Algorithms in HElib

Shai Halevi and Victor Shoup

http://eprint.iacr.org/2014/106

What's HElib?

- A software library implementation of homomorphic encryption (HE)
  - Implements the ring-LWE variant of [BGV12]
  - Written in C++, uses NTL for poly-arithmetic
  - Open source (GPL), available off https://github.com/shaih/HElib/
- Focused on effective use of "ciphertext packing" [SV11] and the routing techniques from [GHS12]

What's Implemented in HElib?

Low level: the cryptosystem itself

- Useful analogy: "assembly language" for homomorphic computations
- The HE schemes serves as a "hardware platform"
  - Defining the available operations and their cost

Higher level: algorithms over this "platform"

- Routing algorithms ⇐ this talk
- Simple linear functions

The HE "platform"

- Each ciphertext encrypts a **vector** $v \in F^n$
  - F can be "any finite field" of our choice
  - n is determined by the system parameters
    - Typical values are $n \in [100, 1000]$
- Operations are rotations, element-wise addition & multiplication
  - This is a SIMD environment
  - Not very different from Intel SSE etc.

The HE "platform"

- Cost measured in time, noise
  - Noise plays the role of circuit depth

Operation	Time cost	Noise cost
Constant Addition	Cheap	Cheap
Addition	Cheap	Cheap
Constant Mult.	Cheap	Moderate
Multiplication	Expensive	Expensive
Rotation	Expensive	Cheap

Routing Procedures

- Rotations ➔ Arbitrary Permutations using "Generalized" Benes networks
- Standard Benes works for size-2^n networks
  - Has depth $2n$, "cost" $4n$
- Two different generalizations:
  - Network size $\prod_{i=1}^{n} a_i$ with depth $2n$, "cost" $2\sum_i a_i$
  - Network size N with depth $2\log_2 N$, cost $4\log_2 N$
- Combining the two generalizations to optimize depth, cost

Routing Procedures

- Rotations ➔ Arbitrary Permutations using "Generalized" Benes networks
- Standard Benes works for size-2^n networks
  - Has depth $2n$, "cost" $4n$
- Two different generalizations:
  - Network size $\prod_{i=1}^{n} a_i$ with depth $2n$, "cost" $2\sum_i a_i$
  - Network size N with depth $2\log_2 N$, cost $4\log_2 N$
- Open: unify these two generalizations
  - May yield a 2x performance improvement

Illustrative Timing Results

Cyclotomic Field	Vector size	Network depth	Network size	Permutation time
$m = 4369$	$n = 256$	3	60	4.1 sec
		7	35	3.6 sec
		10	31	3.8 sec*
$m = 8191$	$n = 630$	5	37	5.0 sec
		7	30	4.3 sec
		9	28	4.0 sec
$m = 21845$	$n = 1024$	5	66	21.2 sec
		7	45	18.3 sec*
		9	41	16.7 sec*

* Larger depth requires larger parameters