APPENDIX

Theorem 2. ***(Accuracy)*** *When $w \times d > R_d \times M + \epsilon$ and $M \geqslant \Omega(d^{4d} log^d(M))$, the decoding of HyperIBLT fails with probability $O(\frac{1}{M^{d-2}})$, where ϵ and R_d are small constants.*

$$R_d = \left(sup \left\{ \alpha \middle| \alpha \in (0,1), \forall x \in (0,1), 1 - e^{-d\alpha x^{d-1}} \right\} \right)^{-1}$$

For example, $R_3 = 1.222, R_4 = 1.295, R_5 = 1.425, R_6 = 1.570$.

Proof. Our result is a further analysis based on the theory of the 2-core in random hyper graph [89], [90] and IBLT. The merge and the sum of values in HyperIBLT do not incur additional errors, because the secure aggregation is lossless and the sum set is fully inserted to one HyperIBLT on the server. The additional error we introduced is the false positives when we use Rehash to verify the pure buckets. The IBLT assumes there is no error when verifying buckets because IBLT uses an additional hashkeySum field that can be long enough, *e.g.*, 64-bit. The results of 2-core and IBLT show that the failure probability without wrong verification is $O(\frac{1}{M^{d-2}})$.

When we use Rehash to verify a non-pure bucket, it has $\frac{1}{w}$ probability to mistake it for a pure one. In the decode procedure, the Rehash runs at most $O(Md)$ times, because each key changes d buckets and there are M keys with a few possible *wrong keys*. As $M = O(wd)$, by Chernoff bound, the rehash is wrong less than $F = O(d^2 log(\delta^{-1}))$ times with probability $1 - \delta$. When $\delta = O(\frac{1}{M^{d-2}})$, the times of wrong rehash will not exceed $F = O(d^3 log(M))$ in most cases (*i.e.*, $1 - O(\delta)$). A wrong rehash will incur a wrong key with a wrong deletion that influences d buckets. There is at most Fd buckets can be influenced, called poisoned buckets. The existing study [46] of poisoned bucket shows that, although some wrong keys could occur, they will be automatically recovered by the decode operation, and the probability of no key failing to be decoded due to the poisoned bucket is $O((\frac{Fd}{M})^d) = O(\frac{d^{4d} log^d(M)}{M^{(d-1)}})$. When setting $M = \Omega(d^{4d} log^d(M))$, the failure probability is $\delta = O(\frac{1}{M^{d-2}})$. Although $d^{4d} log^d(M)$ is large for a large d, in practice, $M \geqslant 10^4$ is always sufficient, corresponding to a memory cost of around 50KB. □