Exact Encrypted Lookups at 0.07 µs on One GPU: Measured, Verified, and Compared on the Same Card

Graham Mefford, Duke University (student; independent work) — grahammefford@gmail.com

Abstract. Exact nonlinear functions are where fully homomorphic encryption spends its money: every comparison, activation or decision rule in an encrypted program ends in a lookup table, and every lookup needs a refresh. We report a GPU implementation that evaluates exact, refreshed 4-bit lookup tables on CKKS ciphertexts at 138.5 bits of estimated security in 0.072 µs per lookup on one H100 and 0.141 µs on a consumer RTX 3090, amortized over 65,536 values. Measured on the same H100, this is about 750 times less time per lookup than the GPU programmable bootstrapping of TFHE-rs 1.8.2. On the same GPU it is 14 times less time per value than a leading library's bare CKKS bootstrap, which computes no lookup at all. Every output of every reported run was decrypted and checked against its plaintext reference: zero mismatches across hundreds of millions of lookups. Beyond the numbers, the talk presents the discipline that makes such claims checkable by others (predictions registered before each run, same-machine controls, deliberately broken variants that must fail, and an analytic failure bound with its assumption stated) and an encrypted fraud-scoring model whose accuracy equals plaintext while the decryption key never leaves the client. We close with the limits of these results and the directions that failed. The construction will be described in a forthcoming paper.

1. Results

Comparison	Hardware	Result	Correctness
Exact refreshed 4-bit lookup, 138.5-bit estimated security	RTX 3090 (24 GB)	0.141 µs per lookup, amortized over 65,536 values	0 wrong; 2 seeds x 12 cycles
Same, data-centre GPU	H100 SXM	0.072 µs per lookup	0 wrong; 2 seeds x 12 cycles
vs TFHE-rs 1.8.2 CUDA, 4-bit programmable bootstrap	same H100	54.2 µs vs 0.072 µs: ~750x less time per lookup	0 wrong both; every output decrypted
vs GPU port of OpenFHE functional bootstrapping method	same RTX 3090	1.486 µs vs 0.138 µs: 10.7x	0 mismatches both
vs Cheddar bare CKKS bootstrap (its own test; no lookup)	same RTX A6000	2.25 µs vs 0.160 µs per value: 14.1x	0 wrong (ours)
vs OpenFHE functional bootstrapping, one CPU core	rental prices	~496x lower USD per lookup	both exact
Encrypted fraud scorecard (public ULB data), client-held key	RTX 3090	53 µs GPU time per record; AUC 0.9534, equal to plaintext	verifier: 0 mismatches

An "exact" lookup means that every decrypted output, rounded to its 4-bit (or 2-bit) alphabet, equals the plaintext table value. Each run decrypts and checks every output. Rival systems ran on the same pod with their shipped parameters and benchmarks. The comparison against TFHE-rs tried 1, 2, 4 and 8 concurrent streams; a second stream added about 1%, and 4 or 8 streams exceeded GPU memory.

2. How the numbers are verified

Registered predictions. Before each GPU run we record the expected result. Misses are kept and explained. **Same-pod controls.** Every speedup is a ratio against a control run on the same rented machine in the same session, because host-to-host variance on rented GPUs reaches 25-30%. **Canaries.** Each optimization ships with a deliberately broken variant that must fail; a canary removing one correction step produced 3.93 million wrong outputs, confirming that the checker detects real errors. **Security.** Parameters are checked with the lattice estimator (138.5 bits for the main set, 128.2 bits for a faster variant). **Failure probability.** An analytic bound of at most 2^-133.6 per lookup, conditional on a stated assumption about the noise tail; finite zero-error tests are reported separately and are not offered as a bound. **Decryption-output attacks (IND-CPA-D).** A written check of the deployment conditions: only rounded outputs are released and only compiled programs run.

3. Deployment

A compiler maps a trained scorecard model onto the encrypted computation and refuses to emit a program that an independent evaluator does not reproduce on at least 200,000 random inputs. The client generates keys with a ChaCha20-based sampler, sends evaluation keys and ciphertexts to the server, and keeps the secret key; outputs were byte-identical to a single-process run. Two unseen models ran through the compiler without hand edits.

4. Limits

The amortized figures need batches of 65,536 values and are not single-value latency; TFHE-rs is designed for latency per value. Steady-state timings assume a fixed model whose plaintext constants are built once; with per-batch random tables, wall time per cycle is about 1.5x higher. The TFHE-rs output used here is not composable without a second bootstrap, which favours TFHE-rs by up to 2x. Cost ratios depend on GPU rental prices (the same speed priced at on-demand 4090 or 5090 rates gives 81-165x). Each measurement is from one host; independent reproduction is in progress.

5. Negative results

Directions that failed under the same discipline: a lower-precision refresh, schedules that skip refreshes (rejected by multiplicative-depth limits), and further gains past a noise floor set by rescaling. We report them because they map where the remaining cost lies.