Self-Identifying Sensor Data

Jeffrey Vaughan

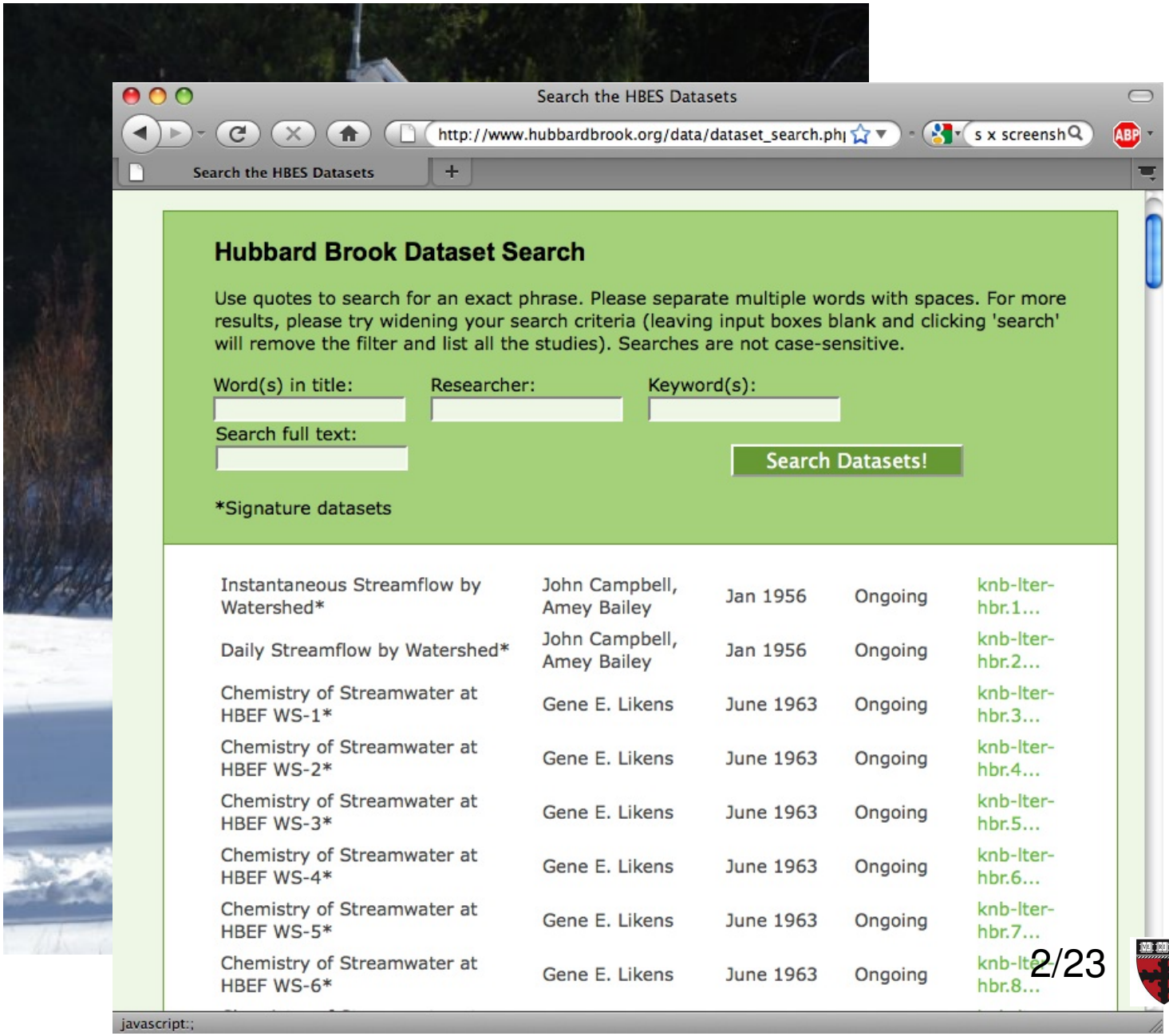
Department of Computer Science
Harvard University

With Stephen Chong (Harvard) and Christian Skalka (UVM)

IPSN
April 13, 2010

Example: Hubbard Brook Ecosystem Study

Example: Hubbard Brook Ecosystem Study



Data sharing: a tension

- Scientists want their work to be used and cited.
- But don't want use without attribution.

Goal

Track data provenance in a light-weight, cooperative way.

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Snow (µm)
312344
23818
123275
435493

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (µm)
312344
23818
123275
435493

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (μm)
312185
23338
123425
435215

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (μm)

312185
23338
123425
435215

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (μm)		Snow (μm)
312185	Publish →	312185
23338		23338
123425		123425
435215		435215

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (µm)		Snow (µm)
312185	Publish →	312185
23338		~~23338~~
123425		123425
435215		435215

Sample

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (µm)		Snow (µm)
312185	Publish →	312185
23338		
123425		123425
435215		435215

Sample

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (μm)		Snow (μm)
312185	Publish →	312190
23338		
123425		123430
435215		435220

Sample

Round

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (μm)		Snow (μm)
312185	Publish →	312190
23338		435220
123425		123430
435215		

Sample

Reorder

Round

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (μm)		Snow (μm)
312185	Publish →	312190
23338		435220
123425		123430
435215		

Key Idea: Encode provenance in *insignificant bits*.

Sensor
w/ error
±1mm

Provenace
Mark: 54321

Snow (µm)

312185
23338
123425
435215

Publish →

Snow (µm)

312190
435220
123430

Recover ↓

Provenace
Mark: 54321

Key Idea: Encode provenance in *insignificant bits*.

Provenace
Mark: 54321

Sensor
w/ error
±1mm

Snow (µm)
312185
23338
123425
435215

Publish →

Snow (µm)
312190
435220
123430

Recover ↓

Mark 54321: Hubbard Brook Ecosystem Study Snowfall Experiment

Done

← Lookup

Provenace
Mark: 54321

The scientific/educational threat model is unusual.

Scientists. . .

- are cooperative: they want to properly cite and attribute data.
- use legacy workflows, and will be harmed by schema/format changes.
- care a lot about data integrity, but understand the bounds on measurement precision and accuracy.

Outline

Encoding and Decoding

Self-identifying data based on three functions

- $\mathit{Encode} : \mathit{Dataset} \times \mathit{Mark} \rightarrow \mathit{AnnotatedDataset}$
- $\mathit{Check} : \mathit{Dataset} \times \mathit{Mark} \rightarrow \mathit{bool}$
- $\mathit{Recover} : \mathit{Dataset} \rightarrow \mathit{Mark}$

Self-identifying data based on three functions

- $Encode : Dataset \times Mark \rightarrow AnnotatedDataset$
- $Check : Dataset \times Mark \rightarrow bool$ (cf. *One-bit* watermarking)
- $Recover : Dataset \rightarrow Mark$ (cf. *Blind* watermarking)

Where

$$\begin{aligned} Mark &\triangleq int32 \\ Datapoint &\triangleq int \\ Dataset &\triangleq AnnotatedDataset \triangleq List\ of\ Datapoint \end{aligned}$$

Data is collected for scientific analysis.

Sensors measure real-world phenomena.

- High-order *significant bits* contain meaningful data.
- Low-order *insignificant bits* contain noise.
- Acceptable encodings should be minimally *perceptible*.

Guiding principle

Never alter significant bits.

Encoding must be robust against transformation.

Sampling Store metadata in many points: *Pointwise redundancy.*

Truncation

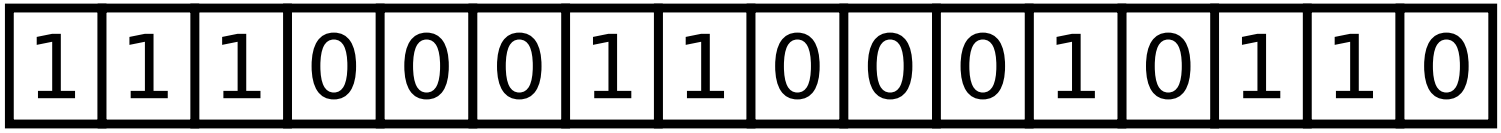
- Store key metadata in more significant bits.
- Other metadata stored in several bit locations: *Positional redundancy*

Reordering Order independent encoding.

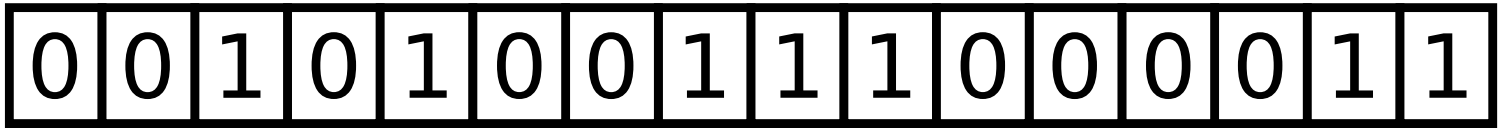
Insertions/Bit Flips Enable recovery even with “bad” datapoints.

Encoding must use limited supply of insignificant bits

Datapoint



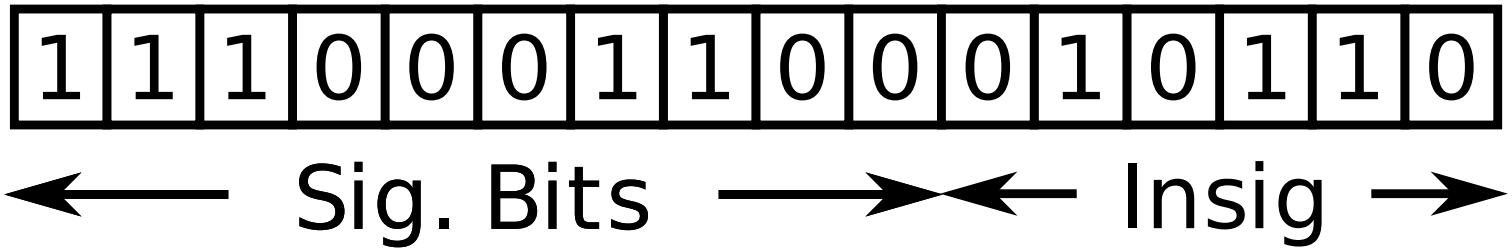
Mark



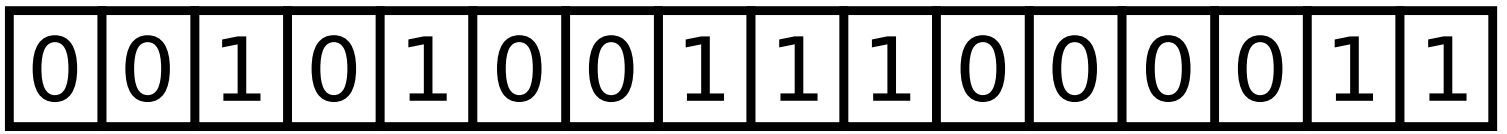
Encoding must use limited supply of insignificant bits

$L_{md} = 6$

Datapoint



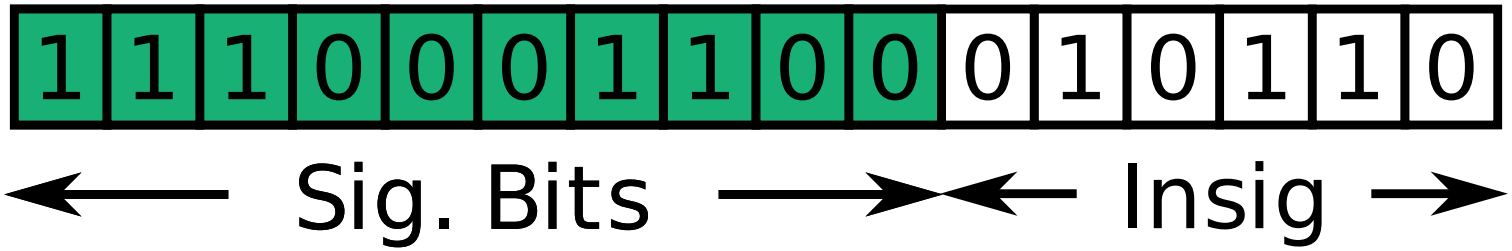
Mark



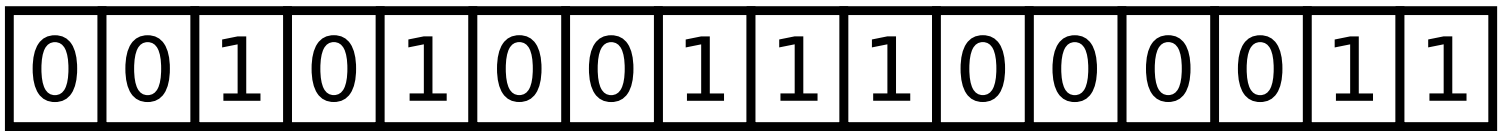
Encoding must use limited supply of insignificant bits

$L_{md} = 6$

Datapoint



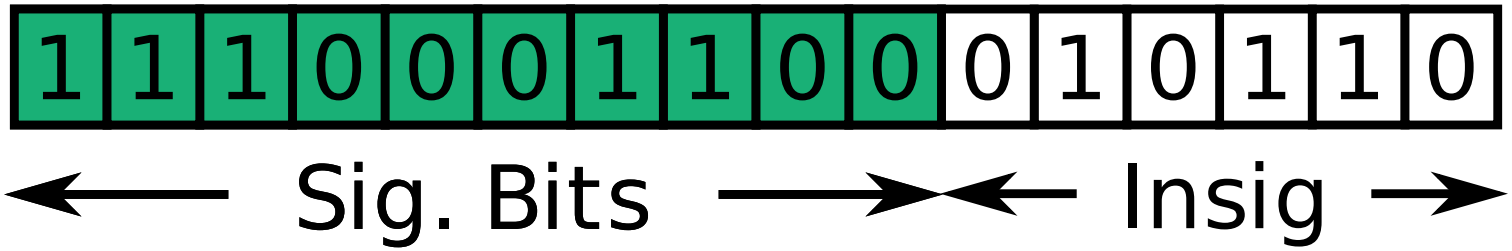
Mark



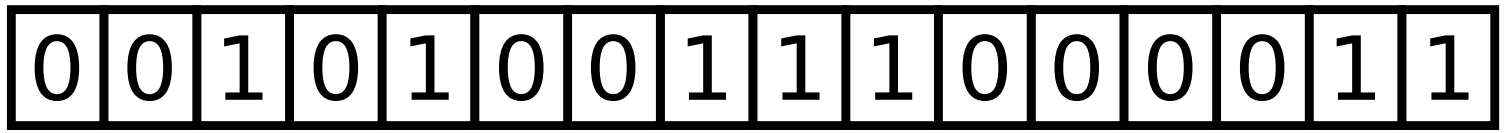
Encoding must use limited supply of insignificant bits

$L_{md} = 6$

Datapoint



Mark

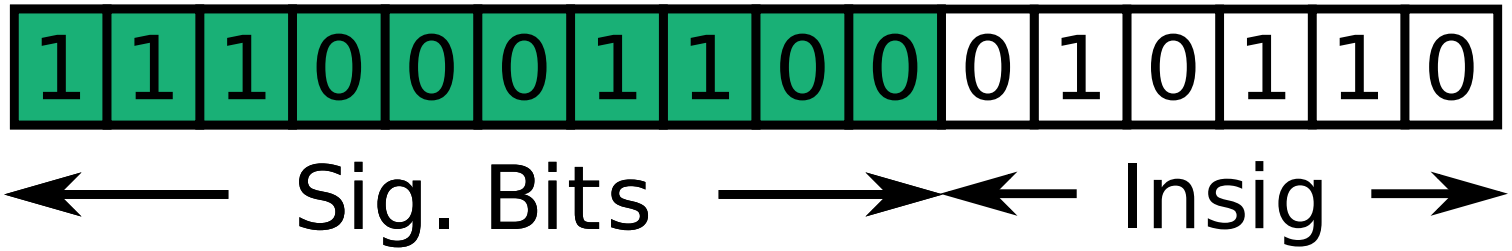


Number of Provenance Pieces: $N_{pp} = 4$

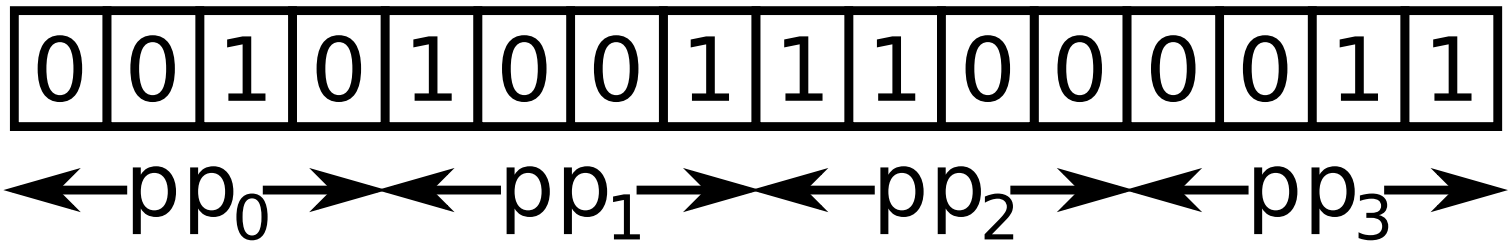
Encoding must use limited supply of insignificant bits

$L_{md} = 6$

Datapoint



Mark

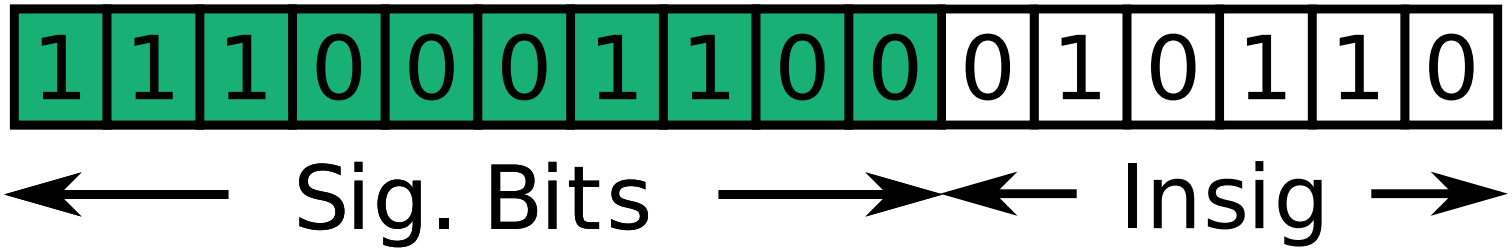


Number of Provenance Pieces: $N_{pp} = 4$

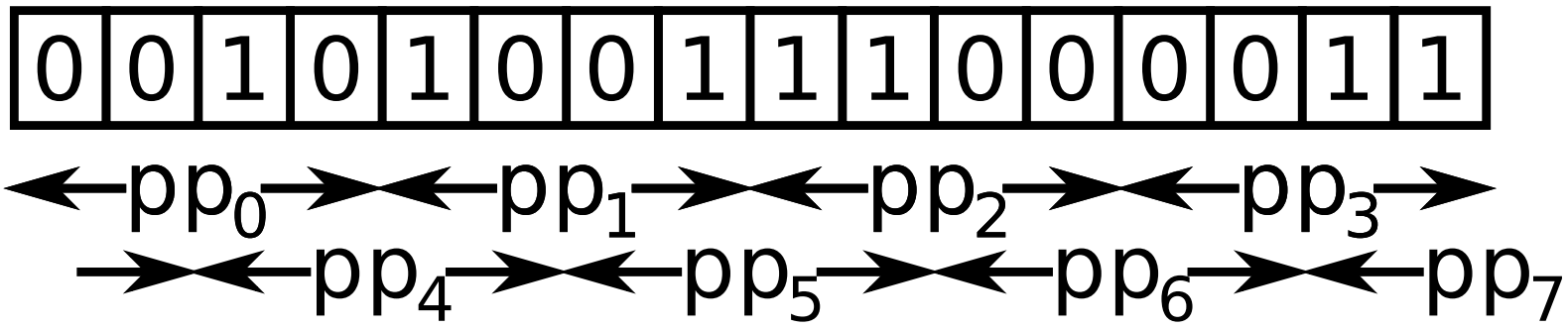
Encoding must use limited supply of insignificant bits

$L_{md} = 6$

Datapoint



Mark

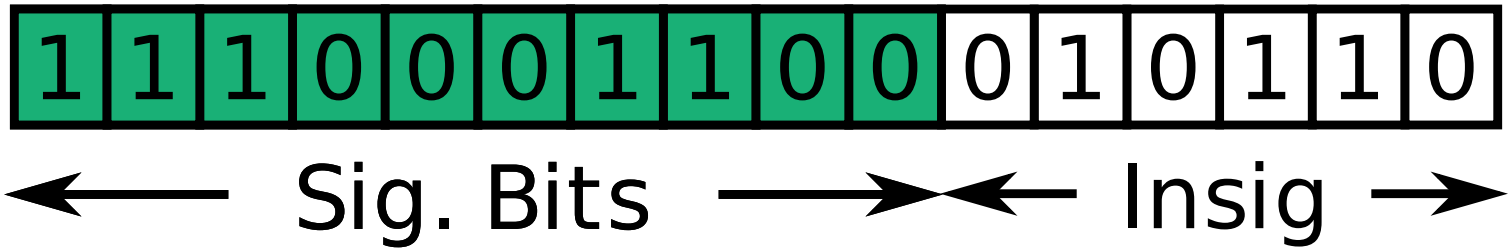


Number of Provenance Pieces: $N_{pp} = 8$

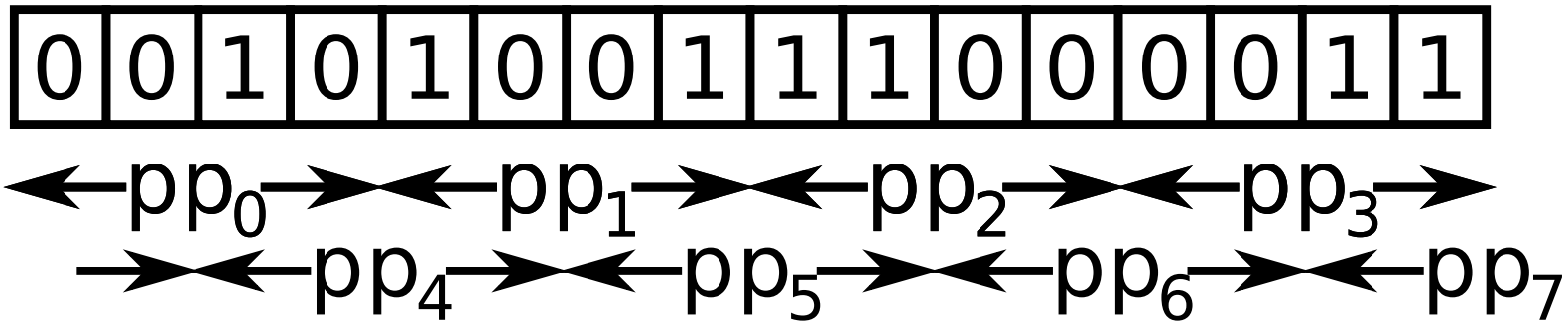
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint

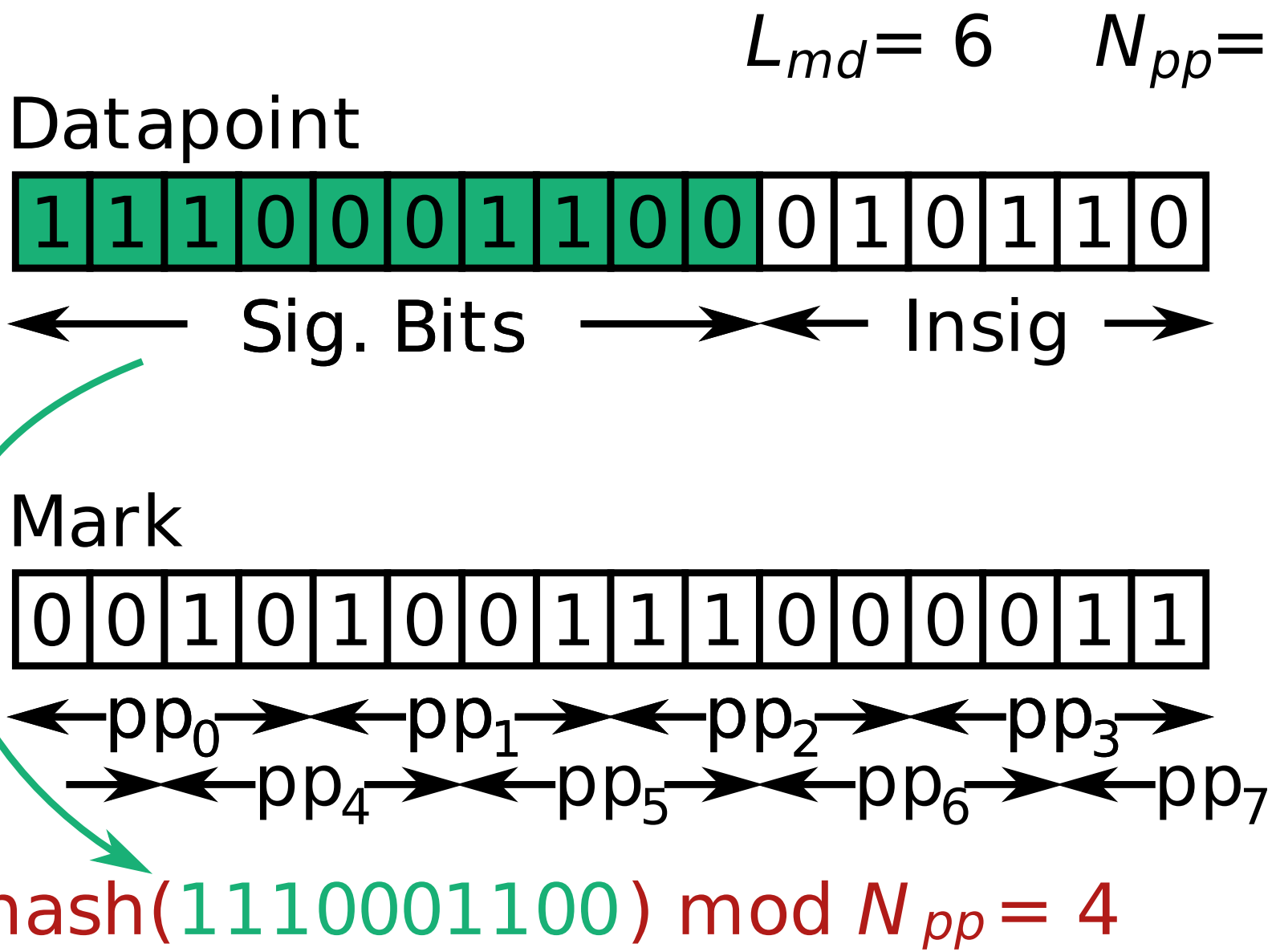


Mark



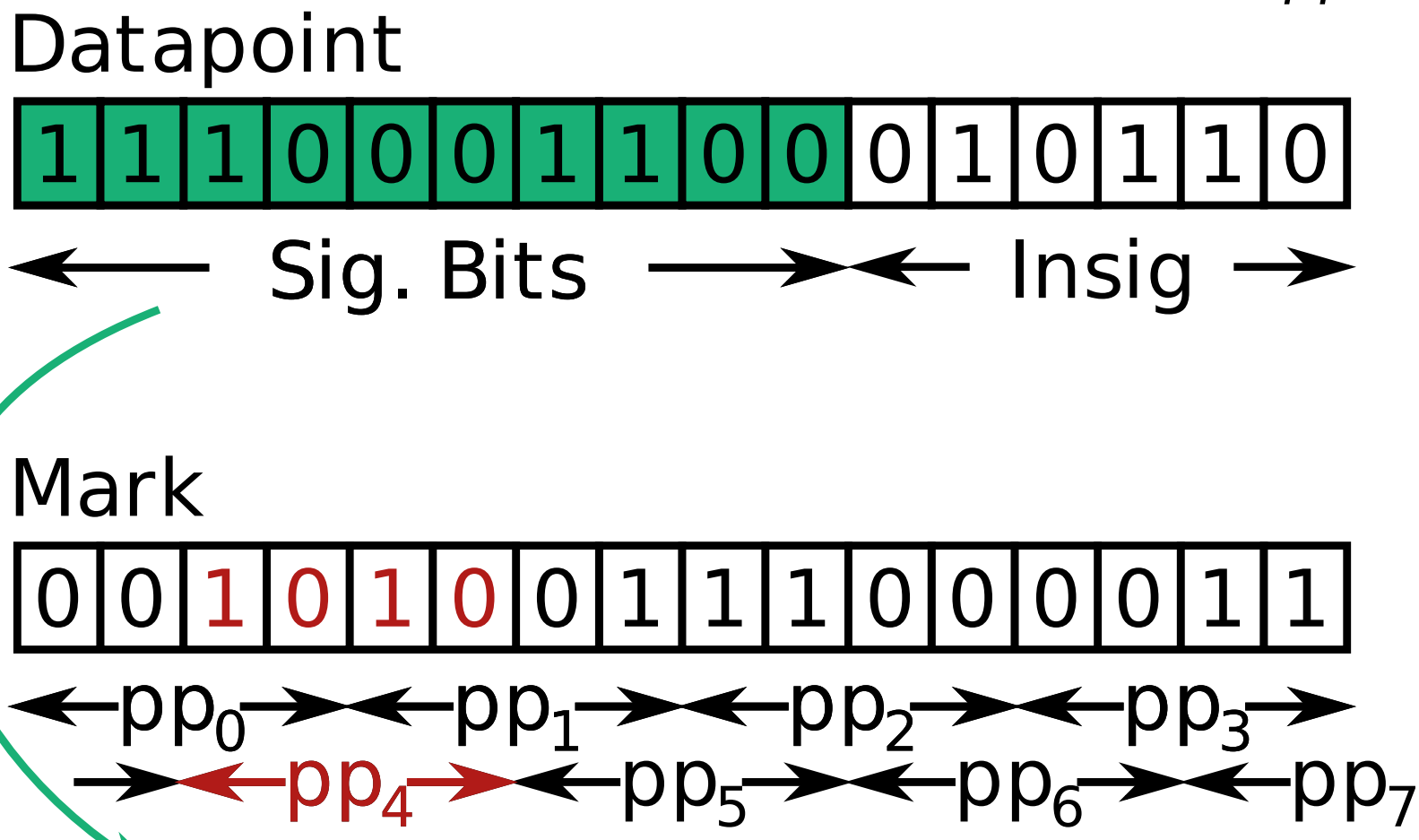
Number of Provenance Pieces: $N_{pp} = 8$

Encoding must use limited supply of insignificant bits

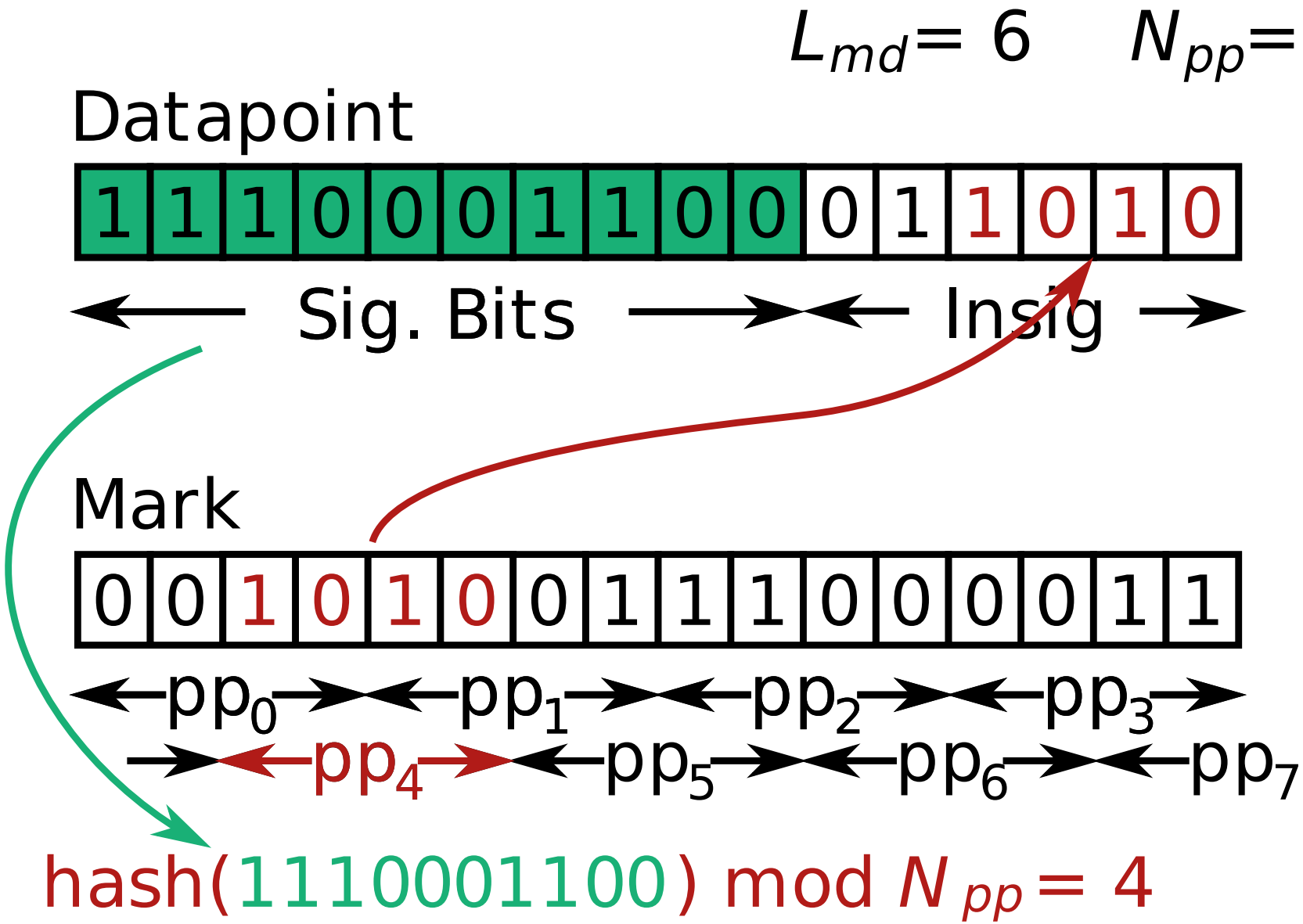


Encoding must use limited supply of insignificant bits

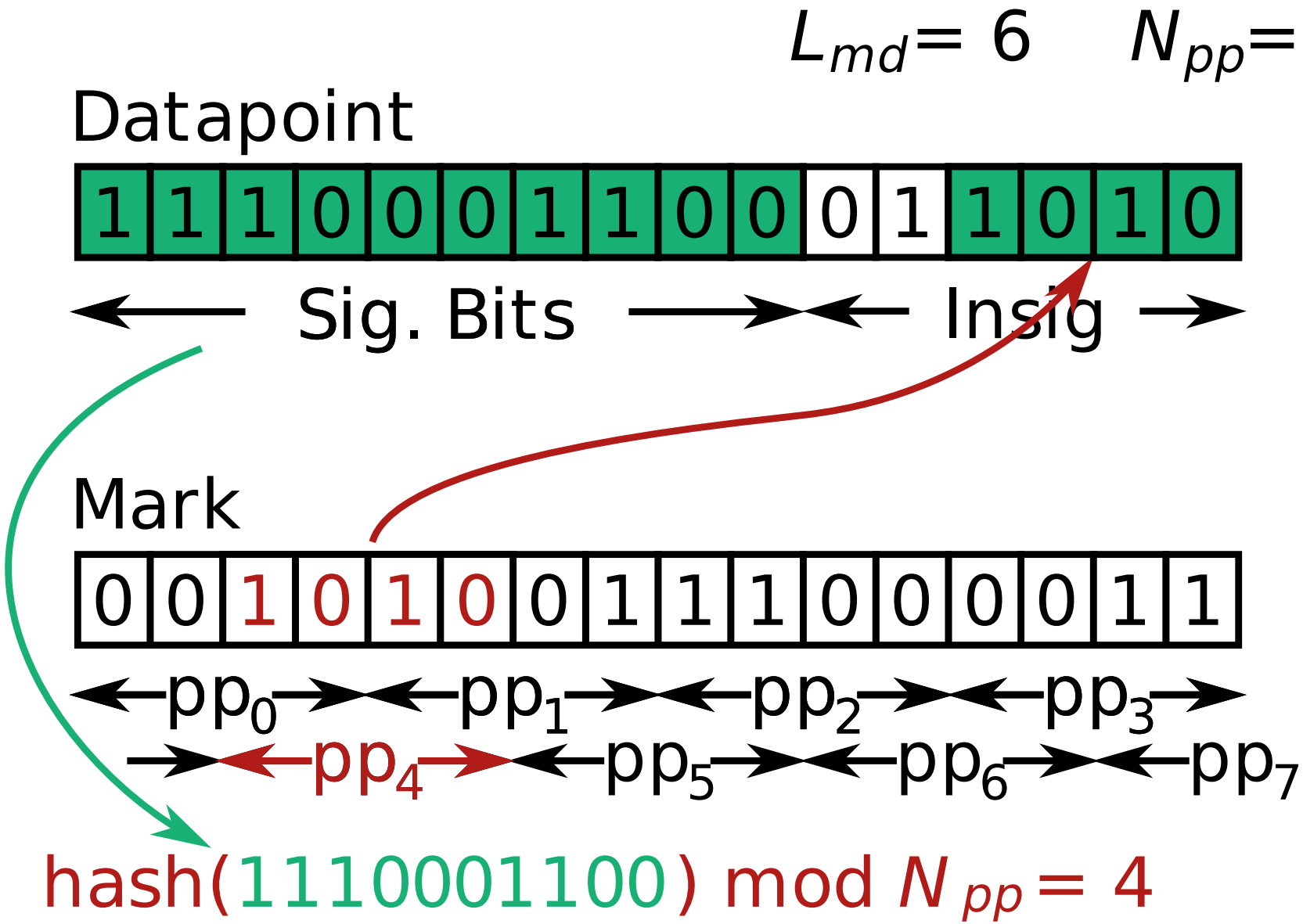
$L_{md} = 6$ $N_{pp} = 8$



Encoding must use limited supply of insignificant bits



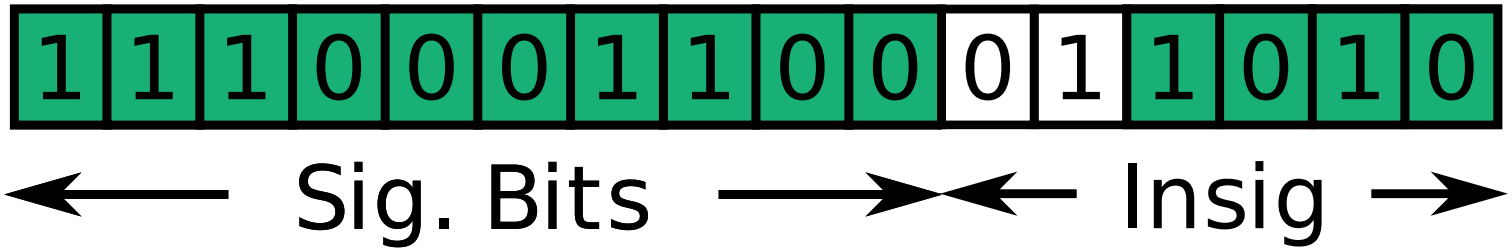
Encoding must use limited supply of insignificant bits



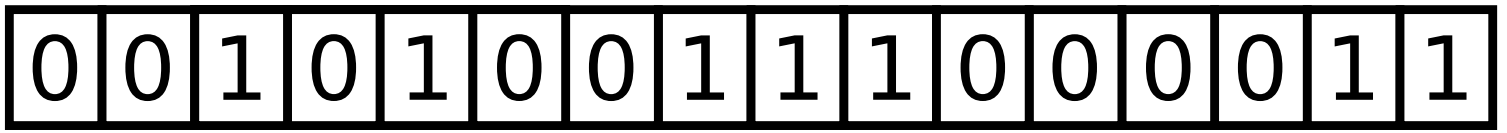
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



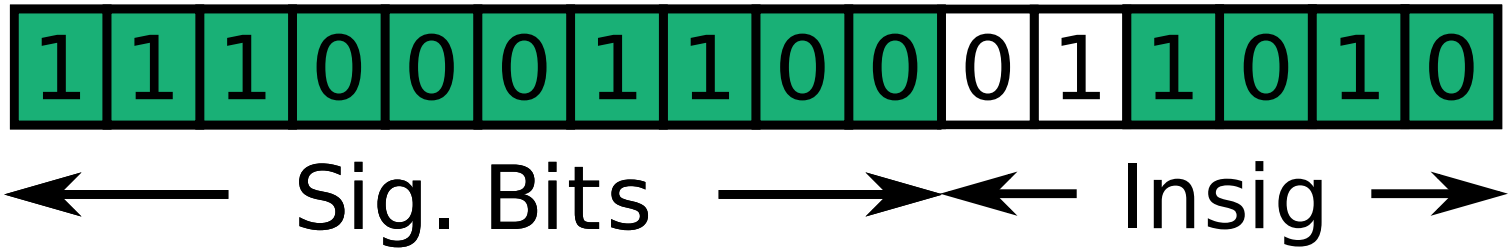
Mark



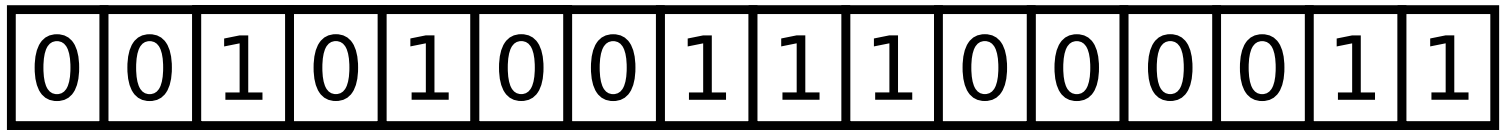
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



Mark



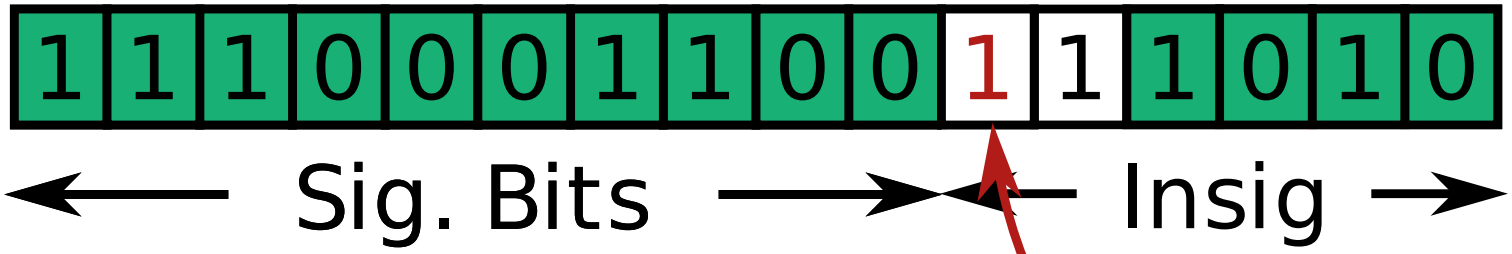
Parameter check bit:

pc = hash(Sig. Bits@N_{pp}) mod 2 = 1

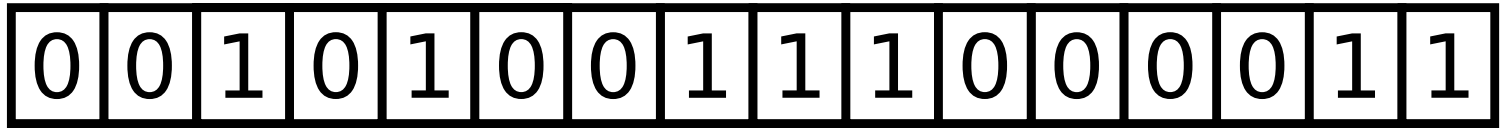
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



Mark



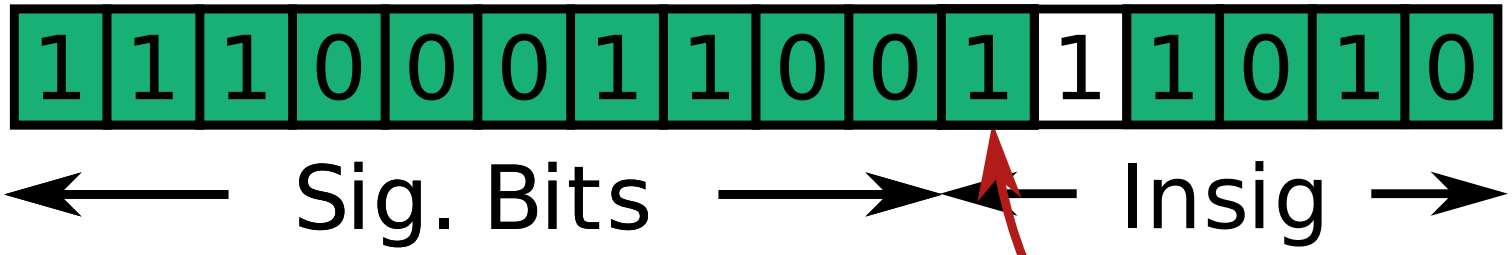
Parameter check bit:

pc = hash(Sig. Bits@N_{pp}) mod 2 = 1

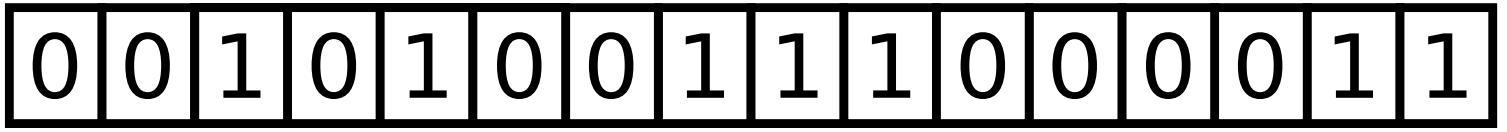
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



Mark



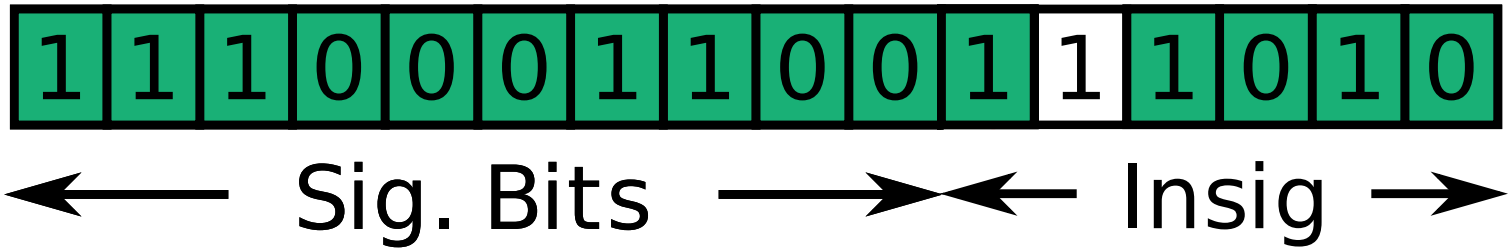
Parameter check bit:

pc = hash(Sig. Bits@N_{pp}) mod 2 = 1

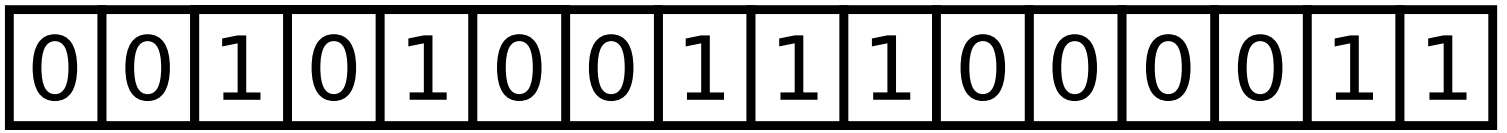
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



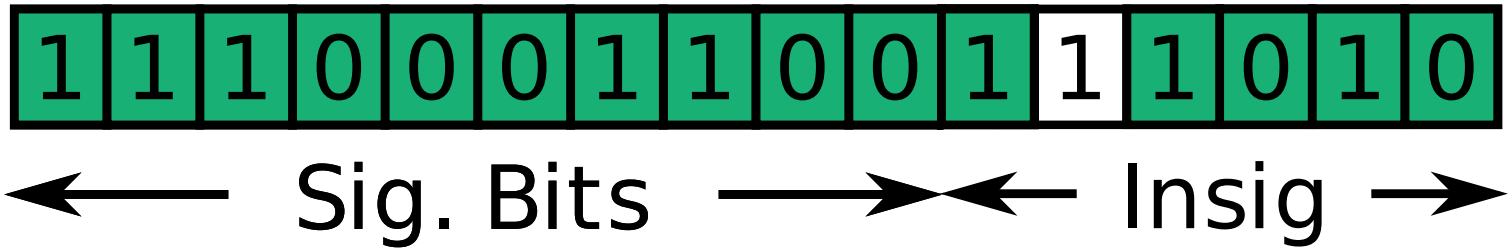
Mark



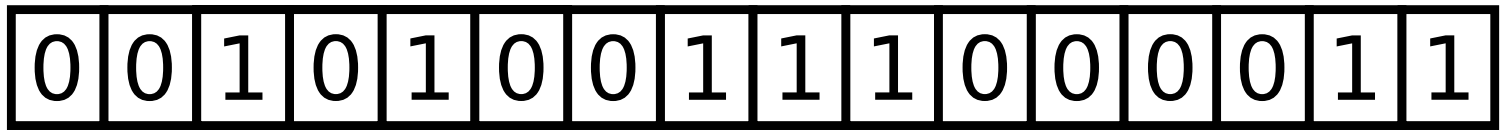
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



Mark



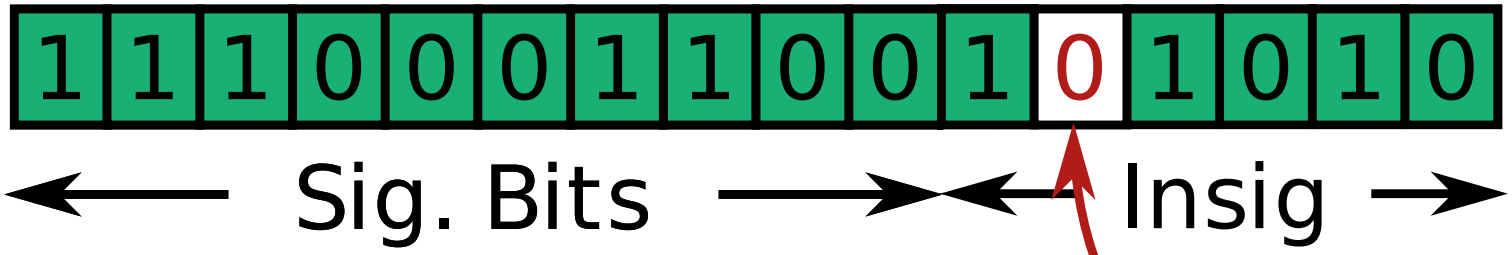
Mark check bit:

mc = hash(Sig. Bits@Mark) mod 2 = 0

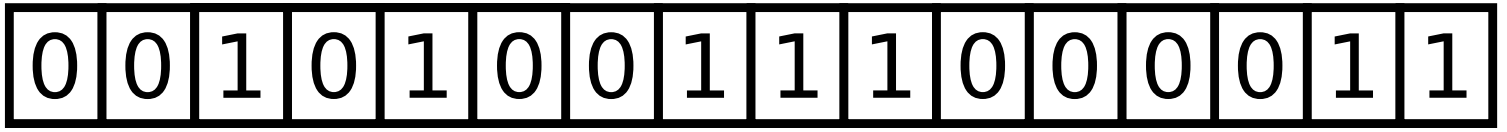
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



Mark



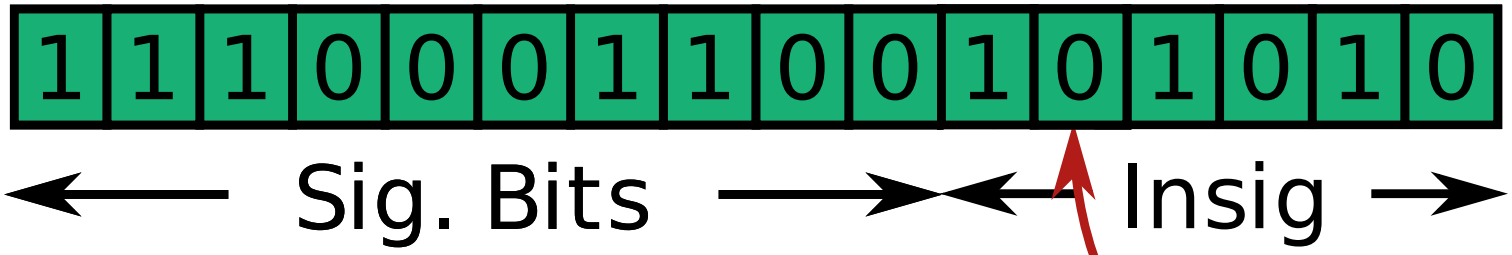
Mark check bit:

mc = hash(Sig. Bits@Mark) mod 2 = 0

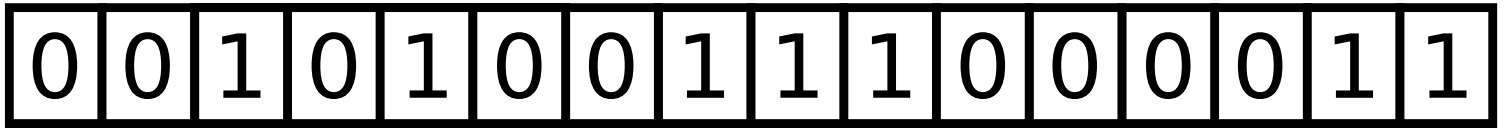
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



Mark



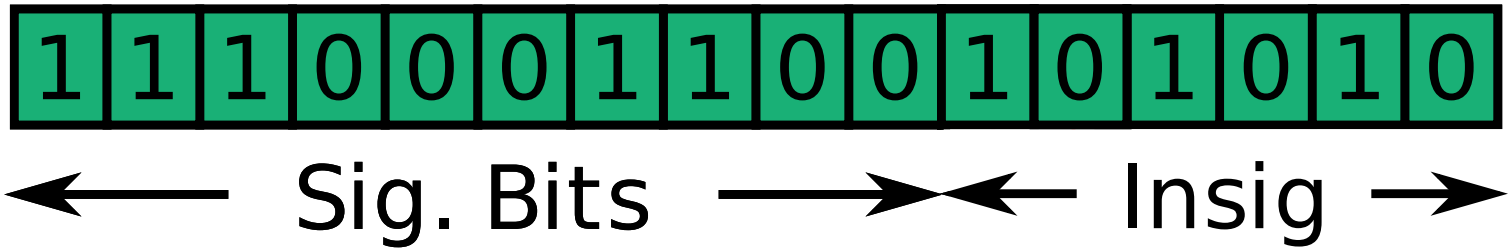
Mark check bit:

mc = hash(Sig. Bits@Mark) mod 2 = 0

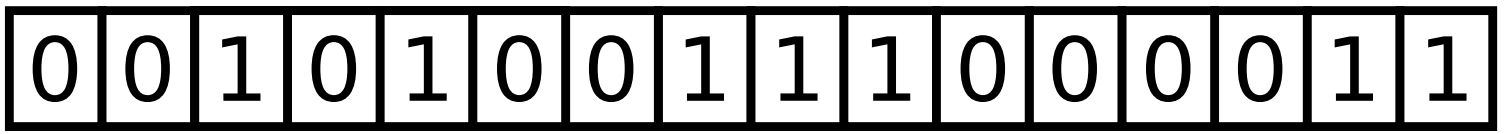
Encoding must use limited supply of insignificant bits

$L_{md} = 6$ $N_{pp} = 8$

Datapoint



Mark



Decoding uses all metadata from annotated datasets.

parameter check: recovery of encoding parameters

mark check: checking if a mark is consistent with dataset

provenance piece: recover an unknown mark from a dataset

Parameter check bit allows for parameter recovery.

Goal

Let D be an annotated data set with k elements. Find L_{md} and N_{pp} used in encoding.

Parameter check bit allows for parameter recovery.

Goal

Let D be an annotated data set with k elements. Find L_{md} and N_{pp} used in encoding.

Algorithm sketch (*FindParams*)

Enumerate L_{md} and N_{pp} candidates, rejecting incorrect guesses.
(200 or fewer candidates in the worst case.)
Calculate *pc* for each guess and compare with data points in D.

- Correct guess: Probability *pc*-consistent $= 1$.
- Incorrect guess:
  - Probability consistent with any data point $\approx 1/2$.
  - Probability consistent with all points $\approx (1/2)^k$.

Define *Check* function using mark check bit.

Goal

$$Check(D, m) = true$$

iff mark m is encoded into dataset D.

Define *Check* function using mark check bit.

Goal

$$Check(D, m) = true$$

iff mark m is encoded into dataset D.

Algorithm sketch (*Check*)

Recover parameter L_{md} (and N_{pp}) using *FindParams*.

For each data point in D, recompute *mc*.

- Correct guess: Probability *mc*-consistent $= 1$.
- Incorrect guess:
  - Probability consistent with any data point $\approx 1/2$.
  - Probability consistent with all points $\approx (1/2)^k$.

Recover function based on *Check*.

Goal

$$\mathit{Recover}(D) = m$$

iff mark m is encoded into dataset D.

Recover function based on *Check*.

Goal

$$Recover(D) = m$$

iff mark m is encoded into dataset D.

Algorithm sketch (*Recover*)

Let $suggestions = GatherSuggestions(D)$
Let $candidates : List\ mark = Merge(suggestions)$

for each $m \in candidates$
    if $Check(D, m)$
        return m

Recover function based on *Check*.

Goal

$$Recover(D) = m$$

iff mark m is encoded into dataset D.

Algorithm sketch (*Recover*)

Let $suggestions = GatherSuggestions(D)$
Let $candidates : List\ mark = Merge(suggestions)$

for each $m \in candidates$
    if $Check(D, m)$
        return m

Heuristics in *Merge* keep *candidates* list short, and search tractable.

Probabilistic algorithms deal with corruption.

Key Ideas

- Empirical cutoffs < 1 determine when *mc*- and *pc*-consistency scores indicate correct parameters or marks.
- Weighted voting selects likely candidates provenance marks.

See paper for details.

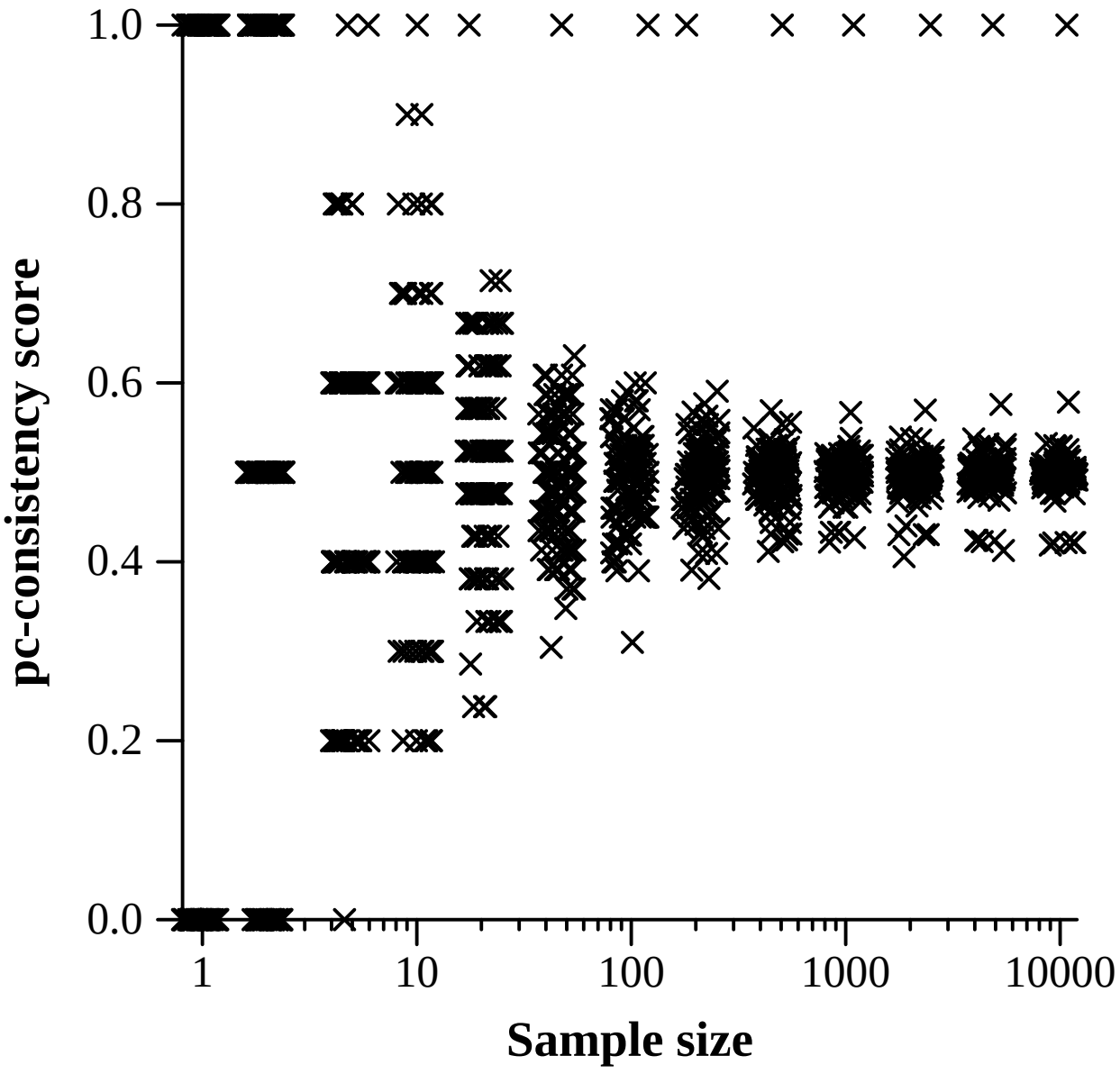
Evaluation

Encoding preserves bulk statistical data properties.

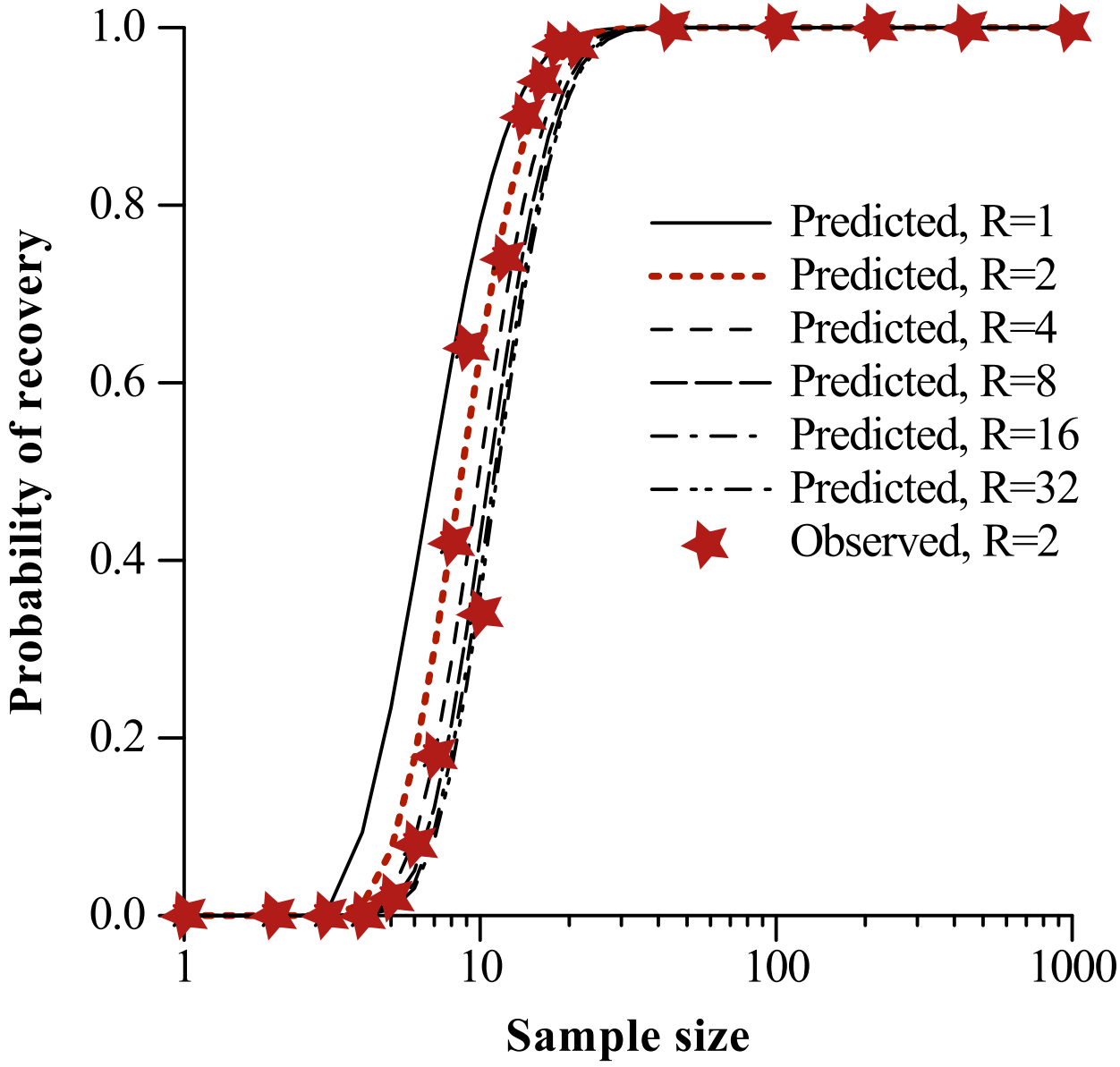
- Assumption: hash mod k samples uniform distribution over $\{0 \ldots k-1\}$.
- Let $\epsilon = 2^{L_{md}}$, the largest value written with insignificant bits.

	Change in mean	Change in variance
worst case	$\epsilon - 1$	$(\epsilon - 1)^2/4$
insignificant bits uniformly dist.	~ 0	0

Recovery is robust against sampling. (1/2)



Recovery is robust against sampling. (2/2)



Conclusion

Self-identifying data solves a new problem.

This work: a mechanism for embedding provenance information into sensor data.

Management of provenance metadata:

- Metadata generation [SensorBase]
- Republication and curation [Park and Heidermann; Ledlie, Ng, et al. ’05]

Structured data:

- Databases [Lee, Bressen, and Madnick ’95; Buneman, Chapmen and Cheney ’06]
- Video and images [Genani and Lindqvist, ’07]

Non-malleable data:

- Digital signatures [Goldwasser, Micali, and Rivest ’88]

Take home message

Self identifying data. . .

- associates sensor data with provenance metadata.
- maintains integrity of significant bits, means, and variance.
- is robust against benign transformations.

Bonus Slides

- Definition of *GatherSuggestions*

Definition of *GatherSuggestions*

Algorithm sketch(*GatherSuggestions*(D))

Recover L_{md} and N_{pp} using *FindParams*.
Let *suggestions* $:= \emptyset$

For each $d \in D$
  Let $(S, pc, mc, pp) = split(d, L_{md})$
  Let $offset = hash(S) \mod N_{pp}$
  Let

$$suggest(i) = \begin{cases} (pp[j], j) & j = i - offset \mod N_{pp} \\ & \quad \text{and } pp[j] \text{ in bounds} \\ * & \text{otherwise} \end{cases}$$

  $suggestions := suggestions \cup \{suggest\}$

return *suggestions*