Efficient Private File Retrieval by Combining ORAM and PIR

Travis Mayberry
Erik-Oliver Blass
Agnes Chan

Northeastern University

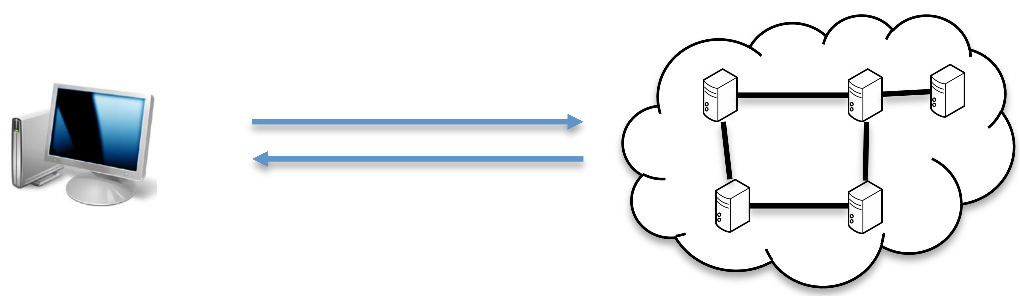
Hiding Access Patterns

Oblivious RAM

- Communication: High
- Rounds: Multiple
- Client computation: None
- Server computation: None

Private Information Retrieval

- Communication: Low
- Rounds: One
- Client computation: Low
- Server computation: High



Contributions

- We introduce a PIR bucket construction which allows recent ORAM protocols to be merged with PIR
- Consider the notion of an ORAM's data latency or online data
  - We define latency to be the amount of communication required before the client has full access to the requested data
- Using our bucket construction with the tree-based scheme of Shi et. al., we obtain an ORAM protocol with:
  - The lowest communication overhead of any constant-client-memory Oblivious RAM
  - Optimal data latency
- We evaluate our scheme on Amazon AWS and show that it has very low overall query time and monetary cost per query

Notation

- n : Number of blocks in the ORAM
- ℓ : Size of each block in bits
- k : Size of one ciphertext in bits

Helpful sample values:

$$n = 2^{25}$$

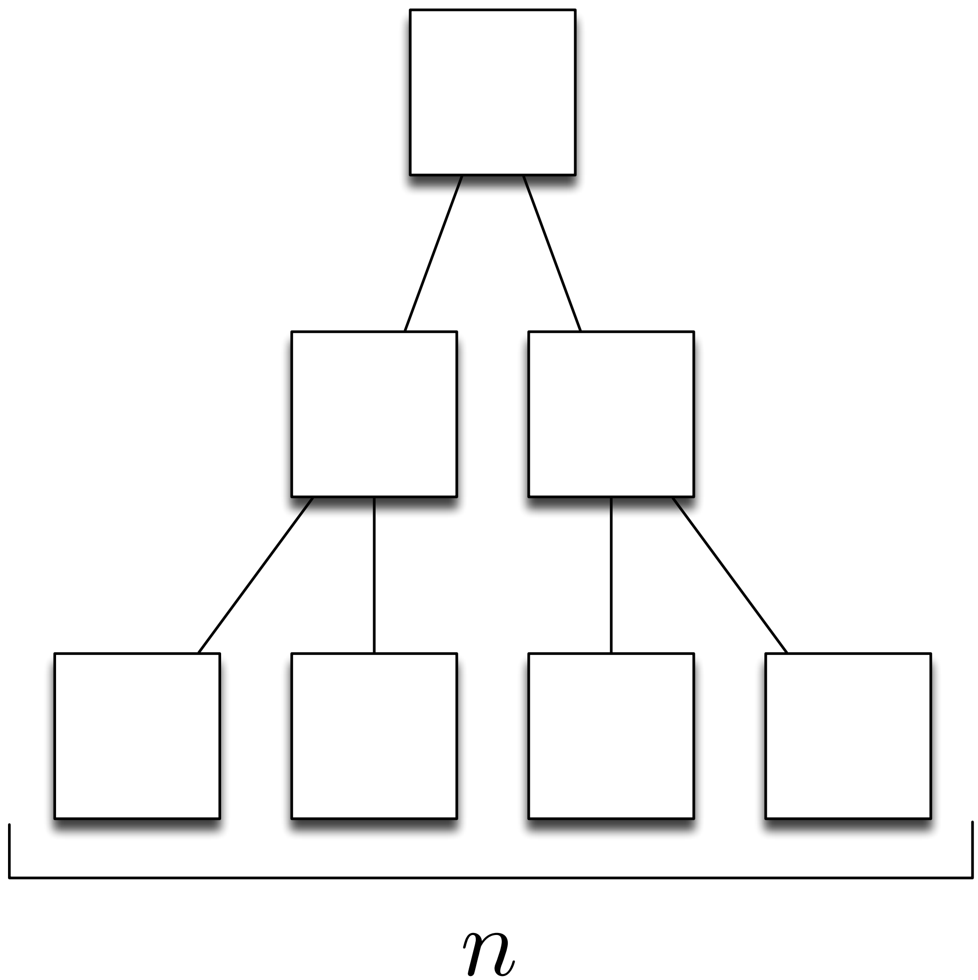
$$\text{Database} = 4 \text{ TB}$$

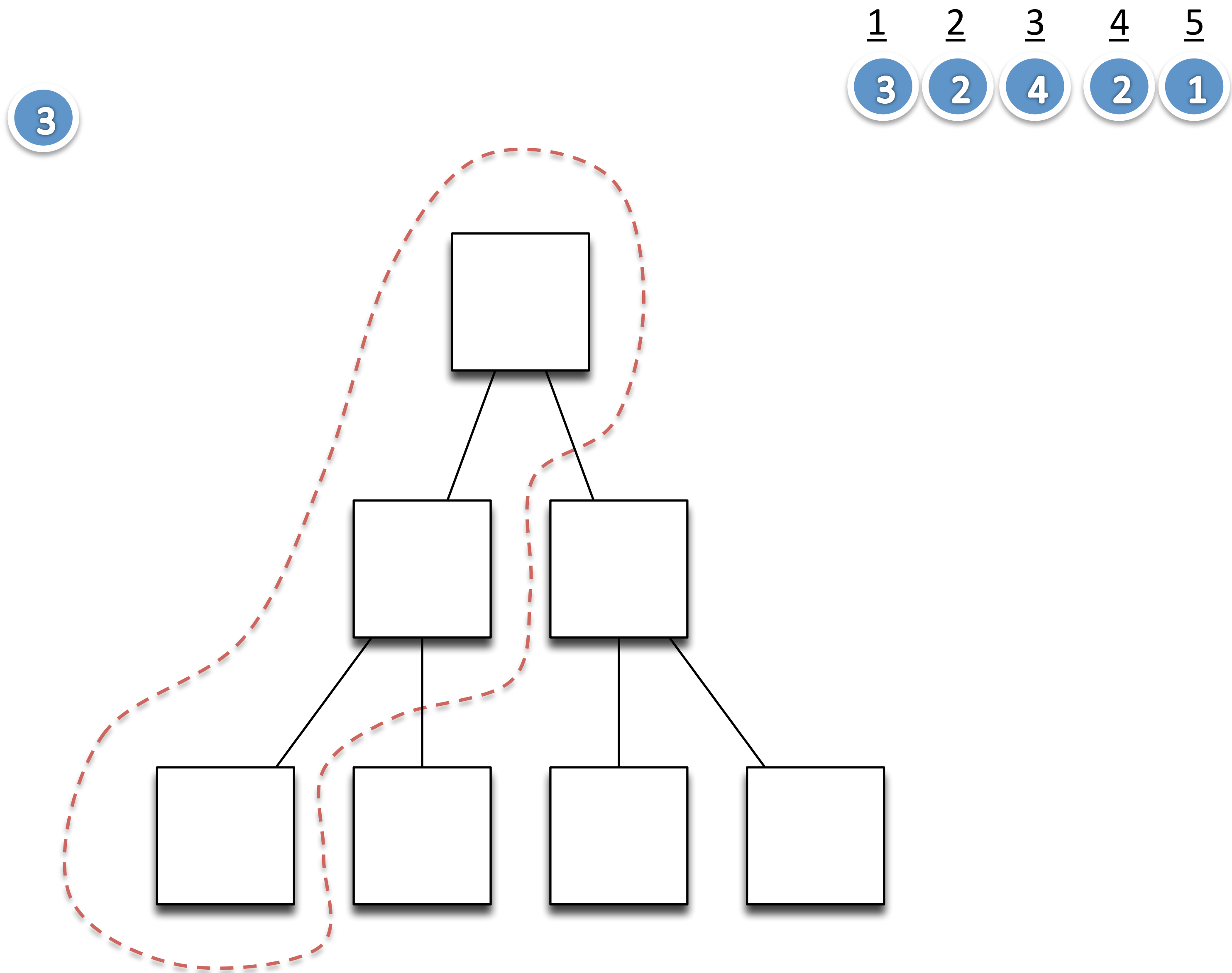
$$\ell = 1 \text{ MB}$$

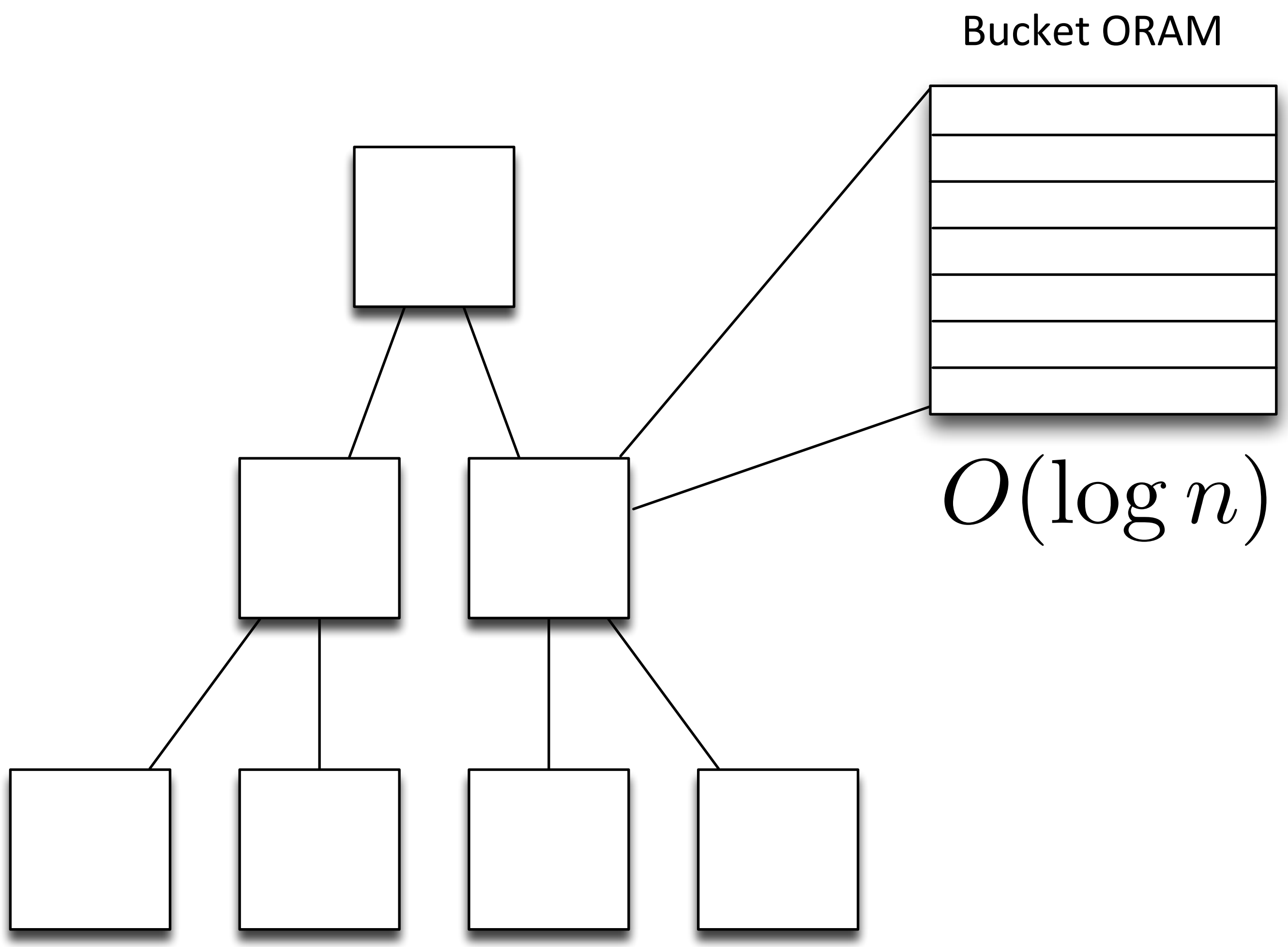
$$k = 2048 \text{ bits}$$

Shi et al

- First poly-logarithmic worst-case oblivious RAM
- New tree based construction
- Achieves $O(\ell \cdot \log^3 n)$ communication, with relatively good constants
- Consists of two phases: data access, and eviction


n


1 2 3 4 5
3 2 4 2 1
3


Bucket ORAM
$O(\log n)$

Private Information Retrieval

- Traditionally very computationally expensive, conjectured that it might never be feasible [SC07]
- Recently advances in homomorphic encryption have lead to practical schemes [MBC13][MG08], especially when ℓ is large compared to n

Query

E(0)
E(1)
E(0)
E(0)

nk

Database

*	$X_{1,1}$	$X_{1,2}$	$X_{1,3}$	$X_{1,4}$	=	E(0)	E(0)	E(0)	E(0)
*	$X_{2,1}$	$X_{2,2}$	$X_{2,3}$	$X_{2,4}$	=	$E(X_{2,1})$	$E(X_{2,2})$	$E(X_{2,3})$	$E(X_{2,4})$
*	$X_{3,1}$	$X_{3,2}$	$X_{3,3}$	$X_{3,4}$	=	E(0)	E(0)	E(0)	E(0)
*	$X_{4,1}$	$X_{4,2}$	$X_{4,3}$	$X_{4,4}$	=	E(0)	E(0)	E(0)	E(0)

+

Response

$E(X_{2,1})$	$E(X_{2,2})$	$E(X_{2,3})$	$E(X_{2,4})$

ℓ

$$O(nk + \ell)$$

10

To change X_i to X’, encrypt “delta”:
$Y_j = X'_j - X_{i,j}$

Query | Server Side

E(0)	*	$E(Y_1)$	$E(Y_2)$	$E(Y_3)$	$E(Y_4)$	=	E(0)	E(0)	E(0)	E(0)
E(1)	*	$E(Y_1)$	$E(Y_2)$	$E(Y_3)$	$E(Y_4)$	=	$E(Y_1)$	$E(Y_2)$	$E(Y_3)$	$E(Y_4)$
E(0)	*	$E(Y_1)$	$E(Y_2)$	$E(Y_3)$	$E(Y_4)$	=	E(0)	E(0)	E(0)	E(0)
E(0)	*	$E(Y_1)$	$E(Y_2)$	$E(Y_3)$	$E(Y_4)$	=	E(0)	E(0)	E(0)	E(0)

nk ℓ

$$O(nk + \ell)$$

11

Encrypted Delta

E(0)	E(0)	E(0)	E(0)
$E(Y_1)$	$E(Y_2)$	$E(Y_3)$	$E(Y_4)$
E(0)	E(0)	E(0)	E(0)
E(0)	E(0)	E(0)	E(0)

\+ \+ \+ \+

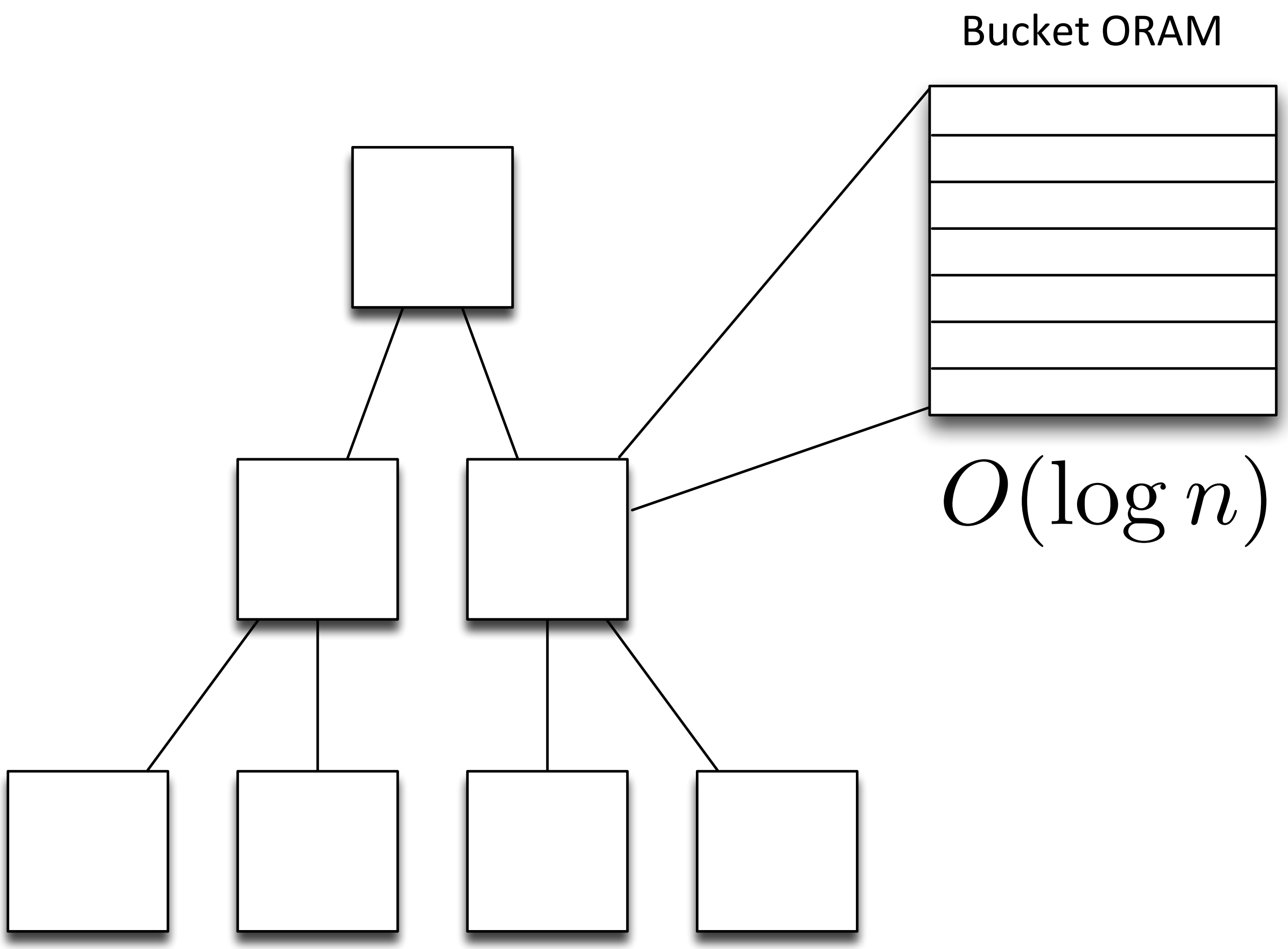
Encrypted Database

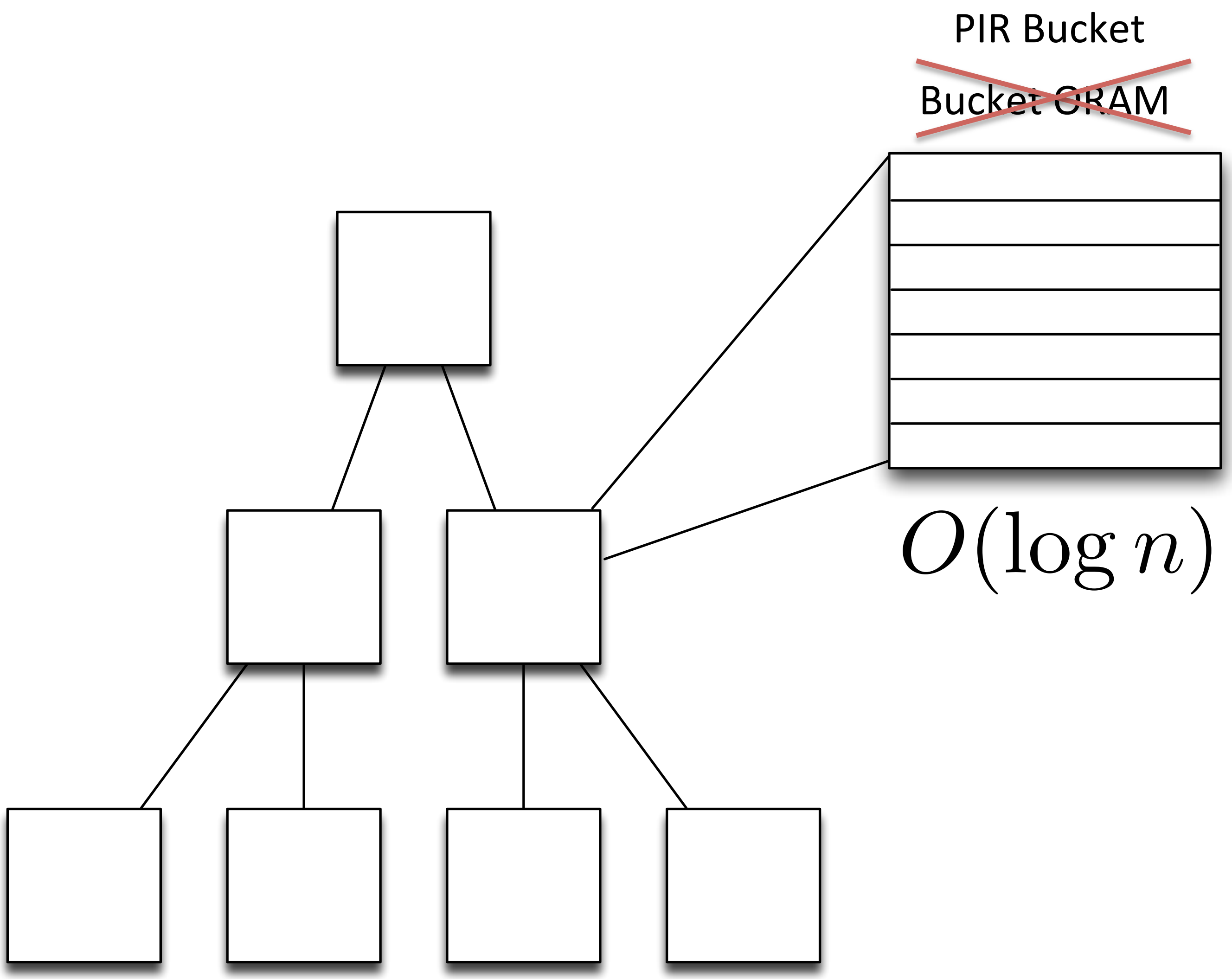
$E(X_{1,1})$	$E(X_{1,2})$	$E(X_{1,3})$	$E(X_{1,4})$
$E(X_{2,1})$	$E(X_{2,2})$	$E(X_{2,3})$	$E(X_{2,4})$
$E(X_{3,1})$	$E(X_{3,2})$	$E(X_{3,3})$	$E(X_{3,4})$
$E(X_{4,1})$	$E(X_{4,2})$	$E(X_{4,3})$	$E(X_{4,4})$

= = = =

$E(X_{1,1})$	$E(X_{1,2})$	$E(X_{1,3})$	$E(X_{1,4})$
$E(X'_1)$	$E(X'_2)$	$E(X'_3)$	$E(X'_4)$
$E(X_{3,1})$	$E(X_{3,2})$	$E(X_{3,3})$	$E(X_{3,4})$
$E(X_{4,1})$	$E(X_{4,2})$	$E(X_{4,3})$	$E(X_{4,4})$

12


Bucket ORAM
$O(\log n)$



$$O(\ell \cdot \log n) \quad \Longrightarrow \quad O(\log n \cdot k + \ell)$$

PIR Bucket

- Read blocks using linear PIR
- Write blocks using linear PIR-Writing
- Requires only additively homomorphic encryption!

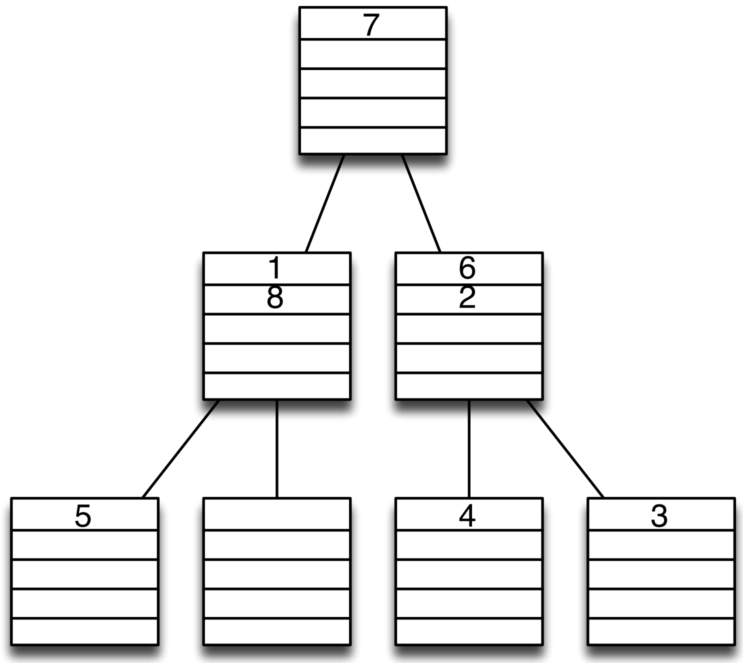
What does this give us?

- Better asymptotic communication
  - Old: $O(\ell \cdot \log^3 n)$
  - New: $O(k \cdot \log^3 n + \ell \cdot \log^2 n)$

	Worst-Case	Practical Worst-Case
Shi et al	$O(l \cdot \log^3(N))$	$O(l \cdot \log^2(N))$
Kushilevitz	$O(\frac{l \cdot \log^2(N)}{\log\log(N)})$	$O(l \cdot \log^3(N)$
Path-PIR Additive	$O(k \cdot \log^3(N) + l \cdot \log^2(N)$	$O(l \cdot \log(N))$
Path-PIR FHE	$O(k \cdot \log(N) + l \cdot \log(N))$	$O(k + l)$
Optimal	$O(\log(N) + l)$	$O(\log(N) + l)$

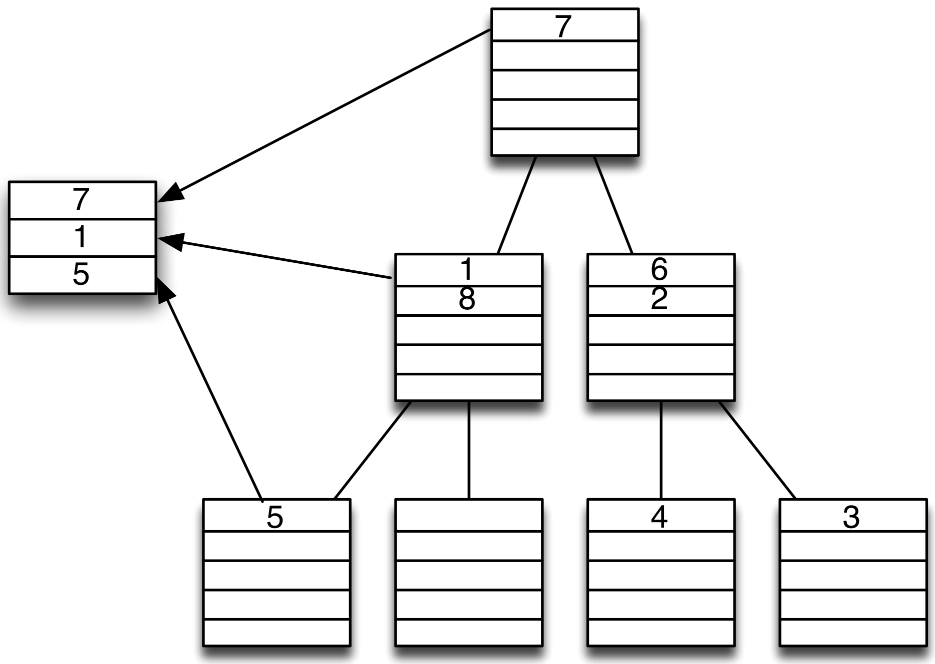
Also interesting: good latency!

1) Client requests to read block 5



1) Client requests to read block 5

2) Naïve way: use PIR to retrieve 1st element of each bucket

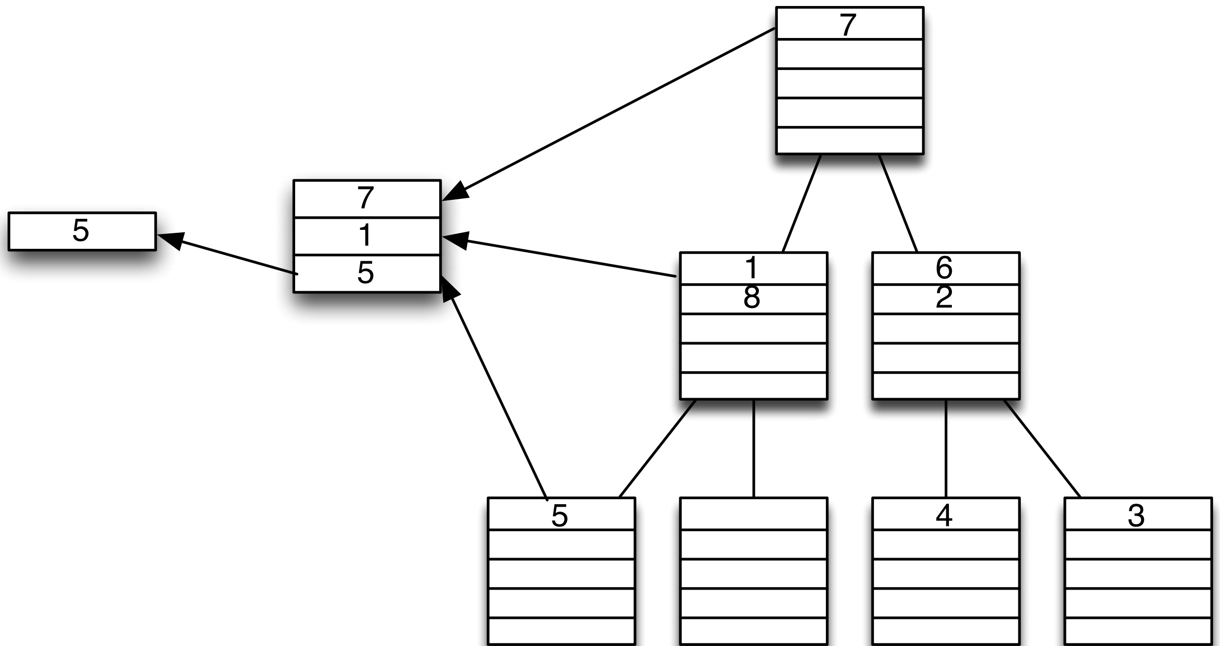


$$O((\ell + k) \cdot \log n)$$

1) Client requests to read block 5

2) Naïve way: use PIR to retrieve 1st element of each bucket

3) Use PIR again to retrieve 3rd element of previous results



$$O(k \cdot \log n + \ell)$$

This is optimal!

What good is that?

- Latency represents how responsive the ORAM is to client interactions
  - If most of the communication happens in the background, after the client receives their data, it is much more acceptable in real world scenarios
- Also allows the client to take advantage of interesting network asymmetries…

Cell network data is expensive ☹

WiFi Data is cheap ☺

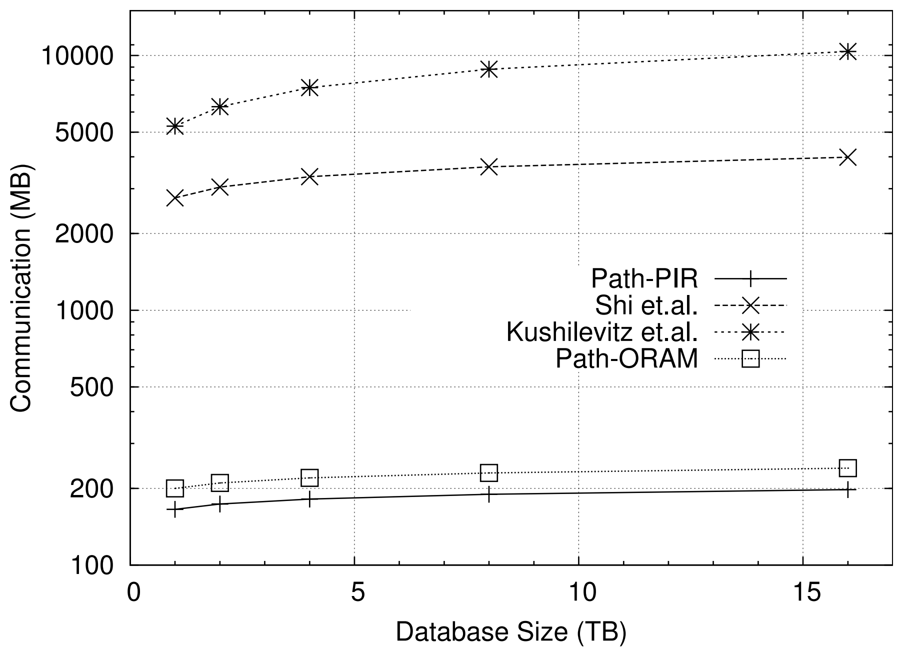
Defer eviction while you are out

Complete “bookkeeping” when you get home

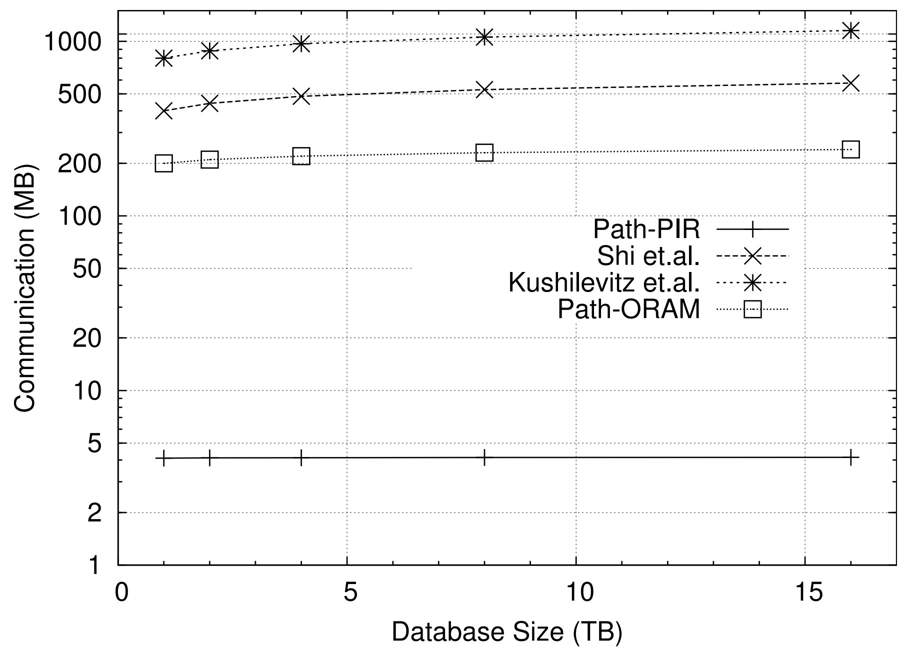
	Latency	Worst-Case	Practical Worst-Case
Shi et al	$O(l \cdot \log^2(N))$	$O(l \cdot \log^3(N))$	$O(l \cdot \log^2(N))$
Kushilevitz	$O(\frac{l \cdot \log^2(N)}{\log\log(N)})$	$O(\frac{l \cdot \log^2(N)}{\log\log(N)})$	$O(l \cdot \log^3(N)$
Path-PIR Additive	$O(k \cdot \log(N) + l)$	$O(k \cdot \log^3(N) + l \cdot \log^2(N)$	$O(l \cdot \log(N))$
Path-PIR FHE	$O(k + l)$	$O(k \cdot \log(N) + l \cdot \log(N))$	$O(k + l)$
Optimal	$O(\log(N) + l)$	$O(\log(N) + l)$	$O(\log(N) + l)$

Communication Comparison

Overall (per query)

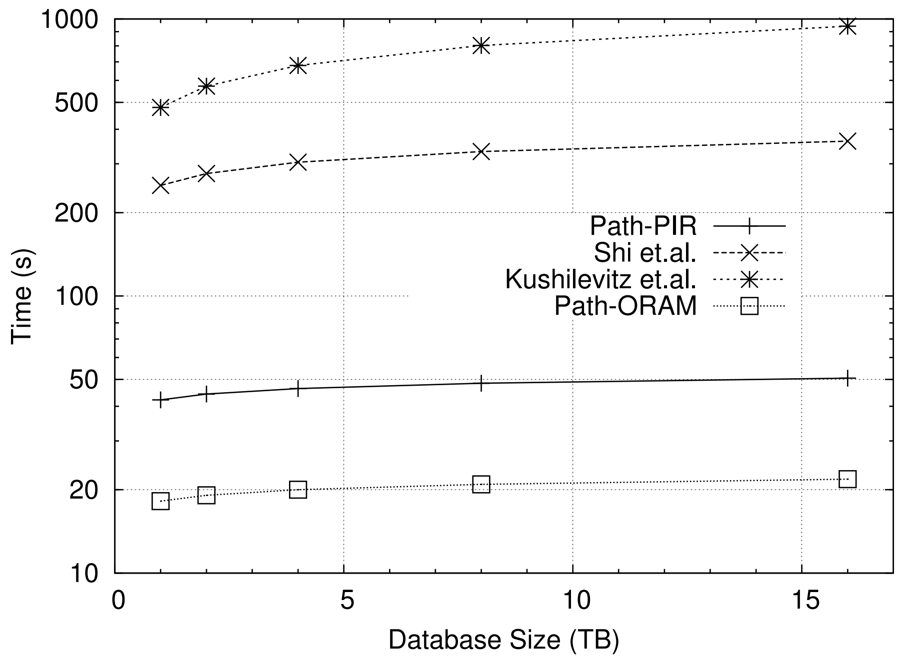


Latency (per query)

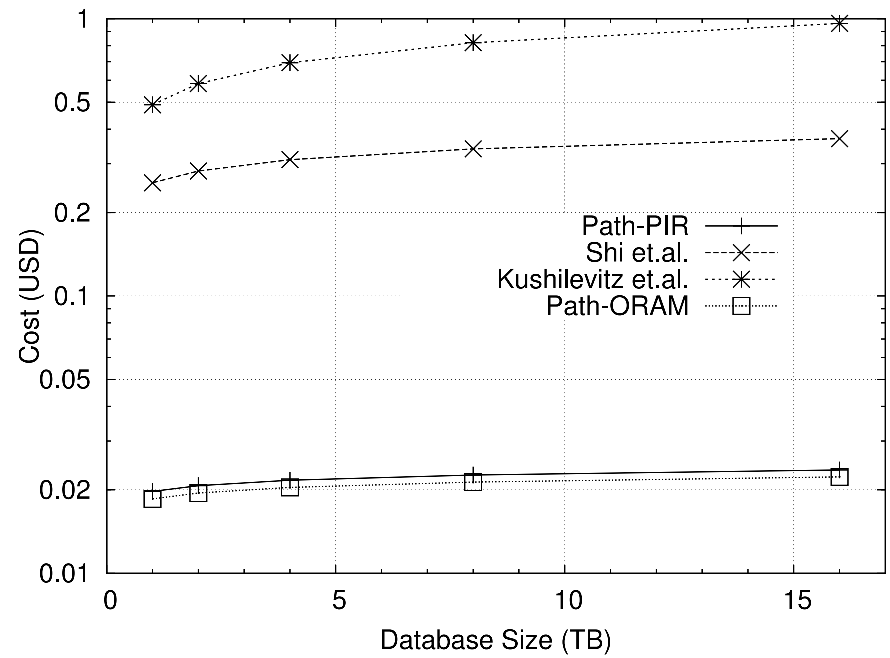


But what about expensive computation?

Total Time (per query)



Dollar Cost on AWS (per query)



Northeastern University

Conclusion

- We have introduced a technique for applying PIR to ORAM protocols which results in significantly decreased communication
- Combining our technique with an existing scheme leads to an efficient ORAM protocol with very low (optimal) latency
- Our protocol was tested on Amazon AWS and shown to be cheaper and faster than related work