Additive Combinatorics and Discrete Logarithm Based Range Protocols

Rafik Chaabouni, Helger Lipmaa, Abhi Shelat

EPFL, Cybernetica AS, University of Virginia

October 3, 2009
Outline I

1 Motivation
   - Zero-Knowledge Proofs
   - Additive Combinatorics
Zero-Knowledge Proofs

- Full security of cryptographic protocols is achieved usually by having a zero-knowledge proof (of knowledge)
Zero-Knowledge Proofs

- Full security of cryptographic protocols is achieved usually by having a zero-knowledge proof (of knowledge)
- Zero-knowledge: does not leak any extra information
Full security of cryptographic protocols is achieved usually by having a zero-knowledge proof (of knowledge)

Zero-knowledge: does not leak any extra information

Proof: the actions of any party are consistent with his committed input $Com(x)$
Full security of cryptographic protocols is achieved usually by having a zero-knowledge proof (of knowledge).

Zero-knowledge: does not leak any extra information.

Proof: the actions of any party are consistent with his committed input $\text{Com}(x)$.

We actually are interested in $\Sigma$-protocols (see the paper).
It is often sufficient to ZK-prove that committed input belongs to a correct set, e.g., is Boolean.
It is often sufficient to ZK-prove that committed input belongs to a correct set, e.g., is Boolean

Example: we are currently implementing an e-voting protocol where for correctness, it is necessary to prove that $x \in [0, H]$
It is often sufficient to ZK-prove that committed input belongs to a correct set, e.g., is Boolean.

Example: we are currently implementing an e-voting protocol where for correctness, it is necessary to prove that $x \in [0, H]$

Without such a ZK proof, the voter could induce “buffer overflow”-type errors.
It is often sufficient to ZK-prove that committed input belongs to a correct set, e.g., is Boolean

Example: we are currently implementing an e-voting protocol where for correctness, it is necessary to prove that $x \in [0, H]$

Without such a ZK proof, the voter could induce “buffer overflow”-type errors
To construct efficient ZK proofs, one needs to assume that $\text{Com}$ satisfies nice algebraic properties.
To construct efficient ZK proofs, one needs to assume that \( \text{Com} \) satisfies nice algebraic properties.

Homomorphic commitment:
\[
\text{Com}(x) \text{Com}(x') = \text{Com}(x + x')
\]
To construct efficient ZK proofs, one needs to assume that $Com$ satisfies nice algebraic properties.

Homomorphic commitment:

$$Com(x)Com(x') = Com(x + x')$$

From this trivially,

$$\prod Com(x_i)^{a_i} = Com(\sum a_ix_i)$$
To construct efficient ZK proofs, one needs to assume that $Com$ satisfies nice algebraic properties.

Homomorphic commitment:
$$Com(x)Com(x') = Com(x + x')$$

From this trivially,
$$\prod Com(x_i)^{a_i} = Com(\sum a_i x_i)$$

Example: to prove that $x \in [0, 2^\ell - 1]$, commit to bits $x_i$, then ZK-prove that $x_i \in [0, 1]$, then compute
$$Com(x) = \prod Com(x_i)^{2^i} = Com(\sum x_i 2^i)$$
Additive Combinatorics

Define $A + B := \{a + b : a \in A \land b \in B\}$
and $b \ast A = \{ba : a \in A\}$
Additive Combinatorics

- Define $A + B := \{a + b : a \in A \land b \in B\}$
- and $b \ast A = \{ba : a \in A\}$
- $A + B$ is sumset, $b \ast A$ is $b$-dilate of $A$
Additive Combinatorics

Define $A + B := \{a + b : a \in A \land b \in B\}$ and $b \ast A = \{ba : a \in A\}$

$A + B$ is sumset, $b \ast A$ is $b$-dilate of $A$

Additive combinatorics is the sexy subject that studies the properties of sumsets
Additive Combinatorics

- Define $A + B := \{a + b : a \in A \land b \in B\}$
  and $b \ast A = \{ba : a \in A\}$
- $A + B$ is sumset, $b \ast A$ is $b$-dilate of $A$
- Additive combinatorics is the sexy subject that studies the properties of sumsets
- Nobel price winners Terry Tao, Tim Gowers work on additive combinatorics, and recently Luca Trevisan and others have tried to apply additive combinatorics in theoretical computer science
Last proof works since
\[[0, 2^\ell - 1] = \sum 2^i \times [0, 1]\]
ZK-Proofs and AC

- Last proof works since
  \[ [0, 2^\ell - 1] = \sum 2^i \ast [0, 1] \]
- To prove that \( x \in \text{ValidSet} \):
ZK-Proofs and AC

- Last proof works since
  \[ [0, 2^\ell - 1] = \sum 2^i \ast [0, 1] \]
- To prove that \( x \in \text{ValidSet} \):
  - commit to some \( x_i \), then ZK-prove that \( x_i \in S_i \) for all \( i \), where \( \text{ValidSet} = \sum b_i \ast S_i \), then compute \( \text{Com}(x) = \prod \text{Com}(x_i)^{b_i} \).

Requires:
- efficient sumset-presentation
- \( \text{ValidSet} = \sum b_i \ast S_i \) — small \( n \)
- efficient ZK-proofs that \( x_i \in S_i \) — small/structured sets

Chaabouni, Lipmaa, Shelat
Additive Combinatorics and DL-Based Range Protocols
Last proof works since 
\[ [0, 2^\ell - 1] = \sum 2^i \ast [0, 1] \]

To prove that \( x \in ValidSet \):
- commit to some \( x_i \), then ZK-prove that \( x_i \in S_i \) for all \( i \), where \( ValidSet = \sum b_i \ast S_i \), then compute \( \text{Com}(x) = \prod \text{Com}(x_i)^{b_i} \)

Requires:
- efficient sumset-presentation
  \( ValidSet = \sum b_i \ast S_i \) — small \( n \)
Last proof works since 
\[ [0, 2^\ell - 1] = \sum 2^i \times [0, 1] \]

To prove that \( x \in \text{ValidSet} \):
- commit to some \( x_i \), then ZK-prove that \( x_i \in S_i \) for all \( i \), where \( \text{ValidSet} = \sum b_i \times S_i \), then compute \( \text{Com}(x) = \prod \text{Com}(x_i)^{b_i} \)

Requires:
- efficient sumset-presentation
  \( \text{ValidSet} = \sum b_i \times S_i \) — small \( n \)
- efficient ZK-proofs that \( x_i \in S_i \) — small/structured sets \( S_i \)
Range Proofs

- Range proof: ZK proof that given $c = \text{Com}(x) \land x \in [0, H]$
Range Proofs

- Range proof: ZK proof that given 
  \[ c = \text{Com}(x) \land x \in [0, H] \]

- Proof that \( x \in [L, H + L] \) can be built on this by using the homomorphic properties of \( \text{Com} \), since \( \text{Com}(x + L) = \text{Com}(x)\text{Com}(L) \)
Range Proofs

- Range proof: ZK proof that given
  \[ c = \text{Com}(x) \land x \in [0, H] \]
  - Proof that \( x \in [L, H + L] \) can be built on this by using the homomorphic properties of \( \text{Com} \), since \( \text{Com}(x + L) = \text{Com}(x)\text{Com}(L) \)

- Needed in e-voting, e-auctions and many other applications
Range Proofs: Previous Work

- Folklore: to prove $x \in [0, H]$, prove that $x \in [0, 2^\ell] \land x \in [H - 2^\ell, H]$ for $H \leq 2^\ell < 2H$

- Twice less efficient than proof that $x \in [0, 2^\ell]$

Twice less efficient than proof that $x \in [0, 2^\ell]$
Range Proofs: Previous Work

- Folklore: to prove $x \in [0, H]$, prove that $x \in [0, 2^\ell] \land x \in [H - 2^\ell, H]$ for $H \leq 2^\ell < 2H$
  - Twice less efficient than proof that $x \in [0, 2^\ell]$

- Lipmaa, Niemi, Asokan, 2002: write $[0, H] = \sum G_i \ast [0, 1]$ with $G_i := \lfloor (H + 2^i)/2^{i+1} \rfloor$
Range Proofs: Previous Work

- Folklore: to prove $x \in [0, H]$, prove that $x \in [0, 2^\ell] \land x \in [H - 2^\ell, H]$ for $H \leq 2^\ell < 2H$
  - Twice less efficient than proof that $x \in [0, 2^\ell]$

- Lipmaa, Niemi, Asokan, 2002: write $[0, H] = \sum G_i \ast [0, 1]$ with $G_i := \left\lceil (H + 2^i)/2^{i+1} \right\rceil$
  - Twice more efficient than the folklore proof

Communication complexity: $\Theta(\log H)$

Didn't use the language of additive combinatorics

Chaabouni, Lipmaa, Shelat

Additive Combinatorics and DL-Based Range Protocols
Range Proofs: Previous Work

- Folklore: to prove $x \in [0, H]$, prove that $x \in [0, 2^\ell] \land x \in [H - 2^\ell, H]$ for $H \leq 2^\ell < 2H$
  - Twice less efficient than proof that $x \in [0, 2^\ell]$

- Lipmaa, Niemi, Asokan, 2002: write $[0, H] = \sum G_i \ast [0, 1]$ with $G_i := \left\lfloor \frac{(H + 2^i)}{2^{i+1}} \right\rfloor$
  - Twice more efficient than the folklore proof
  - It’s easy to prove that $x_i \in [0, 1]$
Range Proofs: Previous Work

- Folklore: to prove $x \in [0, H]$, prove that $x \in [0, 2^\ell] \land x \in [H - 2^\ell, H]$ for $H \leq 2^\ell < 2H$
  - Twice less efficient than proof that $x \in [0, 2^\ell]$
- Lipmaa, Niemi, Asokan, 2002: write $[0, H] = \sum G_i \ast [0, 1]$ with $G_i := \lceil (H + 2^i)/2^{i+1} \rceil$
  - Twice more efficient than the folklore proof
  - It’s easy to prove that $x_i \in [0, 1]$
  - Communication complexity: $\Theta(\log H)$
Range Proofs: Previous Work

- Folklore: to prove $x \in [0, H]$, prove that $x \in [0, 2^\ell] \land x \in [H - 2^\ell, H]$ for $H \leq 2^\ell < 2H$
  - Twice less efficient than proof that $x \in [0, 2^\ell]$

- Lipmaa, Niemi, Asokan, 2002: write $[0, H] = \sum G_i * [0, 1]$ with $G_i := \left\lfloor (H + 2^i)/2^{i+1} \right\rfloor$
  - Twice more efficient than the folklore proof
  - It’s easy to prove that $x_i \in [0, 1]$
  - Communication complexity: $\Theta(\log H)$
  - Didn’t use the language of additive combinatorics
Range Proofs: Previous Work

- Camenisch, Chaabouni, Shelat 2008:

\[ 0, u^\ell - 1 \] = \sum_{i} u^i \times [0, u - 1] \]

ZK proof that \( x^i \in [0, u - 1] \) done by letting verifier to sign values 0, \ldots, \( u - 1 \), and the prover to prove that he knows signatures on all values. Uses specific signatures schemes based on bilinear pairings. By selecting optimal \( u \), the communication complexity is \( \Theta(\log H / \log \log H) \).

To prove that \( x \in [0, H] \), prove that \( x \in [0, u^\ell - 1] \land x \in [H - (u^\ell - 1), H] \) for \( H \leq u^\ell - 1 < 2H \) — twice less efficient.
Range Proofs: Previous Work

Camenisch, Chaabouni, Shelat 2008:
- Write $[0, u^\ell - 1] = \sum u^i \ast [0, u - 1]$
Range Proofs: Previous Work

Camenisch, Chaabouni, Shelat 2008:

- Write \([0, u^\ell - 1] = \sum u^i \ast [0, u - 1]\)
- ZK proof that \(x_i \in [0, u - 1]\) done by letting verifier to sign values \(0, \ldots, u - 1\), and the prover to prove that he knows signatures on all values \(x_i\)
Camenisch, Chaabouni, Shelat 2008:

- Write $[0, u^\ell - 1] = \sum u^i * [0, u - 1]$
- ZK proof that $x_i \in [0, u - 1]$ done by letting verifier to sign values $0, \ldots, u - 1$, and the prover to prove that he knows signatures on all values $x_i$
- Uses specific signatures schemes based on bilinear pairings
Range Proofs: Previous Work

Camenisch, Chaabouni, Shelat 2008:

- Write \([0, u^\ell - 1] = \sum u^i \ast [0, u - 1]\)
- ZK proof that \(x_i \in [0, u - 1]\) done by letting verifier to sign values \(0, \ldots, u - 1\), and the prover to prove that he knows signatures on all values \(x_i\)
- Uses specific signatures schemes based on bilinear pairings
- By selecting optimal \(u\), the communication complexity is \(\Theta(\log H / \log \log H)\)
Range Proofs: Previous Work

Camenisch, Chaabouni, Shelat 2008:
- Write \([0, u^\ell - 1] = \sum u^i \ast [0, u - 1]\)
- ZK proof that \(x_i \in [0, u - 1]\) done by letting verifier to sign values \(0, \ldots, u - 1\), and the prover to prove that he knows signatures on all values \(x_i\)
- Uses specific signatures schemes based on bilinear pairings
- By selecting optimal \(u\), the communication complexity is \(\Theta(\log H / \log \log H)\)

To prove that \(x \in [0, H]\), prove that
\[
x \in [0, u^\ell - 1] \land x \in [H - (u^\ell - 1), H]
\]
for \(H \leq u^\ell - 1 < 2H\) — twice less efficient
Range Proofs: Previous Work

- Camenisch, Chaabouni, Shelat 2008:
  - Write \([0, u^\ell - 1] = \sum u^i \ast [0, u - 1]\)
  - ZK proof that \(x_i \in [0, u - 1]\) done by letting verifier to sign values \(0, \ldots, u - 1\), and the prover to prove that he knows signatures on all values \(x_i\)
  - Uses specific signatures schemes based on bilinear pairings
  - By selecting optimal \(u\), the communication complexity is \(\Theta(\log H / \log \log H)\)

- To prove that \(x \in [0, H]\), prove that \(x \in [0, u^\ell - 1] \land x \in [H - (u^\ell - 1), H]\) for \(H \leq u^\ell - 1 < 2H\) — twice less efficient
Problem that We Solve

[LAN02]: \([0, H] = \sum G_i \ast [0, 1]\) with
\[G_i = \left\lfloor \frac{(H + 2^i)}{2^{i+1}} \right\rfloor\]
Problem that We Solve

- [LAN02]: $[0, H] = \sum G_i \ast [0, 1]$ with $G_i = \left\lfloor \frac{(H + 2^i)}{2^{i+1}} \right\rfloor$

- Problem: generalize [LAN02] to the case $u > 2$
Problem that We Solve

- **[LAN02]:** $[0, H] = \sum G_i \ast [0, 1]$ with $G_i = \left\lfloor \frac{(H + 2^i)}{2^{i+1}} \right\rfloor$

- Problem: generalize [LAN02] to the case $u > 2$

- Question 1: can we write $[0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u - 1]$ with some $G_i$ and small $\ell$
Problem that We Solve

[LAN02]: $[0, H] = \sum G_i \ast [0, 1]$ with $G_i = \lfloor (H + 2^i)/2^{i+1} \rfloor$

Problem: generalize [LAN02] to the case $u > 2$

Question 1: can we write $[0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u - 1]$ with some $G_i$ and small $\ell$

Question 2: If so, compute $G_i$
Problem that We Solve

- [LAN02]: $[0, H] = \sum G_i \ast [0, 1]$ with $G_i = \lfloor (H + 2^i)/2^{i+1} \rfloor$

Problem: generalize [LAN02] to the case $u > 2$

- Question 1: can we write $[0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u - 1]$ with some $G_i$ and small $\ell$

- Question 2: If so, compute $G_i$
Problem that We Solve

Question 1: can we write
\[ [0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u - 1] \] with some \( G_i \)
and small \( \ell \)
Motivation
Our Results
Previous Work
New Sumset-Representation

Problem that We Solve

Question 1: can we write
\[ [0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u - 1] \] with some \( G_i \)
and small \( \ell \)

Question 2: If so, compute \( G_i \)
Problem that We Solve

Question 1: can we write
\[ [0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u-1] \] with some \( G_i \)
and small \( \ell \)

Question 2: If so, compute \( G_i \)

Answer 1: we can write
\[ [0, H] = \sum G_i \ast [0, 1] + [0, H'] \]
\[ \ell \leq \log_u(H + 1) \text{ and } H' < u - 1 \]
Question 1: can we write
\[ [0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u - 1] \] with some \( G_i \) and small \( \ell \)

Question 2: If so, compute \( G_i \)

Answer 1: we can write
\[ [0, H] = \sum G_i \ast [0, 1] + [0, H'] \]
\[ \ell \leq \log_u(H + 1) \] and \( H' < u - 1 \)
If \( (u - 1) \mid H \) then \( H' = 0 \)
Question 1: can we write
\[ [0, H] = \sum_{i=0}^{\ell-1} G_i \ast [0, u - 1] \] with some \( G_i \) and small \( \ell \)

Question 2: If so, compute \( G_i \)

Answer 1: we can write
\[ [0, H] = \sum G_i \ast [0, 1] + [0, H'] \]
\[ \ell \leq \log_u (H + 1) \] and \( H' < u - 1 \)
If \( (u - 1) \mid H \) then \( H' = 0 \)

Answer 2: we give a semi-closed form for \( G_i \)
Basic Idea

Write $[0, H_0] = G_0 \ast [0, u - 1] + [0, H_1]$ such that $H_1$ is minimal.
Basic Idea

- Write \([0, H_0] = G_0 \ast [0, u - 1] + [0, H_1]\) such that \(H_1\) is minimal
- Equiv.: Cover \([0, H_0]\) with \(u\) intervals of size \(H_1\) that start at periodic positions \(iG_0\)
Basic Idea

- Write \([0, H_0] = G_0 \ast [0, u - 1] + [0, H_1]\) such that \(H_1\) is minimal
- Equiv.: Cover \([0, H_0]\) with \(u\) intervals of size \(H_1\) that start at periodic positions \(iG_0\)

\[ [0, 17] = 6 \ast [0, 2] + [0, 5] = 4 \ast [0, 3] + [0, 5] = 3 \ast [0, 4] + [0, 5] \]
Basic Idea

Cover $[0, H_0]$ with $u$ intervals of minimal size $H_1$ that start at periodic positions $iG_0$. 

Trivially, $H_1 \geq G_0 - 1$ and $(u - 1)G_0 + H_1 = H_0$. 
Basic Idea

- Cover \([0, H_0]\) with \(u\) intervals of minimal size \(H_1\) that start at periodic positions \(iG_0\).
- Trivially, \(H_1 \geq G_0 - 1\) and 
\((u - 1)G_0 + H_1 = H_0\)
Basic Idea

Cover $[0, H_0]$ with $u$ intervals of minimal size $H_1$ that start at periodic positions $iG_0$
Basic Idea

- Cover $[0, H_0]$ with $u$ intervals of minimal size $H_1$ that start at periodic positions $iG_0$
- Trivially, $H_1 \geq G_0 - 1$ and $(u - 1)G_0 + H_1 = H_0$
Basic Idea

- Cover \([0, H_0]\) with \(u\) intervals of minimal size \(H_1\) that start at periodic positions \(iG_0\).
- Trivially, \(H_1 \geq G_0 - 1\) and \((u - 1)G_0 + H_1 = H_0\).
- We need *minimal* \(H_1\) so set \(H_1 := G_0 - 1\).
Basic Idea

- Cover \([0, H_0]\) with \(u\) intervals of minimal size \(H_1\) that start at periodic positions \(iG_0\)
- Trivially, \(H_1 \geq G_0 - 1\) and \((u - 1)G_0 + H_1 = H_0\)
- We need minimal \(H_1\) so set \(H_1 := G_0 - 1\)
- Thus \((u - 1)G_0 + G_0 - 1 = H_0 \iff G_0 = (H_0 + 1)/u\)
Basic Idea

- Cover $[0, H_0]$ with $u$ intervals of minimal size $H_1$ that start at periodic positions $iG_0$
- Trivially, $H_1 \geq G_0 - 1$ and $(u - 1)G_0 + H_1 = H_0$
- We need minimal $H_1$ so set $H_1 := G_0 - 1$
- Thus
  $$(u - 1)G_0 + G_0 - 1 = H_0 \implies G_0 = (H_0 + 1)/u$$
- Since $G_0$ is integer, set $G_0 := \lfloor (H_0 + 1)/u \rfloor$
Basic Idea

- Cover \([0, H_0]\) with \(u\) intervals of minimal size \(H_1\) that start at periodic positions \(iG_0\)
- Trivially, \(H_1 \geq G_0 - 1\) and \((u - 1)G_0 + H_1 = H_0\)
- We need minimal \(H_1\) so set \(H_1 := G_0 - 1\)
- Thus \((u - 1)G_0 + G_0 - 1 = H_0 \implies G_0 = (H_0 + 1)/u\)
- Since \(G_0\) is integer, set \(G_0 := \lfloor (H_0 + 1)/u \rfloor\)
- Also set \(H_1 := H_0 - (u - 1)G_0\)
Basic Idea

- Cover $[0, H_0]$ with $u$ intervals of minimal size $H_1$ that start at periodic positions $iG_0$.

- Trivially, $H_1 \geq G_0 - 1$ and $(u - 1)G_0 + H_1 = H_0$.

- We need minimal $H_1$ so set $H_1 := G_0 - 1$.

- Thus
  $$(u - 1)G_0 + G_0 - 1 = H_0 \implies G_0 = (H_0 + 1)/u.$$ 

- Since $G_0$ is integer, set $G_0 := \lceil(H_0 + 1)/u \rceil$.

- Also set $H_1 := H_0 - (u - 1)G_0$.

- Optimal solution to
  $$[0, H_0] = G_0 \ast [0, u - 1] + H_1.$$
Basic Idea

We got \([0, H_0] = G_0 \ast [0, u - 1] + [0, H_1]\)
with \(H_1 < H_0\)

Chaabouni, Lipmaa, Shelat
Additive Combinatorics and DL-Based Range Protocols
Basic Idea

- We got $[0, H_0] = G_0 \ast [0, u - 1] + [0, H_1]$ with $H_1 < H_0$
- If $H_1 \geq u - 1$, then continue recursively by setting

\[
G_i := \left\lfloor \frac{(H_i + 1)}{u} \right\rfloor \\
H_{i+1} := H_i - (u - 1)G_i
\]
Basic Idea

- We got \([0, H_0] = G_0 \ast [0, u - 1] + [0, H_1]\) with \(H_1 < H_0\)
- If \(H_1 \geq u - 1\), then continue recursively by setting
  \[
  G_i := \left\lfloor \frac{(H_i + 1)}{u} \right\rfloor \\
  H_{i+1} := H_i - (u - 1)G_i
  \]

- It is easy to see that this process stops within \(\ell \leq \log_u(H + 1)\) steps
Basic Idea

- We got \([0, H_0] = G_0 * [0, u - 1] + [0, H_1]\) with \(H_1 < H_0\).
- If \(H_1 \geq u - 1\), then continue recursively by setting

\[
G_i := \left\lfloor (H_i + 1)/u \right\rfloor
\]

\[
H_{i+1} := H_i - (u - 1)G_i
\]

- It is easy to see that this process stops within \(\ell \leq \log_u(H + 1)\) steps.
- Set \(H' := H_\ell = H - \left\lfloor H/(u - 1) \right\rfloor \cdot (u - 1)\).
Theorem

\[ [0, H] = \sum_{i=0}^{\ell} G_i \ast [0, u - 1] + [0, H'] \] with \( \ell \leq \log_u (H + 1) \), \( G_i \) given by recursive formulas, and \( H' \) as in the last slide.

Optimal case: \( u \approx \log_2 H / \log_2 \log_2 H \), then the range proof has length \( \Theta(\log H / \log H \log H) \).
Semi-Closed Form for $G_i$

**Theorem**

Let $H = \sum h_i 2^i$. Then

$$G_i = \left\lfloor \frac{H}{u^i+1} \right\rfloor + \left\lfloor \frac{h_i+1+(\sum_{j=0}^{i-1} h_j \mod u-1)}{u} \right\rfloor$$

See the paper. Proof by induction, requires some case analysis.

[LAN02] result follows: there $u=2$, thus anything $\equiv 0 \mod u-1$.

Chaabouni, Lipmaa, Shelat

Additive Combinatorics and DL-Based Range Protocols
Semi-Closed Form for $G_i$

**Theorem**

Let $H = \sum h_i 2^i$. Then

$$G_i = \left\lfloor \frac{H}{u^{i+1}} \right\rfloor + \left\lfloor \frac{h_i + 1 + (\sum_{j=0}^{i-1} h_j \mod u - 1)}{u} \right\rfloor$$

See the paper. Proof by induction, requires some case analysis.
**Theorem**

Let \( H = \sum h_i 2^i \). Then

\[
G_i = \left\lfloor \frac{H}{u^i+1} \right\rfloor + \left\lfloor \frac{h_i+1 + (\sum_{j=0}^{i-1} h_j \mod u-1)}{u} \right\rfloor
\]

See the paper. Proof by induction, requires some case analysis.

[LAN02] result follows: there \( u = 2 \), thus anything \( \equiv 0 \mod u - 1 \)
More Details

- ZK-proof follows [CCS08], but uses the new sumset-representation of $[0, H]$. 

Additional optimization:
Recall that if $(u - 1) \mid H$ then $H' = 0$.

Instead of $x \in [0, H]$ we prove that $(u - 1) x \in [0, (u - 1) H]$. 

Range proof twice more efficient than [CCS08] for general $H$. 

More Details

- ZK-proof follows [CCS08], but uses the new sumset-representation of $[0, H]$
- Additional optimization:
ZK-proof follows [CCS08], but uses the new sumset-representation of $[0, H]$

Additional optimization:
- Recall that if $(u - 1) \mid H$ then $H' = 0$
More Details

- ZK-proof follows [CCS08], but uses the new sumset-representation of \([0, H]\)
- Additional optimization:
  - Recall that if \((u - 1) \mid H\) then \(H' = 0\)
  - Instead of \(x \in [0, H]\) we prove that \((u - 1)x \in [0, (u - 1)H]\)
ZK-proof follows [CCS08], but uses the new sumset-representation of \([0, H]\)

Additional optimization:
- Recall that if \((u - 1) \mid H\) then \(H' = 0\)
- Instead of \(x \in [0, H]\) we prove that \((u - 1)x \in [0, (u - 1)H]\)

Range proof twice more efficient than [CCS08] for general \(H\)
Questions?

- Our contribution: cryptographic problem solved by reformulating a problem in the language of additive combinatorics, but solving it by a new (independent) technique.
Our contribution: cryptographic problem solved by reformulating a problem in the language of additive combinatorics, but solving it by a new (independent) technique.

Question: Can you use existing techniques from AC?
Questions?

- Our contribution: cryptographic problem solved by reformulating a problem in the language of additive combinatorics, but solving it by a new (independent) technique.

- Question: Can you use existing techniques from AC?

- Open question: devise an “efficient” sumset-representation for a large family of sets $A$. 