(Fully) Homomorphic Encryption

Vasco Rikkers

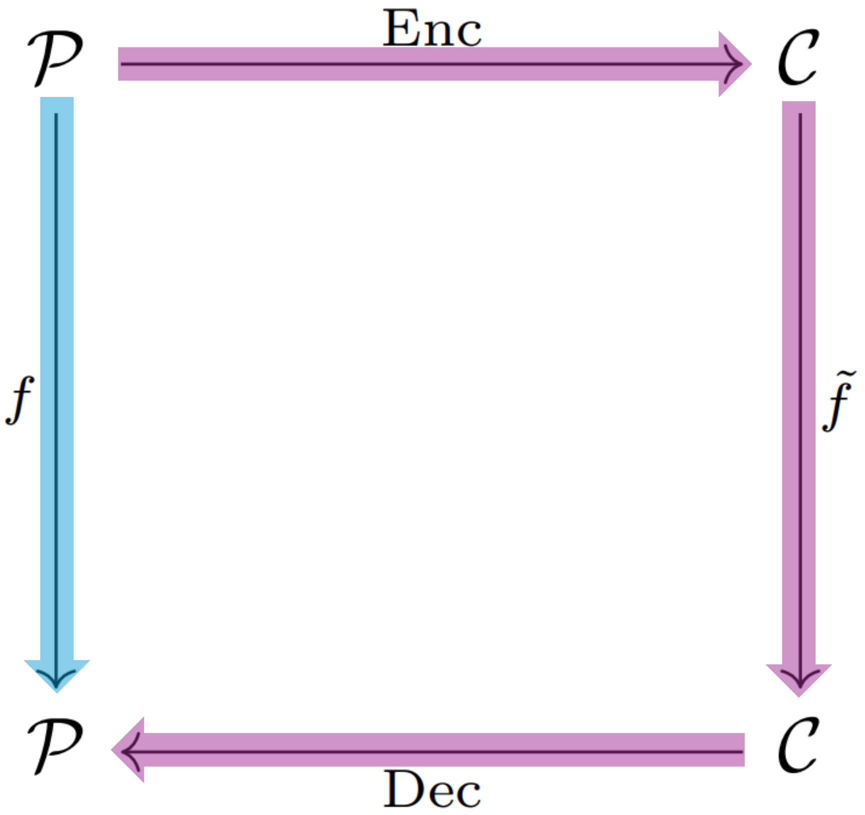
SCS seminar

11-04-2025

UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

What is FHE?

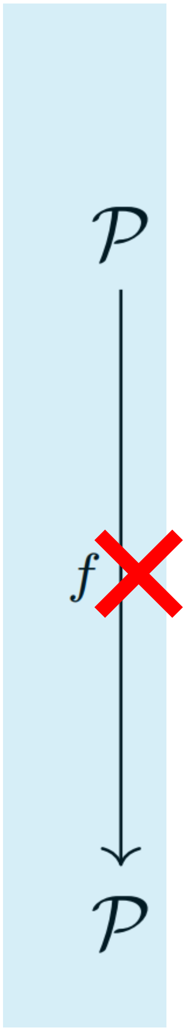
Definition 1. Let $\mathrm{Enc}\colon \mathcal{P} \to \mathcal{C}$ be an encryption function, for some plaintext space $\mathcal{P}$ and some ciphertext space $\mathcal{C}$. Let $\mathrm{Dec}\colon \mathcal{C} \to \mathcal{P}$ be the associated decryption function. We say that the scheme $S = (\mathrm{Enc}, \mathrm{Dec})$ is *fully homomorphic* if for any function $f\colon \mathcal{P} \to \mathcal{P}$ there exists some $\tilde{f}\colon \mathcal{C} \to \mathcal{C}$ such that the following diagram commutes:



$$f(m) = \tilde{f}(Enc(m))$$

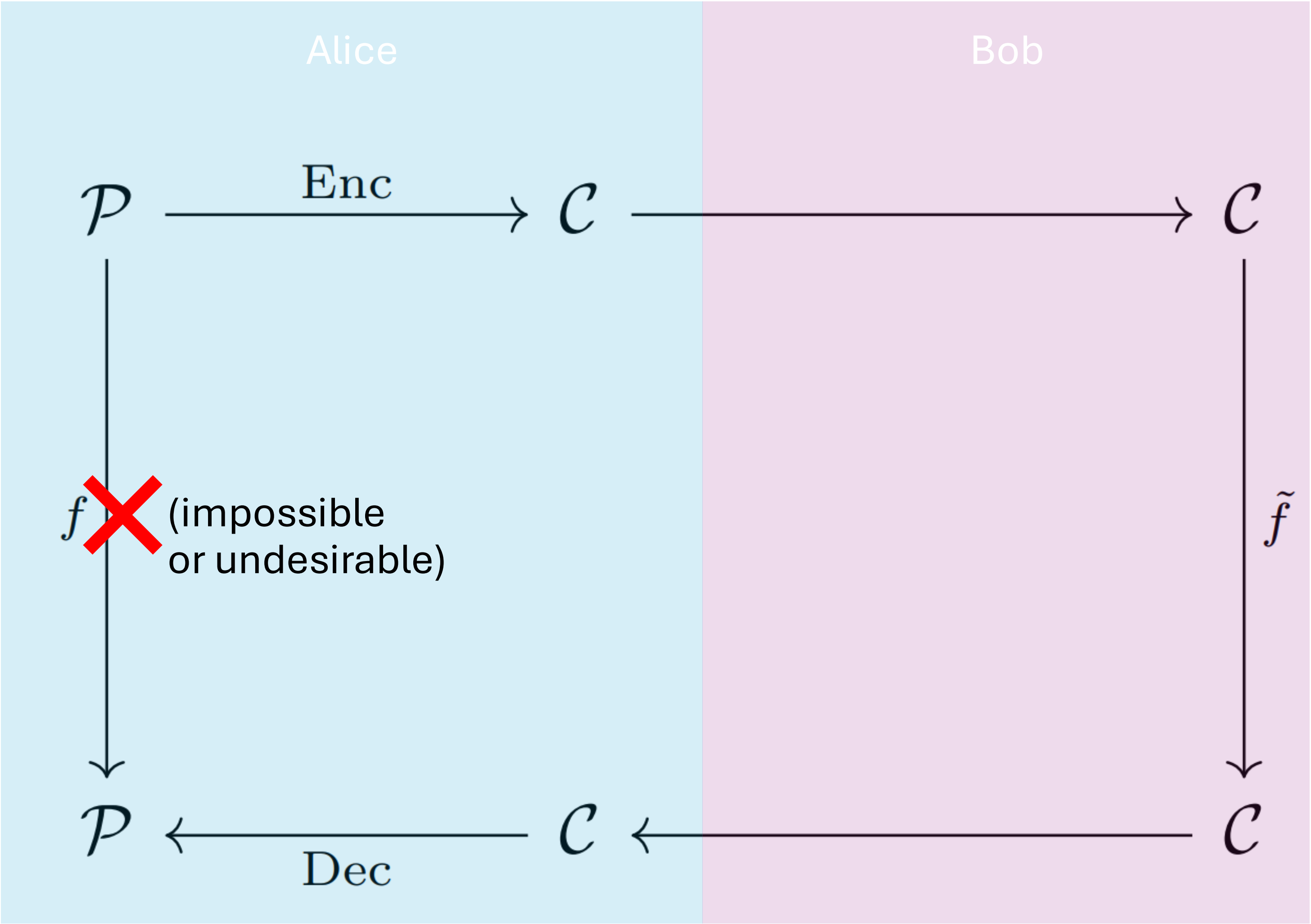
General use case

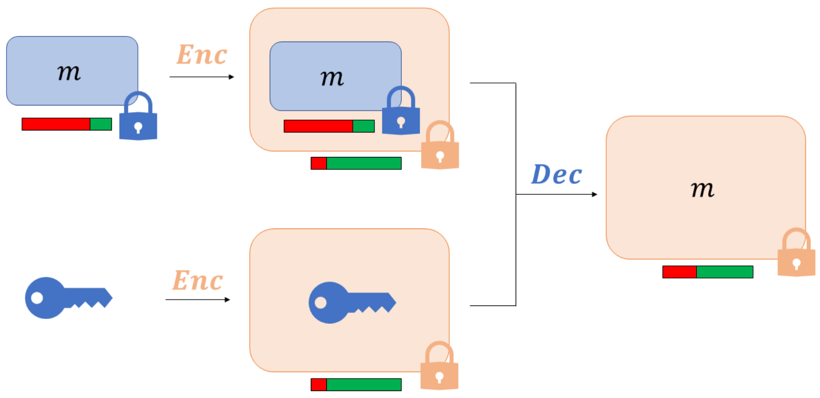
Alice cannot or does not want to calculate f herself.



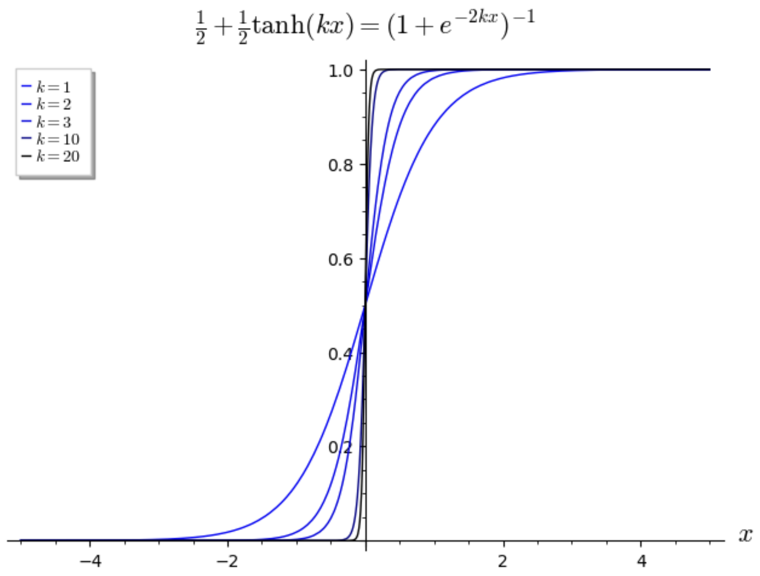
General use case

Alice cannot or does not want to calculate f herself.





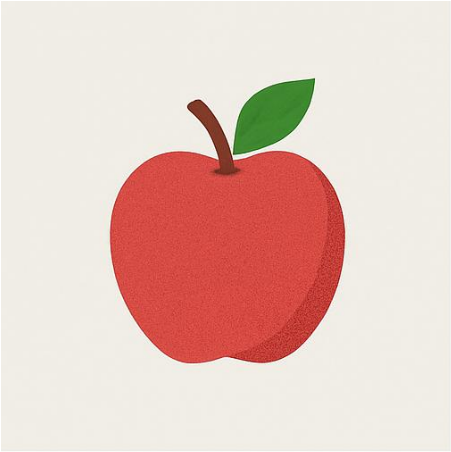
Bootstrapping



Approximation

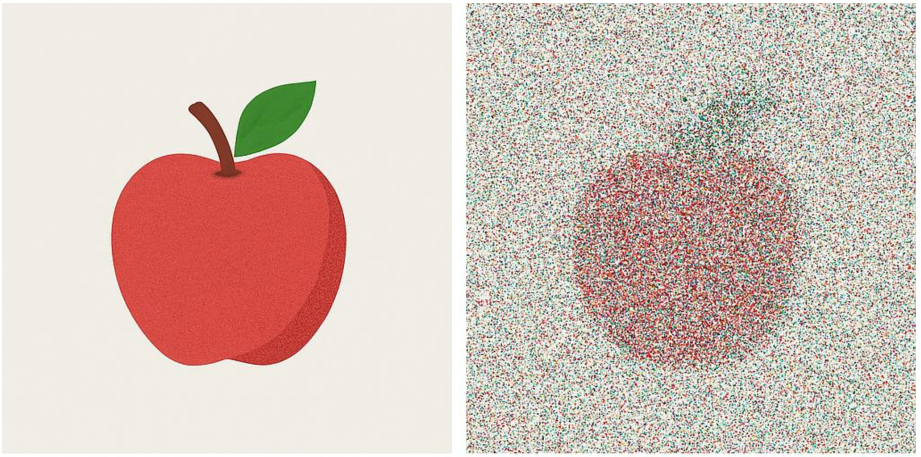
Noise

$$\tilde{f}(Enc(m))$$



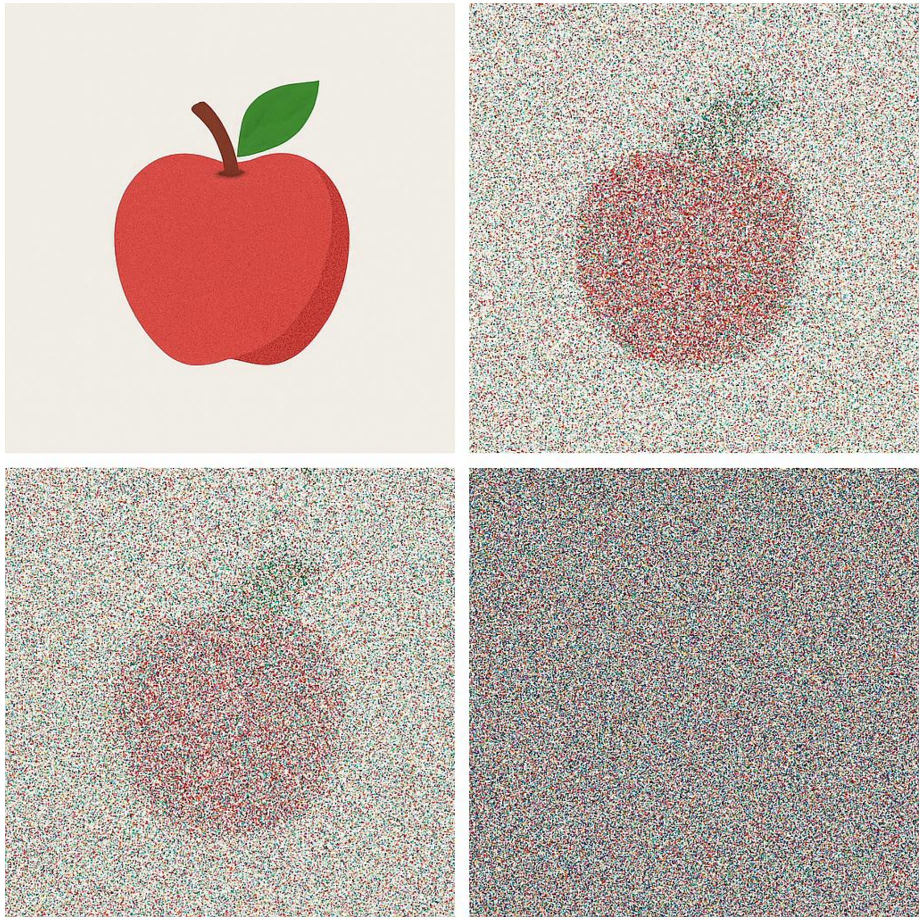
Noise

$$\tilde{f}(\tilde{f}(Enc(m))$$

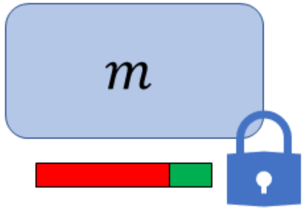


Noise

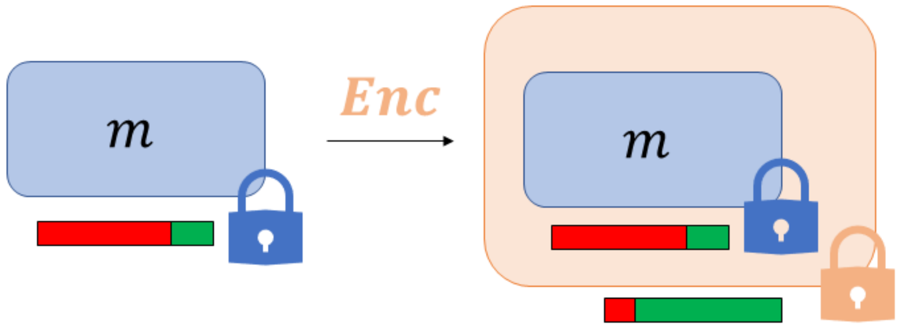
$$\tilde{f}(\tilde{f}(\tilde{f}(\tilde{f}(Enc(m))$$



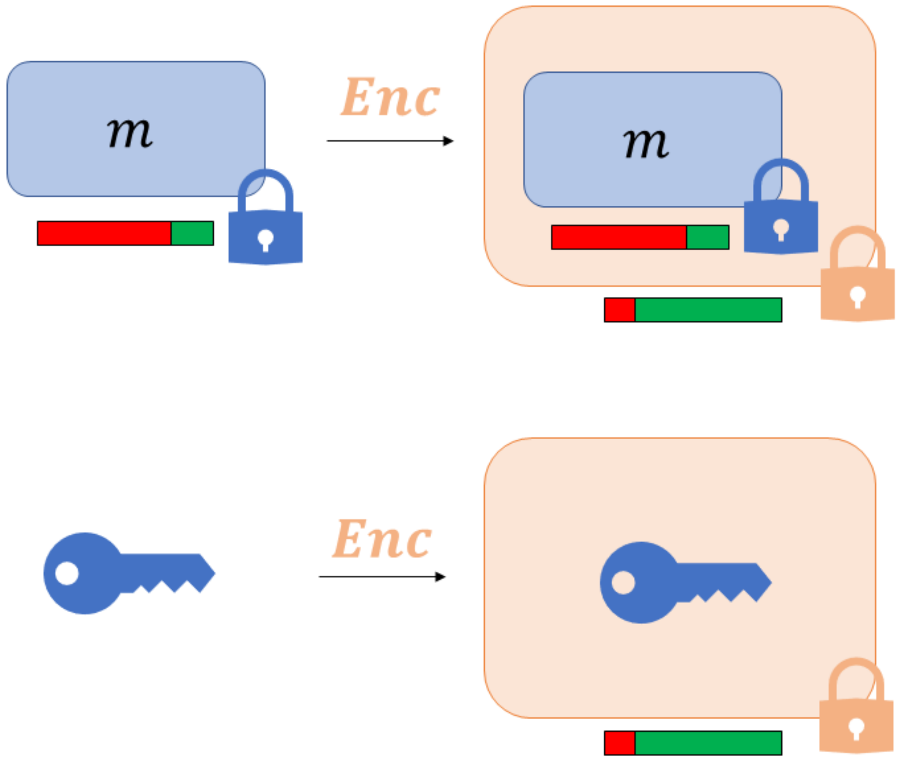
Gentry's breakthrough



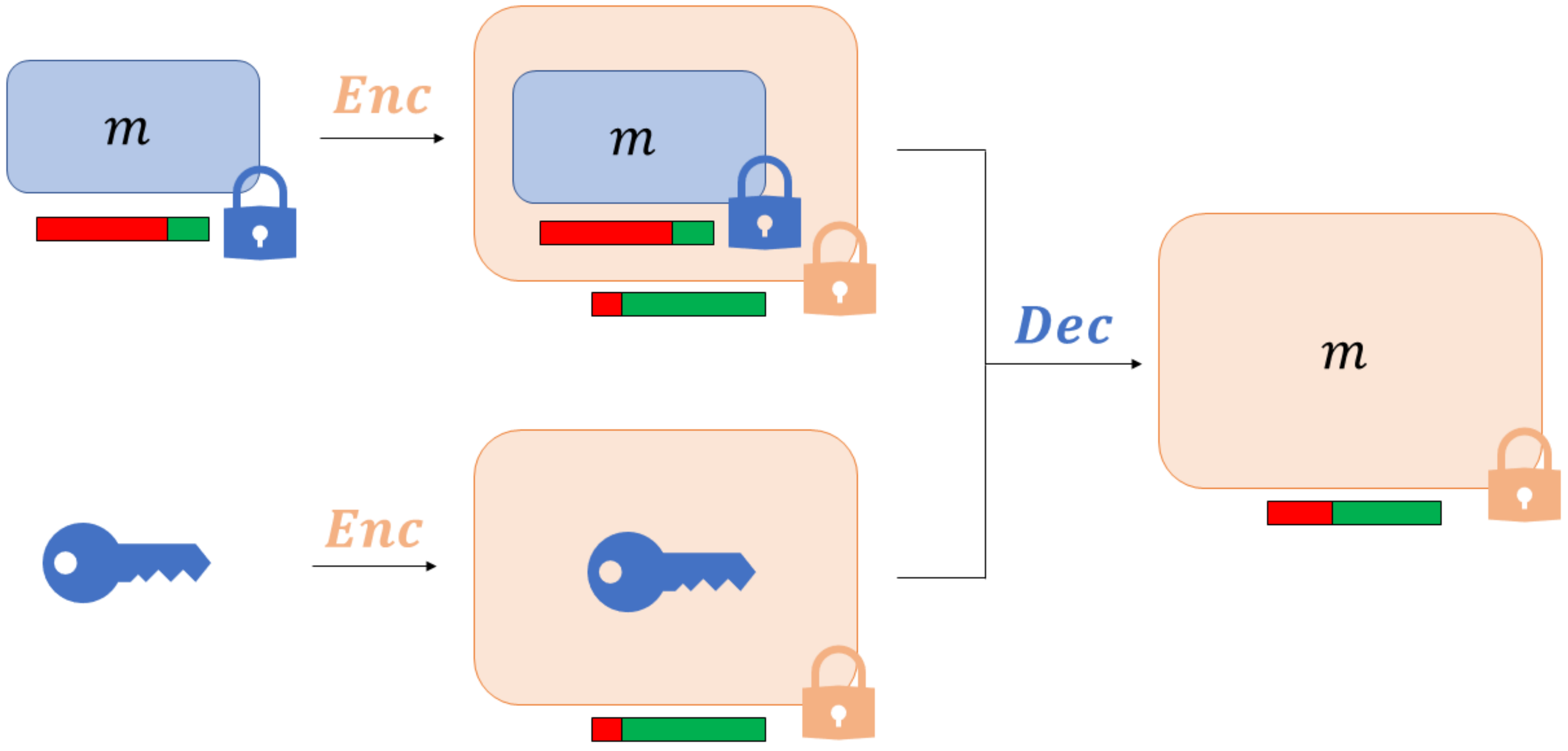
Gentry’s breakthrough



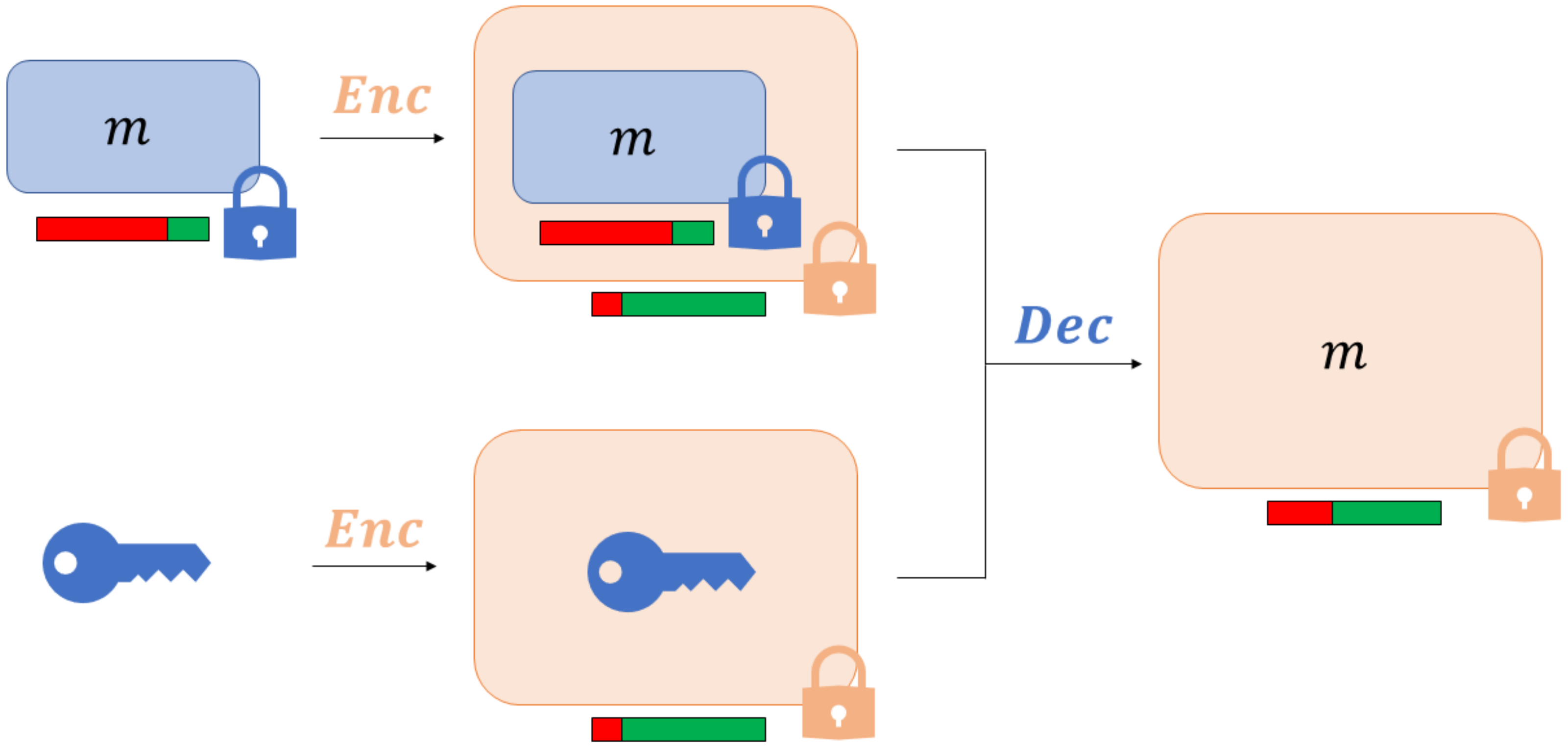
Gentry’s breakthrough



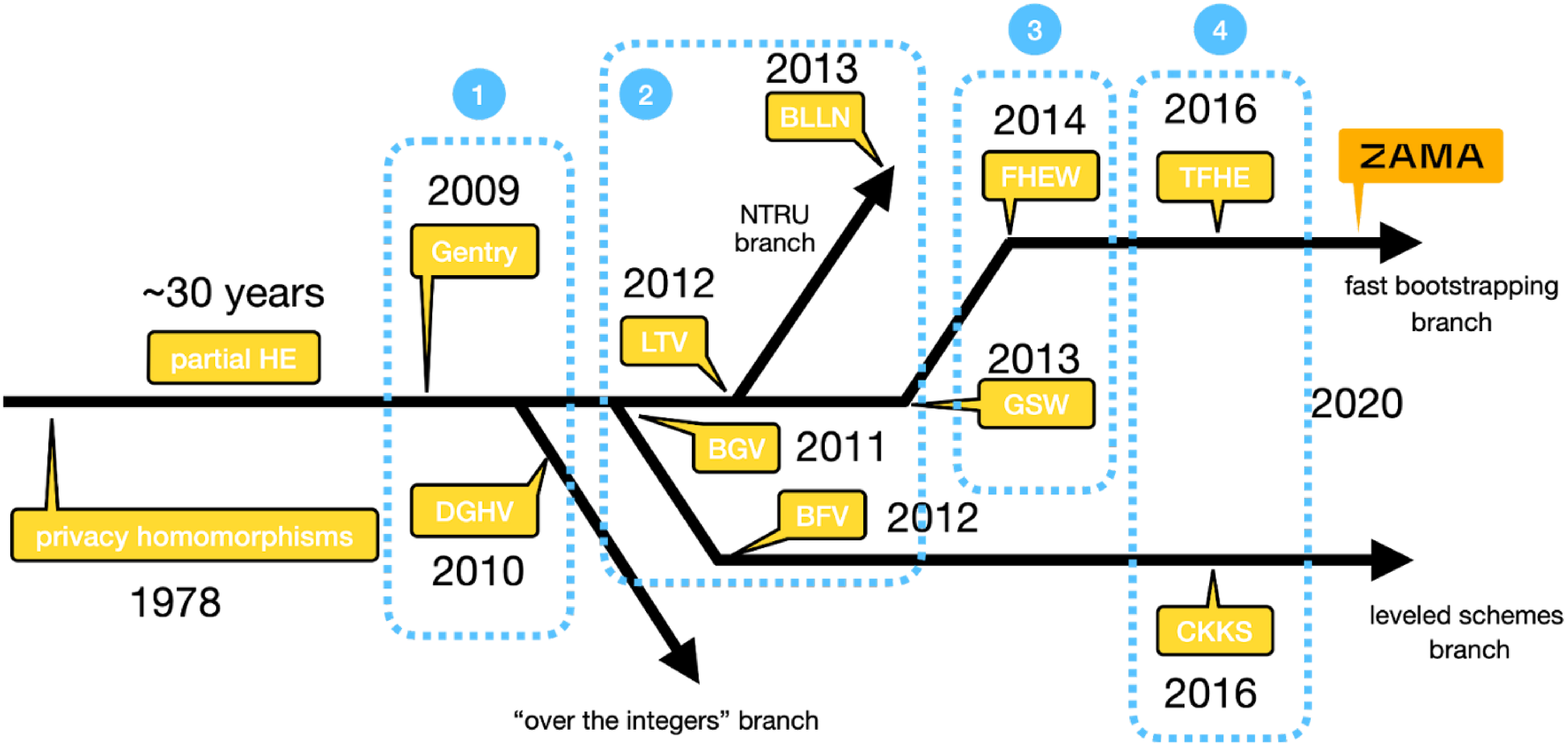
Gentry's breakthrough



Gentry’s breakthrough



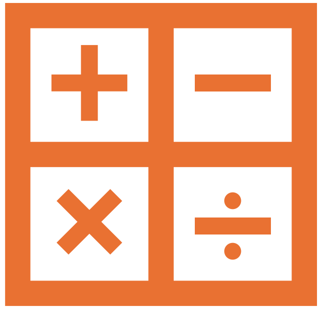
BUT BOOTSTRAPPING IS EXPENSIVE!



1
2
3
4
2009
Gentry
~30 years
partial HE
privacy homomorphisms
1978
DGHV
2010
2012
LTV
NTRU
branch
2013
BLLN
BGV
2011
BFV
2012
2014
FHEW
2013
GSW
2016
TFHE
ZAMA
fast bootstrapping
branch
2020
leveled schemes
branch
CKKS
2016
"over the integers" branch

What $\tilde{f}$ can we evaluate?

Addition



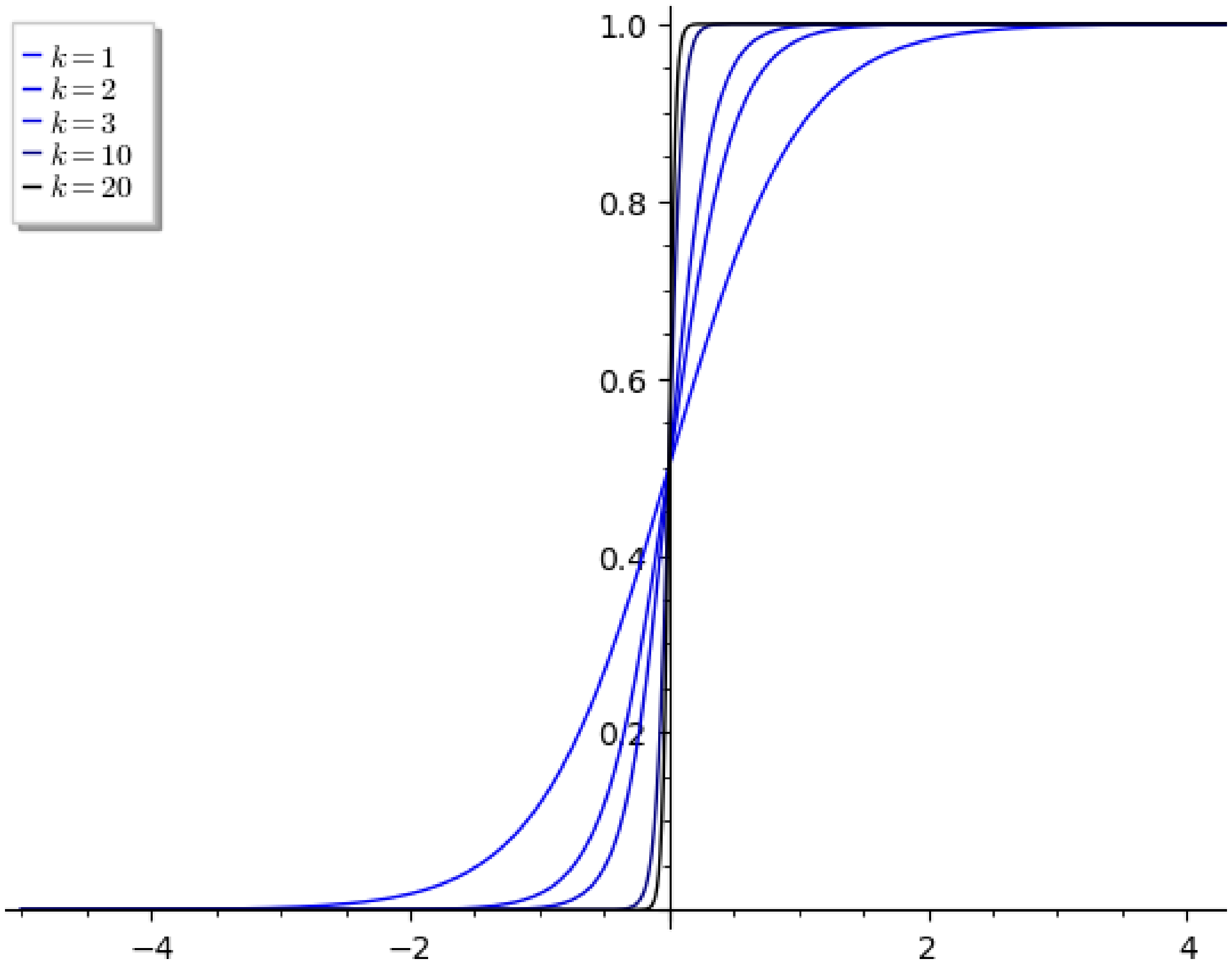
Multiplication

Approximation

- Express another function as a polynomial

- The more accurate we want to be, the more operations we must do

$$\frac{1}{2} + \frac{1}{2}\tanh(kx) = (1 + e^{-2kx})^{-1}$$



Use cases

- AI / machine learning
  - See e.g. OpenAI, DeepSeek, privacy concerns
- Genome sequencing
- Cloud computing
- Blockchain
- And more!

Example— Encrypted Diabetes Prediction

🎯 Goal: Predict risk of diabetes using encrypted patient data

Use Case — Encrypted Diabetes Prediction

🎯 Goal: Predict risk of diabetes using encrypted patient data

📊 Model: Trained logistic regression

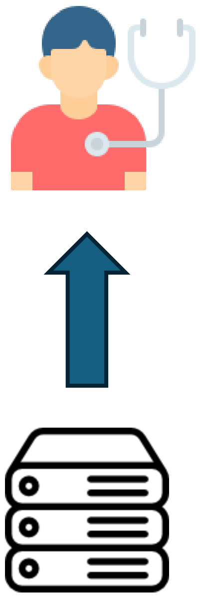
Use Case — Encrypted Diabetes Prediction

🎯 Goal: Predict risk of diabetes using encrypted patient data

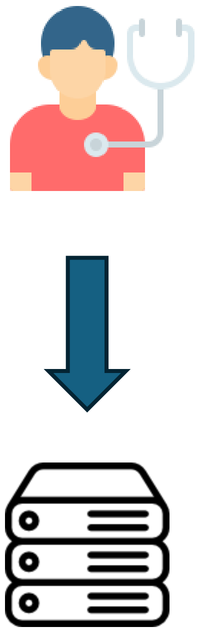
📊 Model: Trained logistic regression

🔐 Data privacy

How?



1. Company sends model to patient



2. Patient sends data to company

Model structure: logistic regression

$\hat{y} = \sigma(w^T x + b)$

Model structure: logistic regression

$$\hat{y} = \sigma(w^T x + b)$$

Model structure: logistic regression

x = input features (e.g., [glucose, BMI, age])

$$\hat{y} = \sigma(w^T x + b)$$

Model structure: logistic regression

w = weights

x = input features (e.g., [glucose, BMI, age])

$$\hat{y} = \sigma(w^T x + b)$$

Step 1: Calculate $w^T x + b$

Patient data $x = [x_1, x_2, x_3]$

1. Encrypt → $Enc(x_1)$, $Enc(x_2)$, $Enc(x_3)$
   - CKKS encryption scheme

2. Compute encrypted dot product:
   - $Enc(z) = \Sigma\, w_i \cdot Enc(x_i) + b$
   - Weights w_i in plaintext

w = weights

x = input features (e.g., [glucose, BMI, age])

$$\hat{y} = \sigma(w^T x + b)$$

σ(z) = 1 / (1 + e^(-z))

Step 3: Calculate Sigmoid function

1. Approximate sigmoid with a polynomial:
   - $P(z) = 0.5 + 0.197z - 0.004z^3$

2. $Enc(\hat{y}) = P(Enc(z))$

3. Server sends back $Enc(\hat{y})$

4. Patient decrypts locally:
   - $\hat{y} \approx 0.87$ → High risk of diabetes

Want to Try FHE Yourself?

Here are some beginner-friendly tools to explore:

✅ Python-based: Concrete ML, TenSEAL (for ML)

💻 C++ powerhouses: SEAL, OpenFHE, Helib (General uses)

🧪 Niche/experimental: TFHE, CUFHE, Lattigo (Speedups etc.)

Acknowledgements

- Zama (figure slide 16)
- Thom Sijpesteijn (diagrams slides 1-4, 10-14)
- Sam Leder (diagrams slides 1-4, 10-14)