-pong decode* rate from using a second, smaller IBLT with the same items.

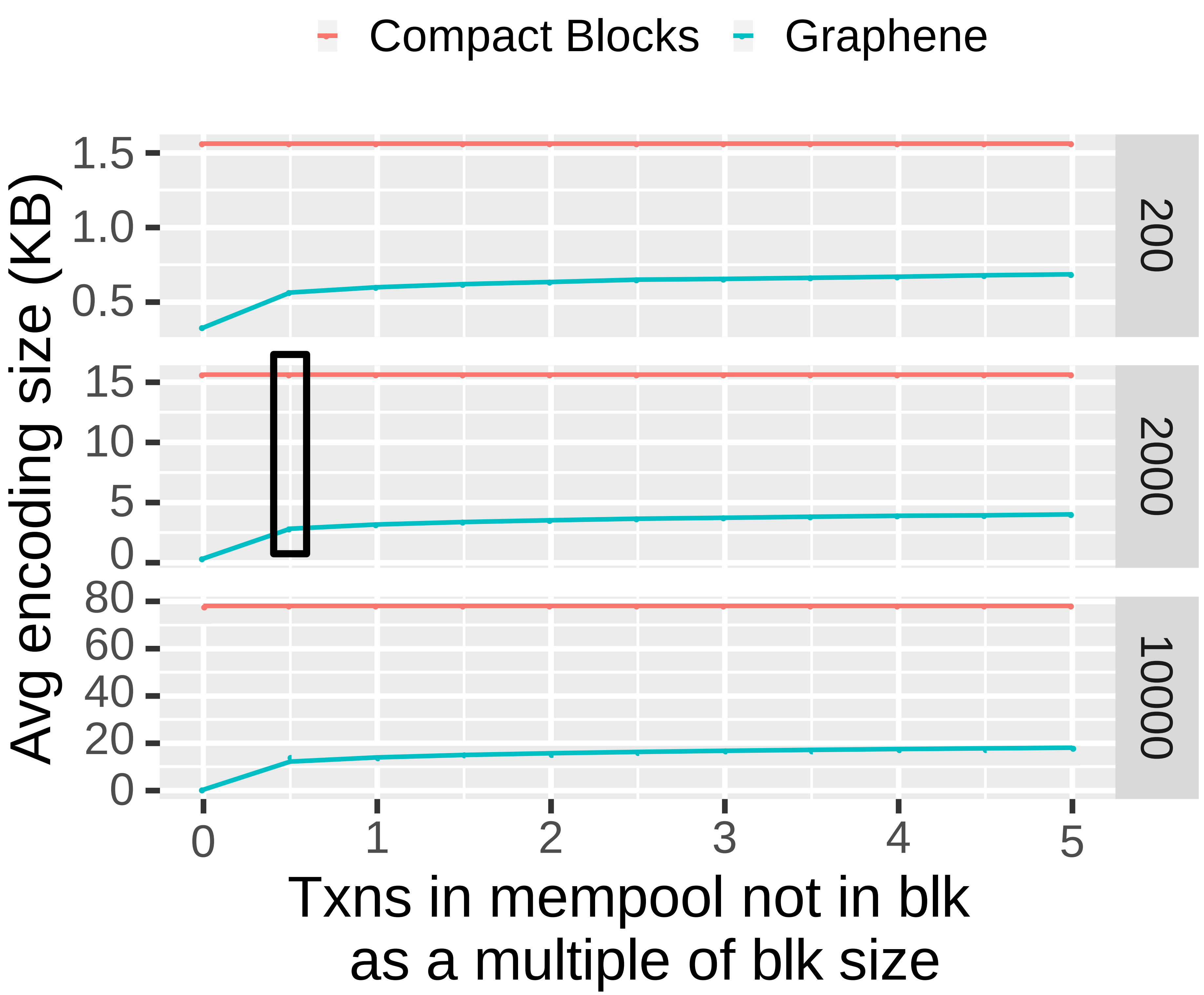
Open-Source Deployment

- Deployed on Bitcoin Cash network via the Bitcoin Unlimited client
  - 1,431 nodes
- Fraction of the size of previous work
- Deploying a protocol requires real engagement with the community
- Adversarial thinking is critical
- Mempools are in-sync less often than expected



Evaluation

- Three block sizes in terms of number of txns
- The receiver's mempool contains
  - All transactions in the block
  - Additional txns as a multiple of the block size
- Improvement with block size



Summary

Thank you!

- **SYSTEMS ISSUES**
  - Security Considerations
  - Transaction Ordering Costs
  - Reducing Processing Time

- **Limitations**

- **CONCLUSION**