Toward Nearly-Zero-Error Sketching via Compressive Sensing

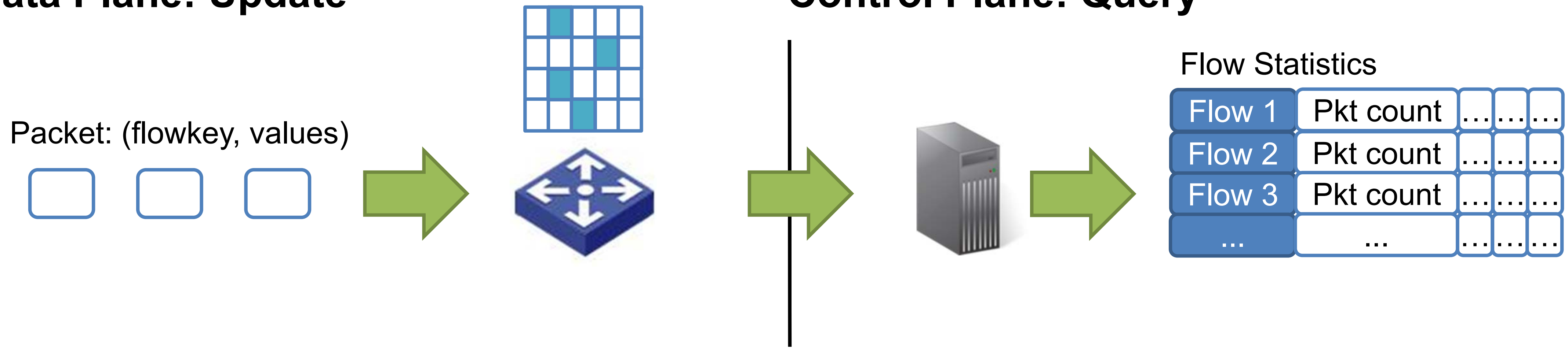
Qun Huang, Siyuan Sheng, Xiang Cheng,

Yungang Bao, Rui Zhang, Yanwei Xu, Gong Zhang

Flow-level Network Monitoring

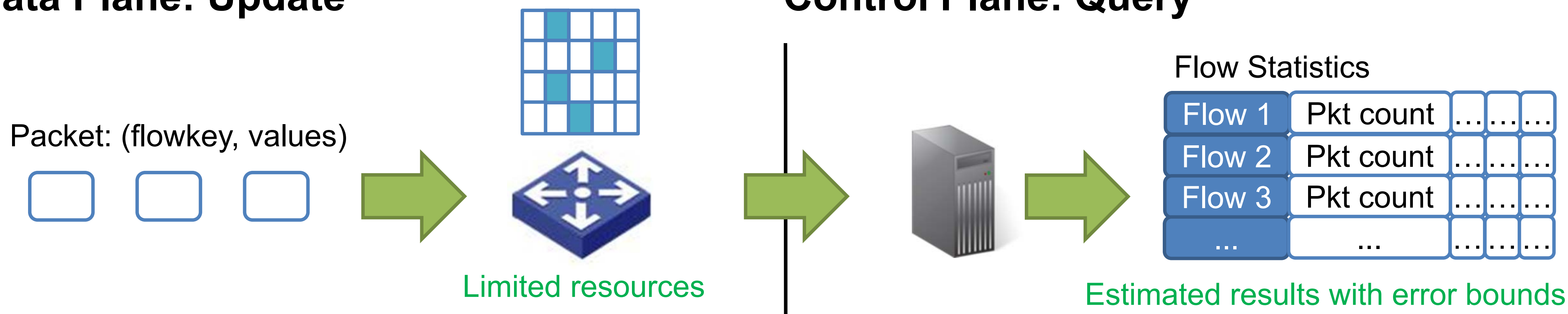
Data Plane: Update

Control Plane: Query



Flow-level Network Monitoring

Data Plane: Update **Control Plane: Query**



Approximate techniques are widely used

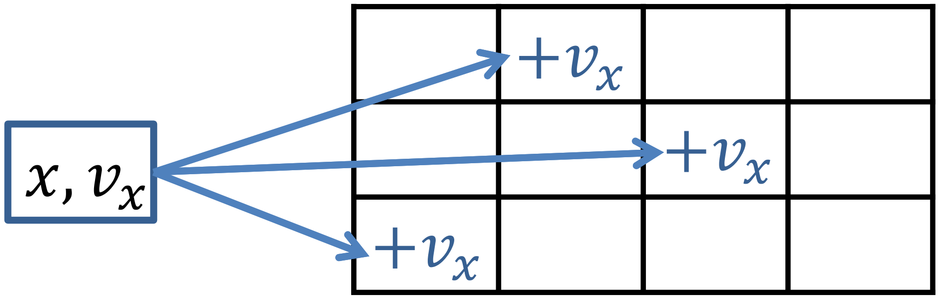
Existing Guarantees Are Not Enough

- Theoretical bounds only apply to specific flows
  - E.g., heavy hitters, super-spreaders, …

- For most flows, the bounds are too loose

Example

Count-Min Sketch to monitor byte count



Configuration
- 10GB network traffic
- 10^4 counters per-row
- >10 rows

Error bound: Per-flow error <210KB

<2% relative error for large flows (>10MB) ☺

Unacceptable for small flows ☹

Small flows matter:
E.g., single-packet TCP flows imply various anomalies

Our Contributions

New algorithms that achieve nearly-zero-error monitoring

Nearly-zero error: for almost all (>99%) flows, the relative error is small (<0.1%)

Sketch

Compressive Sensing

Our Contributions

New algorithms that achieve nearly-zero-error monitoring

Nearly-zero error: for almost all (>99%) flows, the relative error is small (<0.1%)

1. Advantages and limitations of compressive sensing for network monitoring

2. Efficiency of combining sketch and compressive sensing

Sketch

+

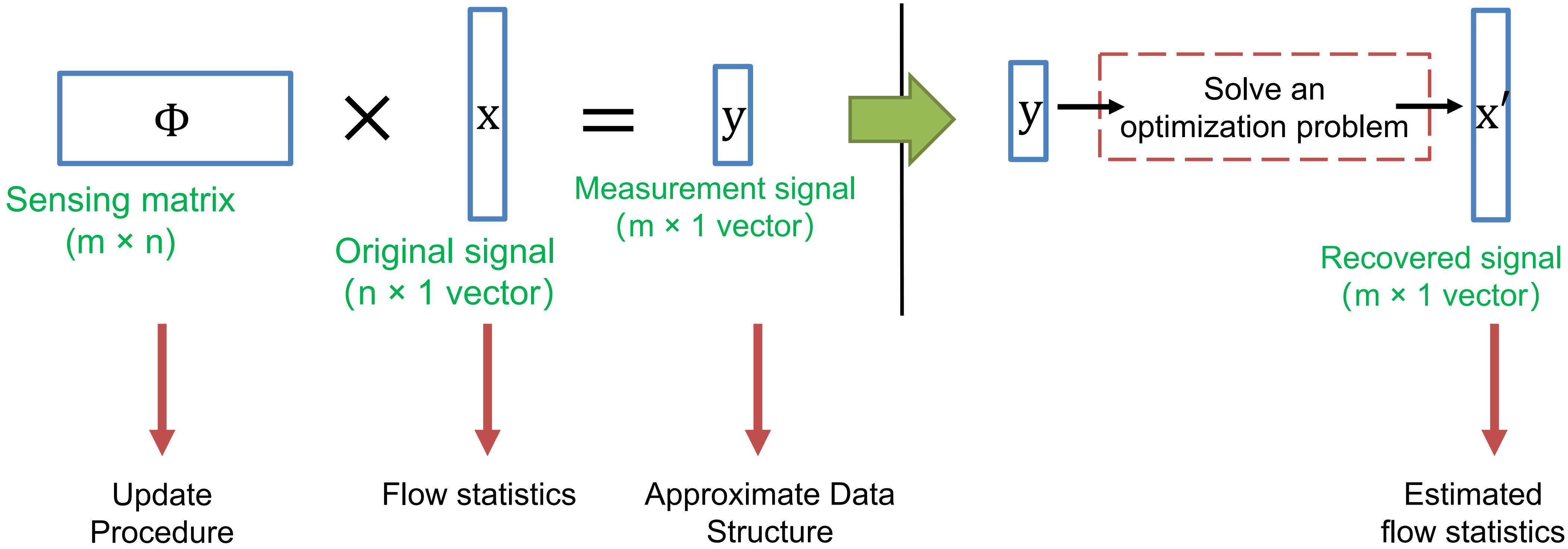
Compressive Sensing

3. A theoretical framework to revisit common approaches in sketch design

4. Two new sketch algorithms that efficiently embrace compressive sensing

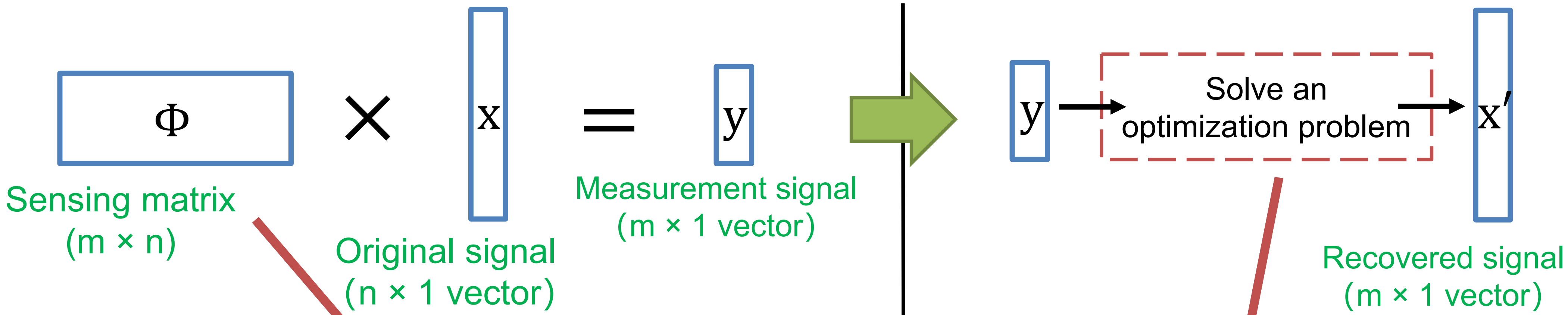
Compressive Sensing

Sensing Operation **Recovery Operation**



Compressive Sensing

Sensing Operation **Recovery Operation**

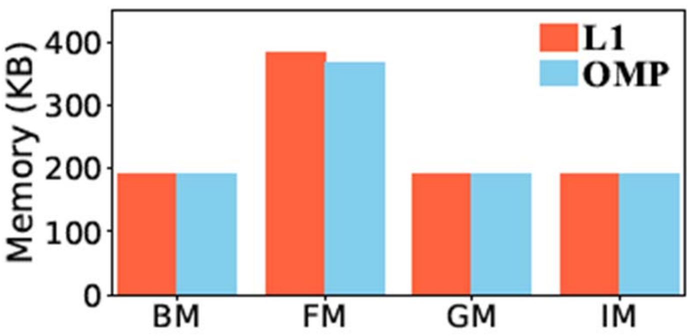


Classical Compressive Sensing:
Guarantee accuracy with recommended
Φ and optimization algorithm

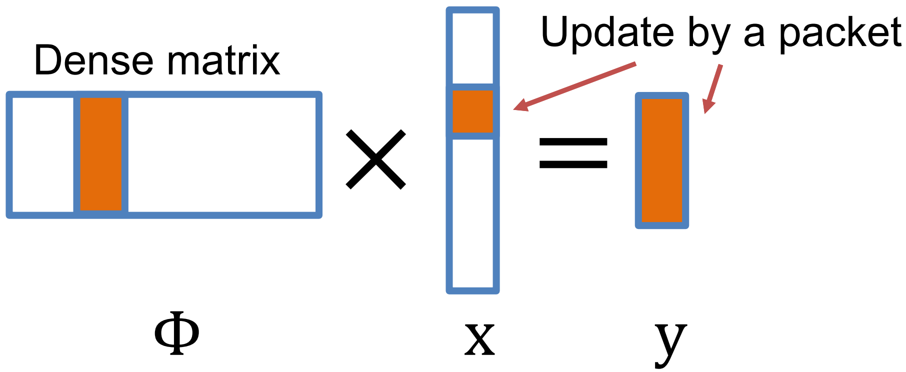
1^{st} Attempt: Directly Adopt

- 4 types of sensing matrix:
  - Bernoulli Matrix (BM), Fourier Matrix (FM), Gaussian Matrix (GM), and Incoherence matrix (IM)
- 2 recovery algorithms:
  - L1 norm minimization (L1)
  - Orthogonal Matching Pursuit (OMP)

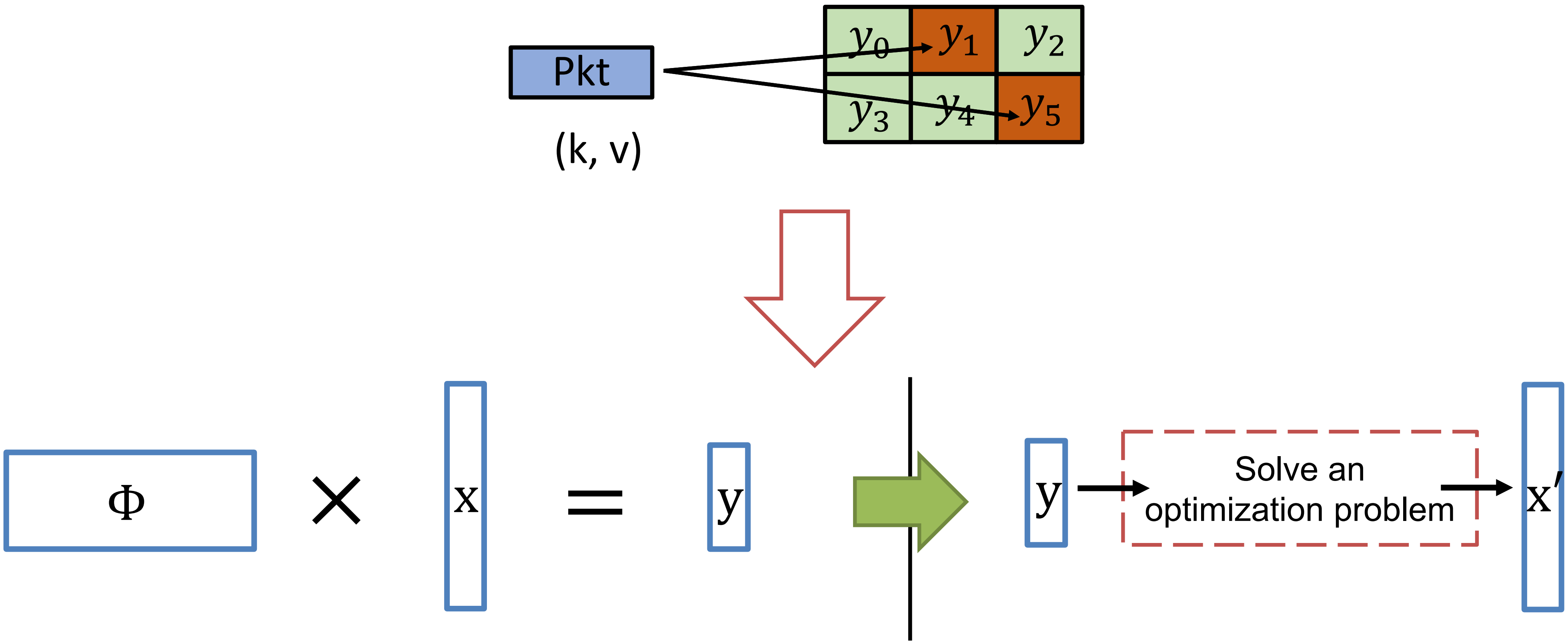
Advantage: limited memory to achieve nearly-zero error (High accuracy)



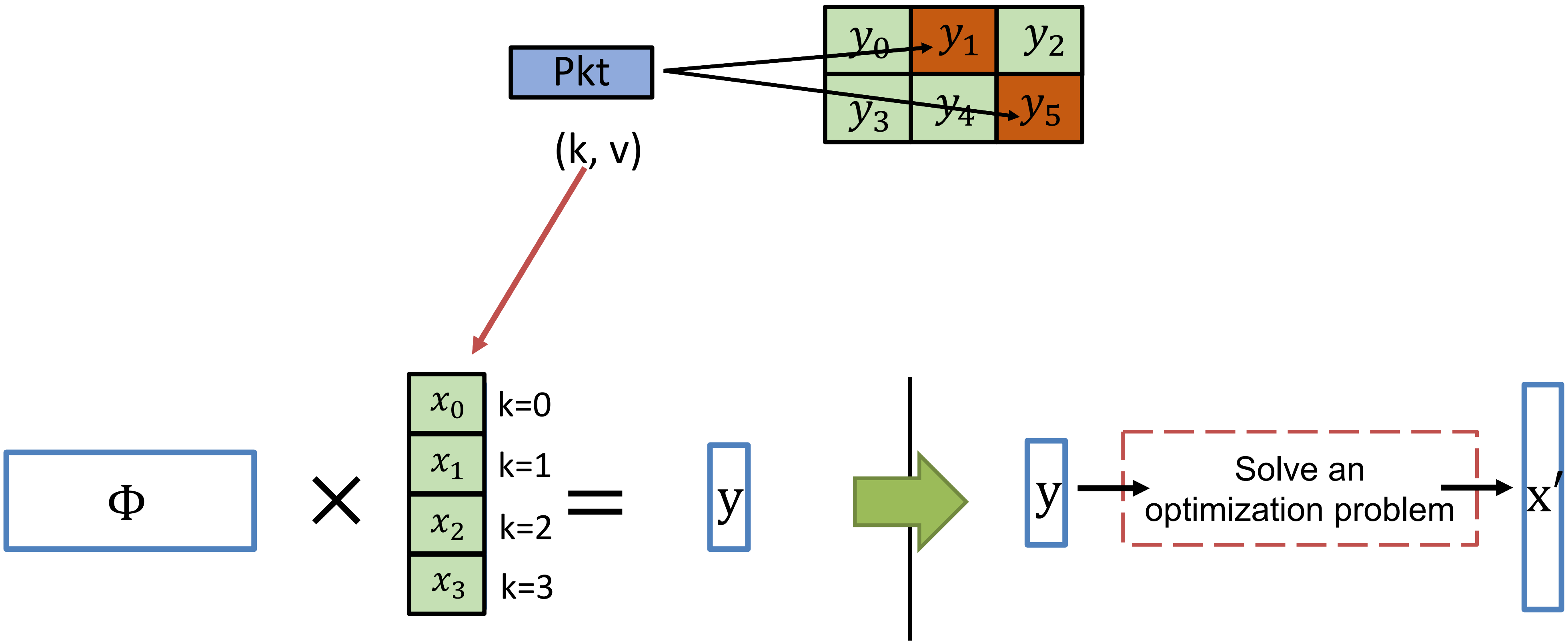
Limitation: each packet incurs a large number of updates (Poor scalability)



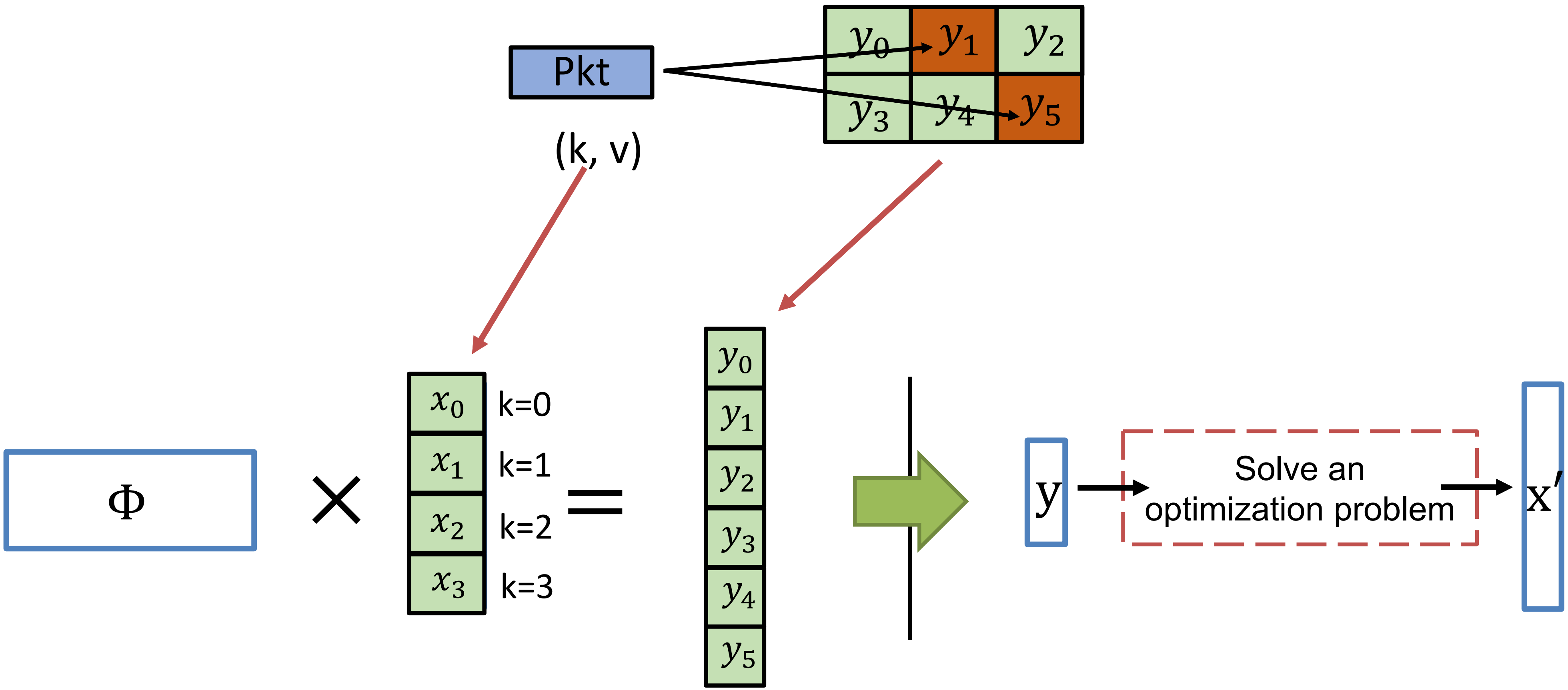
2^{nd} Attempt: Formulate Sketch



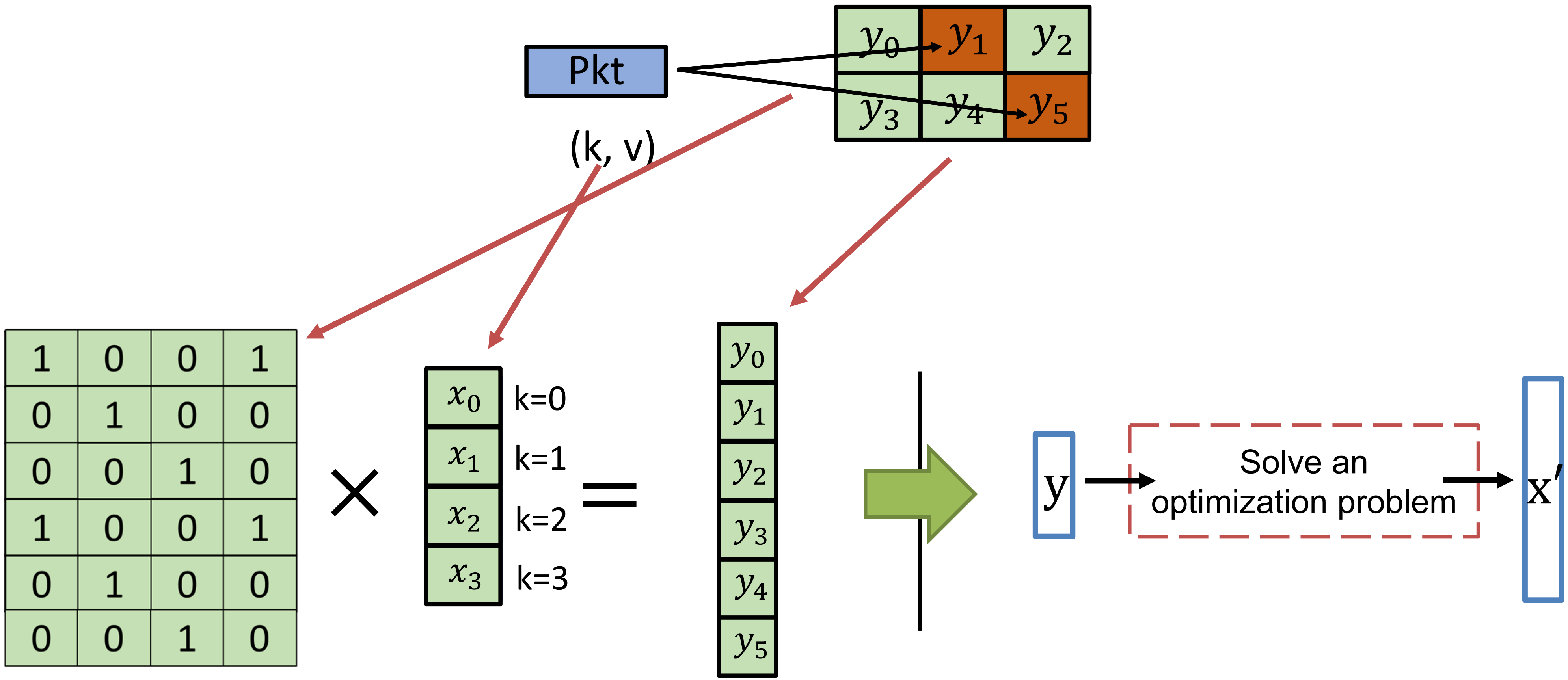
2nd Attempt: Formulate Sketch



2^{nd} Attempt: Formulate Sketch

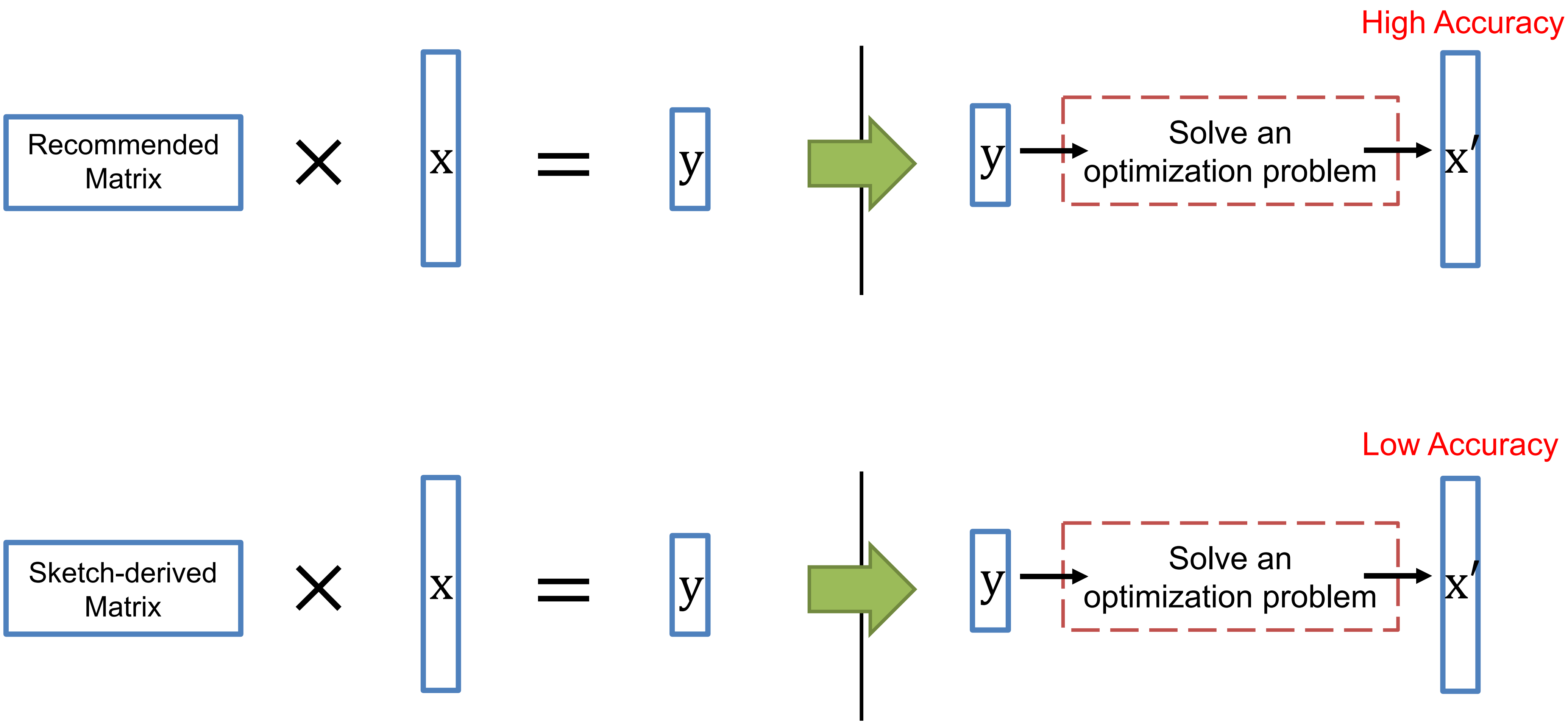


2nd Attempt: Formulate Sketch

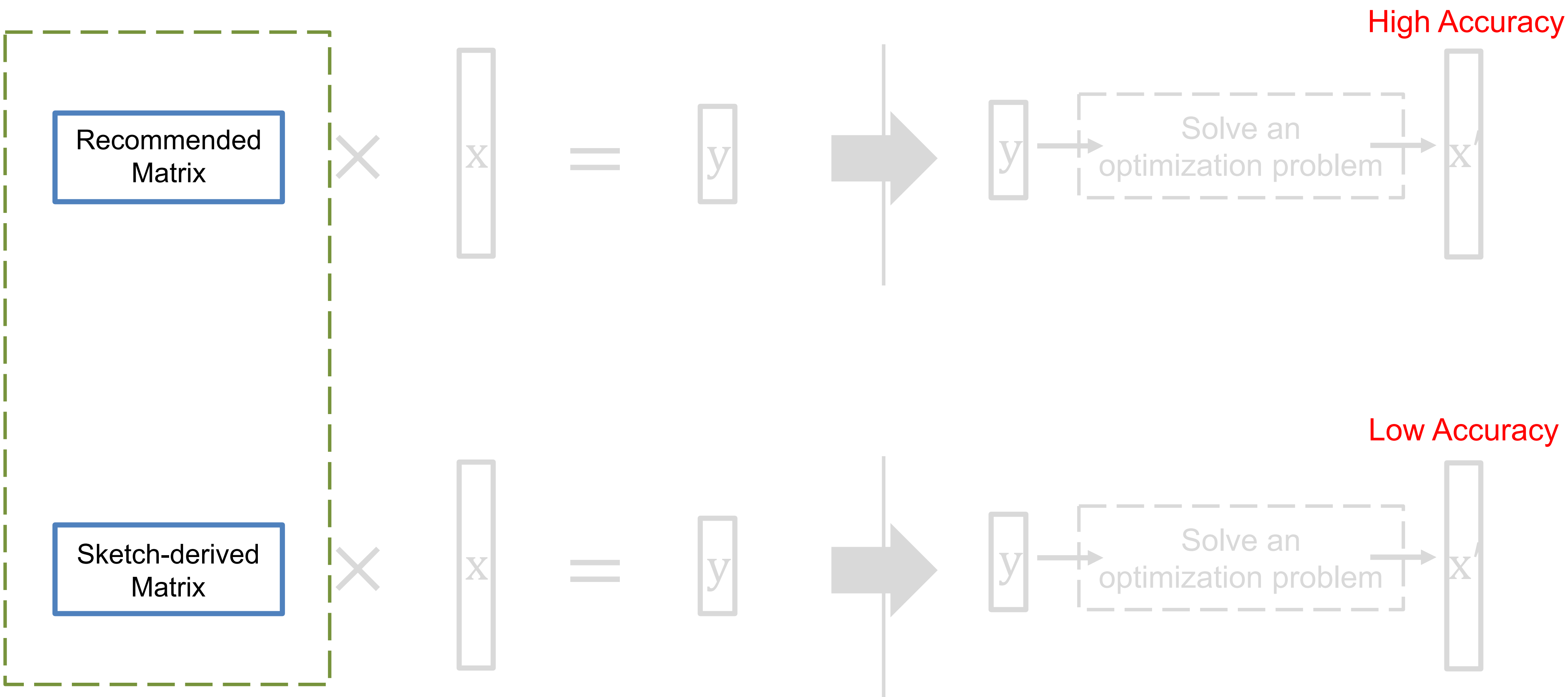


However, the recovery accuracy is low (see paper)

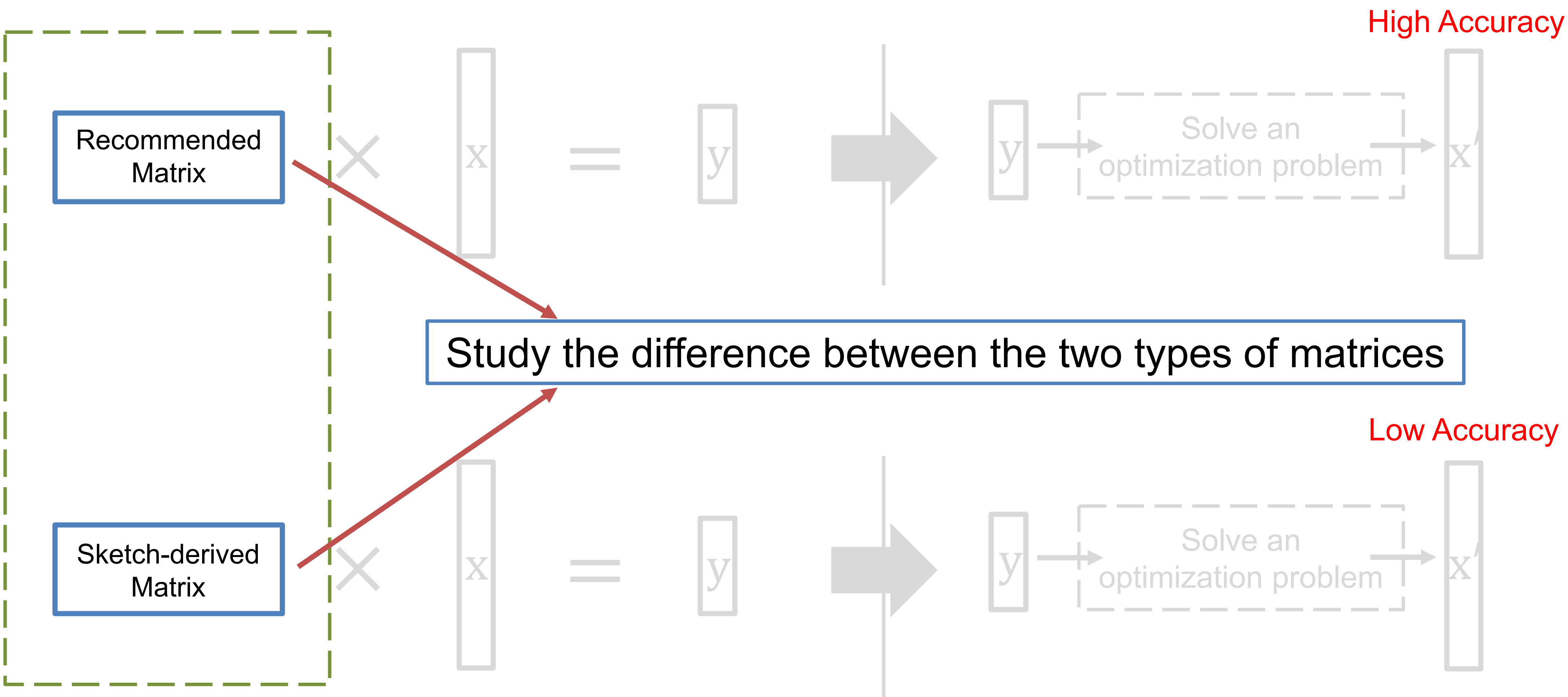
Key Matrix Property



Key Matrix Property

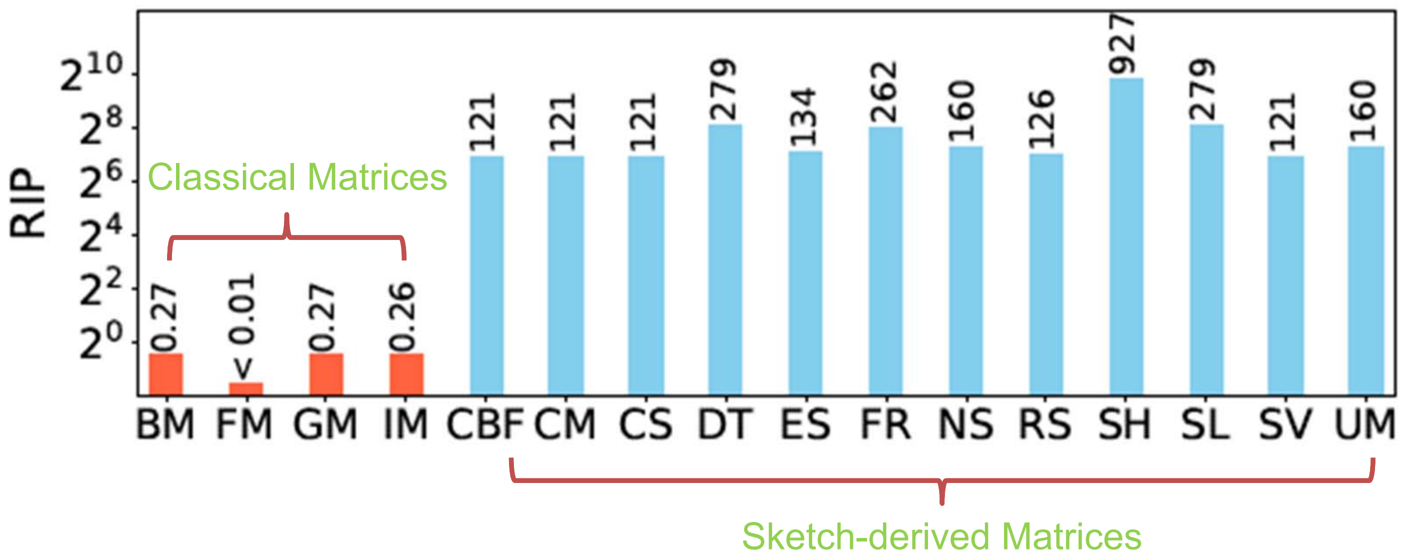


Key Matrix Property

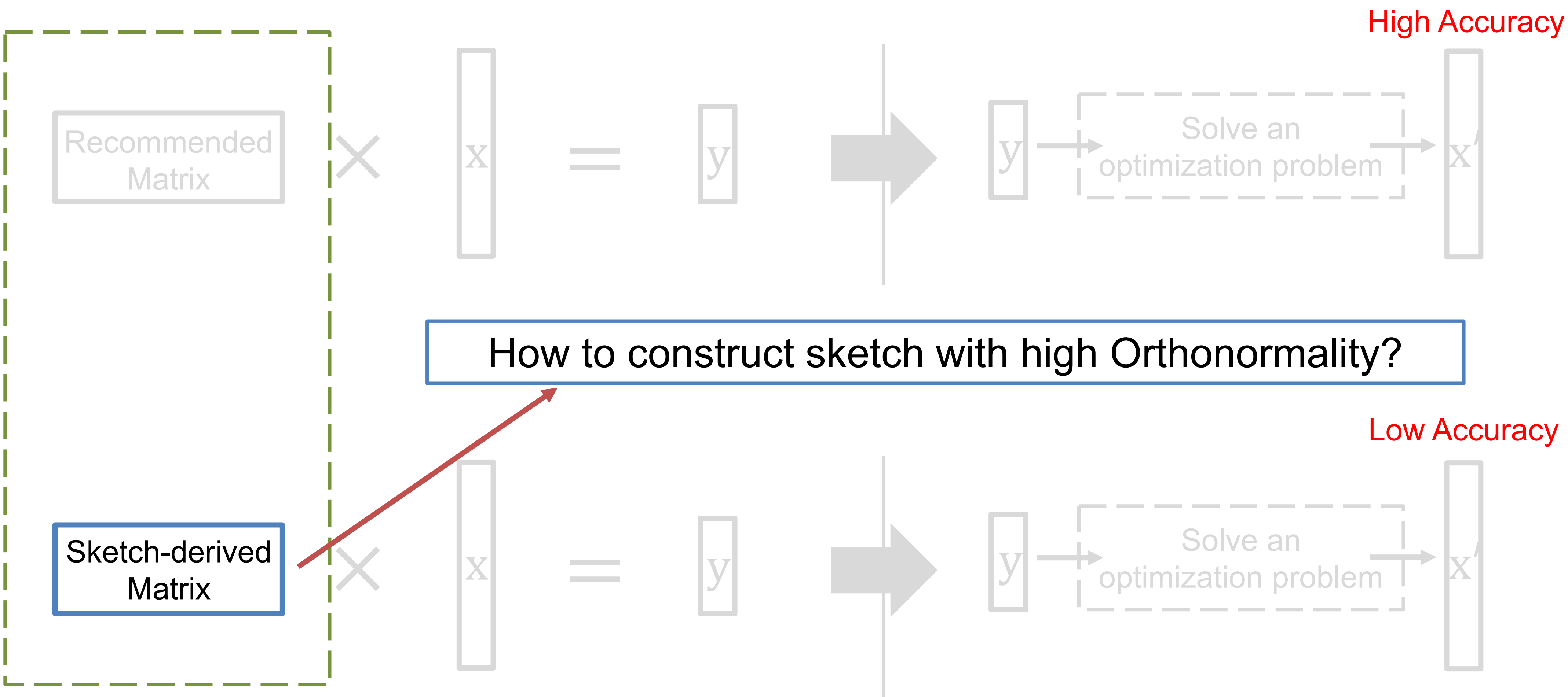


Key Matrix Property: Orthonormality

- **Orthonormality**: ability to preserve the norm of a sparse vector
  - High orthonormality: x can be preserved and accurately recovered
- **RIP** value: quantify orthonormality
  - The lower RIP, the higher orthonormality



Key Matrix Property: Orthonormality



Revisit Sketch Design

How common approaches affect matrix?

Class 1: Fractional Update

Methods:
Sampling, or
Conservative update

Examples:
CU Sketch [SIGCOMM' 02]
NitroSketch [SIGCOMM' 19]

Matrix property:
Fractional elements in matrix

RIP=5.00

1	1	0	1
0	1	1	1
1	0	1	0
0	1	0	0

RIP=2.47

0.7	0.8	0	0.9
0	0.7	0.8	0.7
0.6	0	0.6	0
0	0.5	0	0

Class 2: Adding Rows

Methods:
Maintain multiple simple
sketch structures

Examples:
FlowRadar [NSDI' 16]
UnivMon [SIGCOMM' 16]
SketchLearn [SIGCOMM' 18]

Matrix property:
More rows in the matrix

RIP=4.24

1	1	0	1
0	1	1	1

RIP=2.69

1	1	0	1
0	1	1	1
0	1	0	1
0	0	1	0

More rows

Class 3: Clearing Columns

Methods:
Store flowkeys separately

Examples:
FlowRadar [NSDI' 16]
UnivMon [SIGCOMM' 16]

Matrix property:
Clearing useless columns

RIP=5.00

1	1	0	1
0	1	1	1
1	0	1	0
0	1	0	0

RIP=2.45

1	0	0	0
0	0	1	0
1	0	1	0
0	0	0	0

Class 4: Matrix Decomposition

Methods:
Separate traffic into two parts,
or extract large flows

Examples:
SketchLearn [SIGCOMM' 18]
Elastic Sketch [SIGCOMM' 18]

Matrix property:
Decomposing simpler matrices

RIP=5.00

1	1	0	1
0	1	1	1
1	0	1	0
0	1	0	0

RIP=2.00

1	0	0	0
0	0	0	0
1	0	0	0
0	0	0	0

+

RIP=4.36

0	1	0	1
0	1	1	1
0	0	1	0
0	1	0	0

New Algorithms

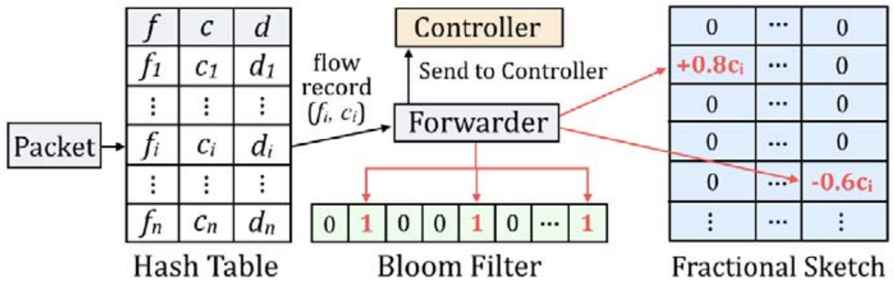
- Existing algorithms are not enough
  - Not specifically designed for compressive sensing
  - Use the common approaches, but not efficiently combine them
  - Orthonormality is not the main goal

- Need new algorithms
  - Combine the approaches more efficiently

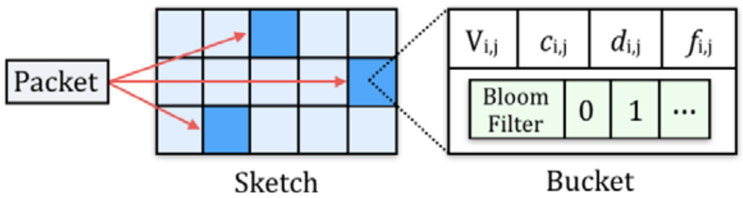
New Algorithms

Algorithm	C1	C2	C3	C4
CU Sketch [25]	Conservative update			
Deltoid [19]		Multiple CM instances		Flow extraction
ElasticSketch [79]				Traffic splitting
FlowRadar [48]		Multiple Bloom Filters	Bloom Filter	Flow extraction
NitroSketch [52]	Sampling	Multiple CS instances	Heap	
RevSketch [67]				Flow extraction
SeqHash [8]		Multiple CM instances		Flow extraction
SketchLearn [37]		Multiple CM instances		Flow extraction
SketchVisor [35]				Traffic splitting
UnivMon [53]		Multiple CS instances	Heap	
SeqSketch	Fractional update		Bloom Filter + Controller	Splitting + Controller
EmbedSketch	Fractional update		Bloom Filter + controller	Extraction + Controller

SeqSketch



EmbedSketch



Results

- RIP values <3
- Accuracy
  - 100% precision and 100% recall
  - <0.1% relative error for >99.7% flows
- Robustness under different memory configuration
- Low resource usage
  - Hardware resources
  - Bandwidth
- Recovery time

Conclusion

➢ Problem: nearly-zero-error monitoring

1. Classical compressive sensing achieves high accuracy but has poor scalability

2. Existing sketch algorithms suffer from low accuracy due to the low orthonormality

Sketch

Compressive Sensing

3. A framework can study the orthonormality of various common approaches

4. New algorithms are needed to combine the approaches more efficiently

Source Code Available: https://github.com/N2-Sys/NZE-Sketch

Thank You