Parallel Homomorphic Encryption

Seny Kamara – Microsoft Research

Mariana Raykova – IBM Research

Big Data

The scale of data we create is growing rapidly

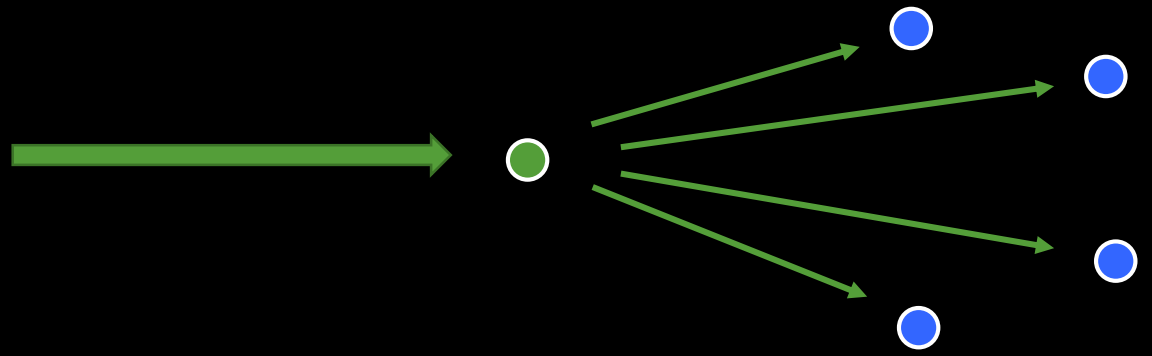
- Walmart: 2.5 petabytes of transaction data per day
- Jets: 10 terabytes of sensor data per 30 mins of flight
- Large Hadron Collider: 40 terabytes per second

How do we process this data?

Too much for any single machine (even supercomputer)

Clusters of machines

Cluster Computing



Distribute data

Synchronization

Fault tolerance

Parallel algorithms

MapReduce [Dean-Ghemawat04]

A framework
- Distributed file system
- Fault tolerance
- Synchronization

A model for parallel computation
- easy to design parallel algorithms

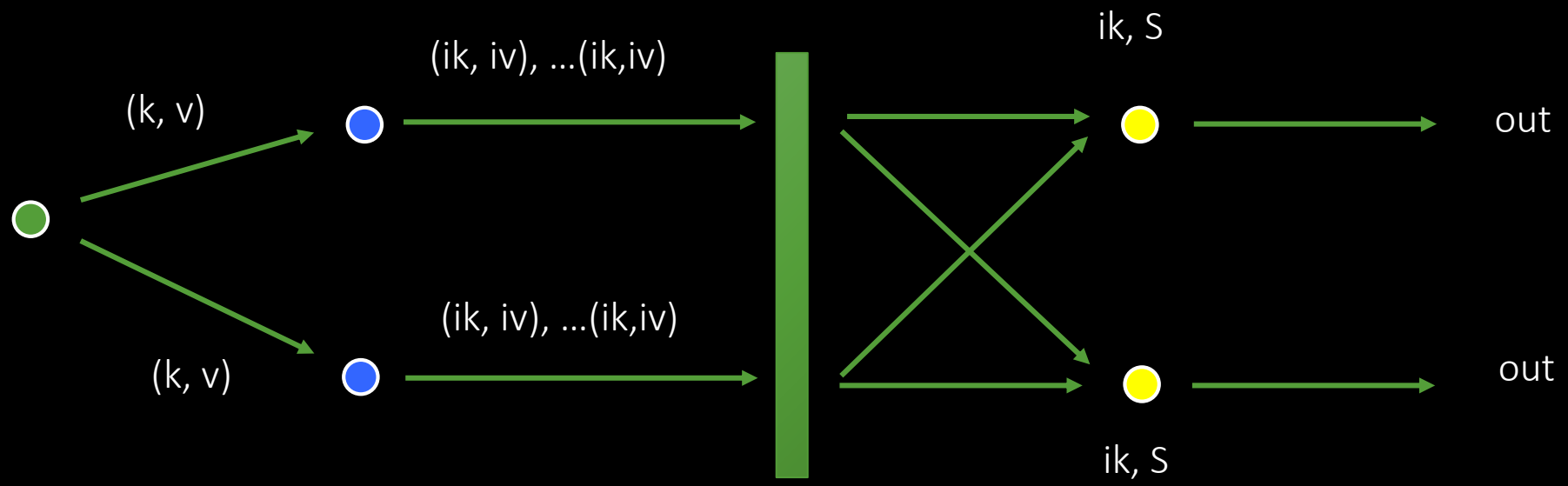
Standard for processing Big Data

MapReduce [Dean-Ghemawat04]

MapReduce program

Map(k_i, v_i) → (ik_1, iv_1), …, (ik_t, iv_t)

Reduce(ik_i, S_i) → out_i

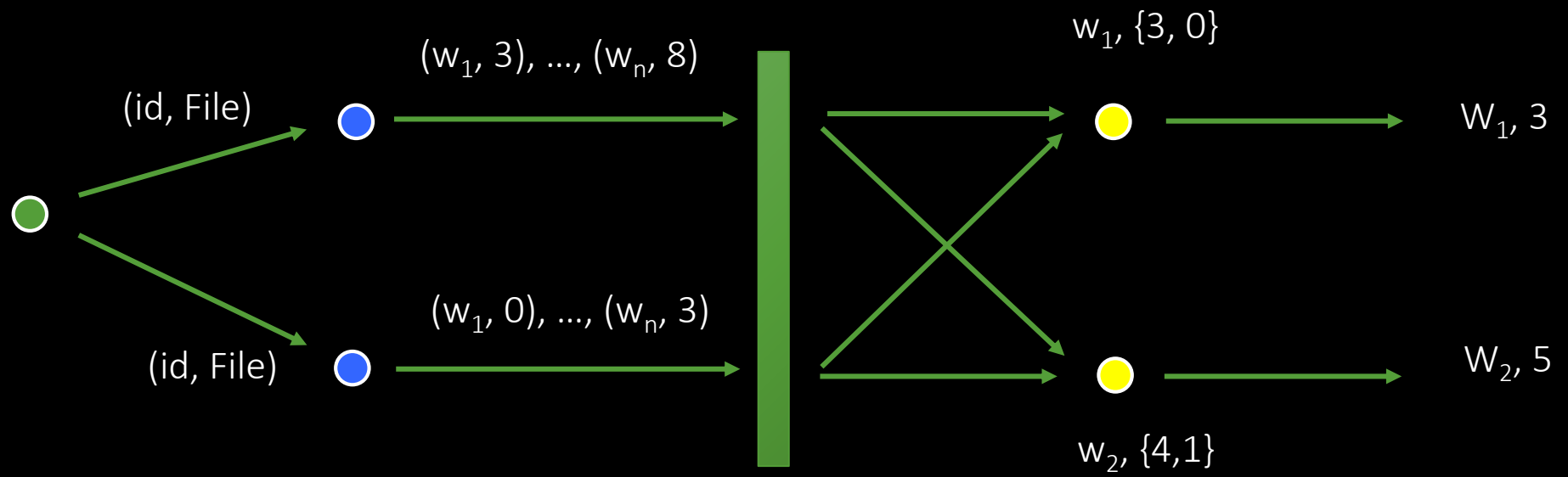


MapReduce [Dean-Ghemawat04]

MapReduce algorithm

Map(k_i, v_i) → (ik_1, iv_1), …, (ik_t, iv_t)

Reduce(ik_i, S_i) → out_i



MapReduce

Many MapReduce algorithms

- IR: counts, searching, sorting, pagerank, HITS, …
- ML: PCA, neural networks, regression, support vector machines, …
- Graphs: BFS, DFS, pagerank, minimum spanning tree, …

The Big Data Stack

Pig, ...

analytics languages

HBase, Hive, Hadapt, ...

databases (SQL & NoSQL)

Hadoop, MapR, Hortonworks, Cloudera, ...

MapReduce frameworks

Amazon Elastic MapReduce, Azure HDInsight

Cloud-based MapReduce

What if I don’t trust the Cloud?

MapReduce on Encrypted Data?

Use homomorphic encryption!

- Client encrypts data
- Cluster computes homomorphically

Question?

- Can homomorphic evaluation be done in parallel?
- Can it be done on a standard MapReduce cluster?

Parallel Homomorphic Encryption

PHE = (Gen, Enc, Eval, Dec)

Gen(1^k)

Enc(K, m)

Eval(f, c_1, …, c_n) ≈ MapReduce algorithm

Dec(K, c)

PHE = (Gen, Enc, Parse, Map, Reduce, Merge, Dec)

Parse(c) generates (encrypted) key-value pairs for mappers

Map(k, v) homomorphically evaluates map algorithm

Reduce(ik, S) homomorphically evaluates reduce algorithm

Security

CPA-security

Adversary cannot learn any information about message from ciphertext

Note

Here single-input security is enough

Constructions

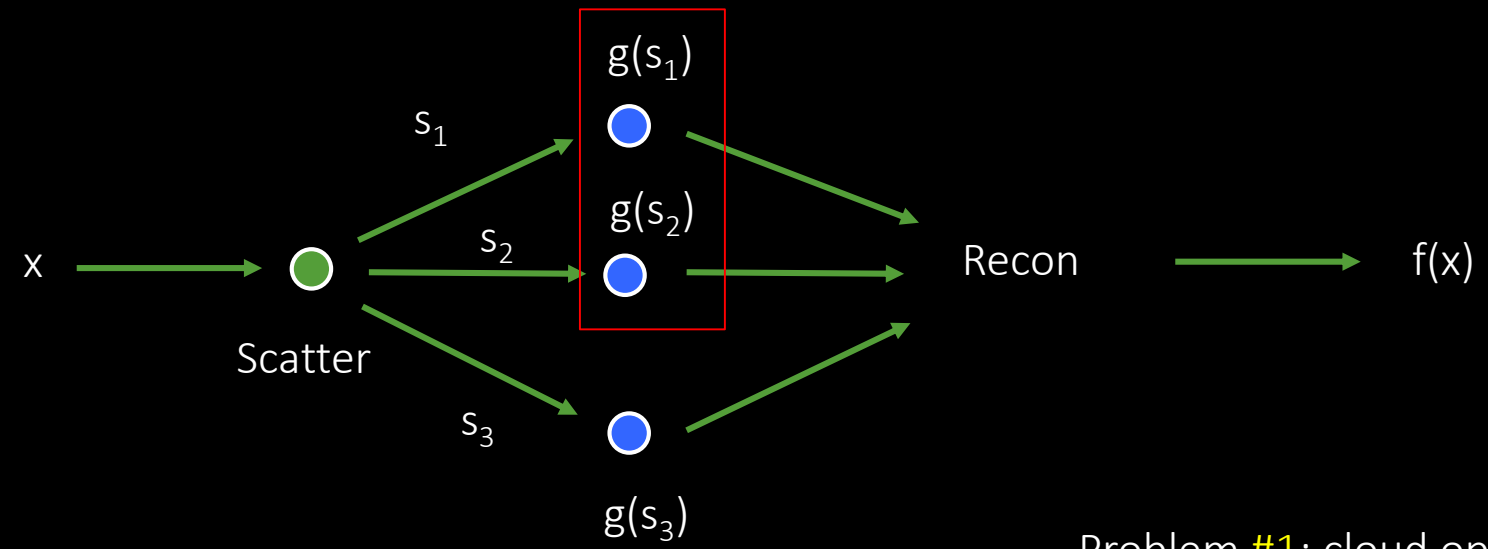
A High-Level Framework

PHE = Randomized reductions + homomorphic encryption

Randomized reductions [Beaver-Feigenbaum90, Beaver-Feigenbaum-Killian-Rogaway97]

(Scatter, Recon) is RR from f to g if

A High-Level Framework



Problem #1: cloud operates all workers

Problem #2: Recon can be expensive

Solutions

Randomized reduction with t = n

- Univariate polynomials
- Multivariate polynomials

Outsource Recon

- Simple enough to be evaluated with single multiplication

Reduction for Univariate Polynomials

$\text{Scatter}_q(x)$

Set $n = 2q+1$

Sample $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ at random in F_q^n (all distinct)

Choose degree-2 *permutation* polynomial P_x such that $P_x(0) = x$

Set $\mathbf{s} = (s_1, \ldots, s_n) = (P_x(\alpha_1), \ldots, P_x(\alpha_n))$

Output $\mathbf{s}$ and $st = \boldsymbol{\alpha}$

$\text{Recon}_q(st, y_1, \ldots, y_n)$

Interpolate Q through points $(\alpha_1, y_1), \ldots, (\alpha_n, y_n)$

Output $Q(0)$

Reduction for Univariate Polynomials

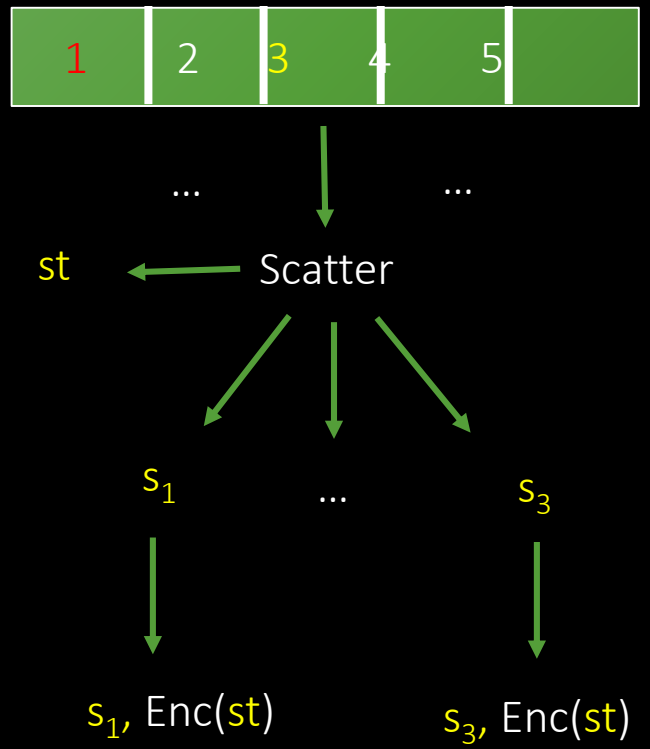
Correctness

- Secret sharing is “homomorphic”
- Interpolation of $Q(p_x(\alpha_1)), \ldots, Q(p_x(\alpha_n))$ at 0 results in $Q(p_x(0)) = Q(x)$

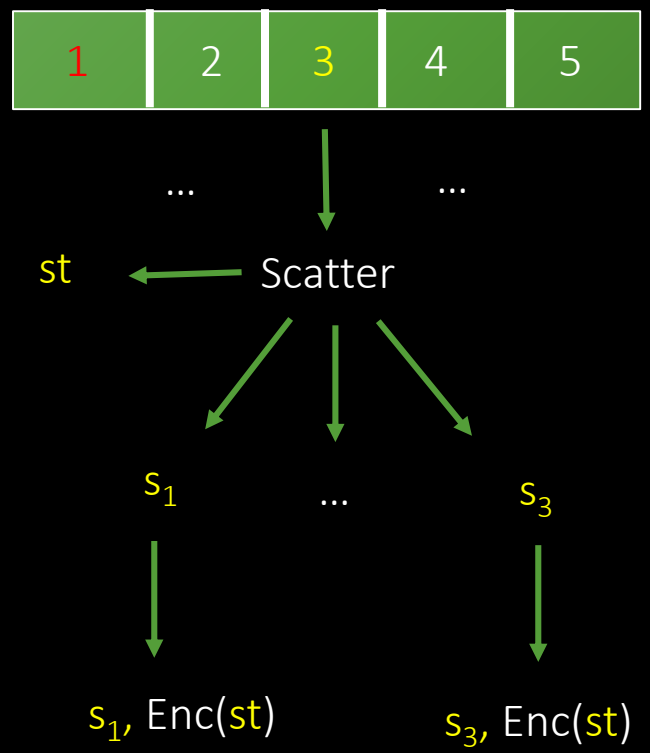
Security

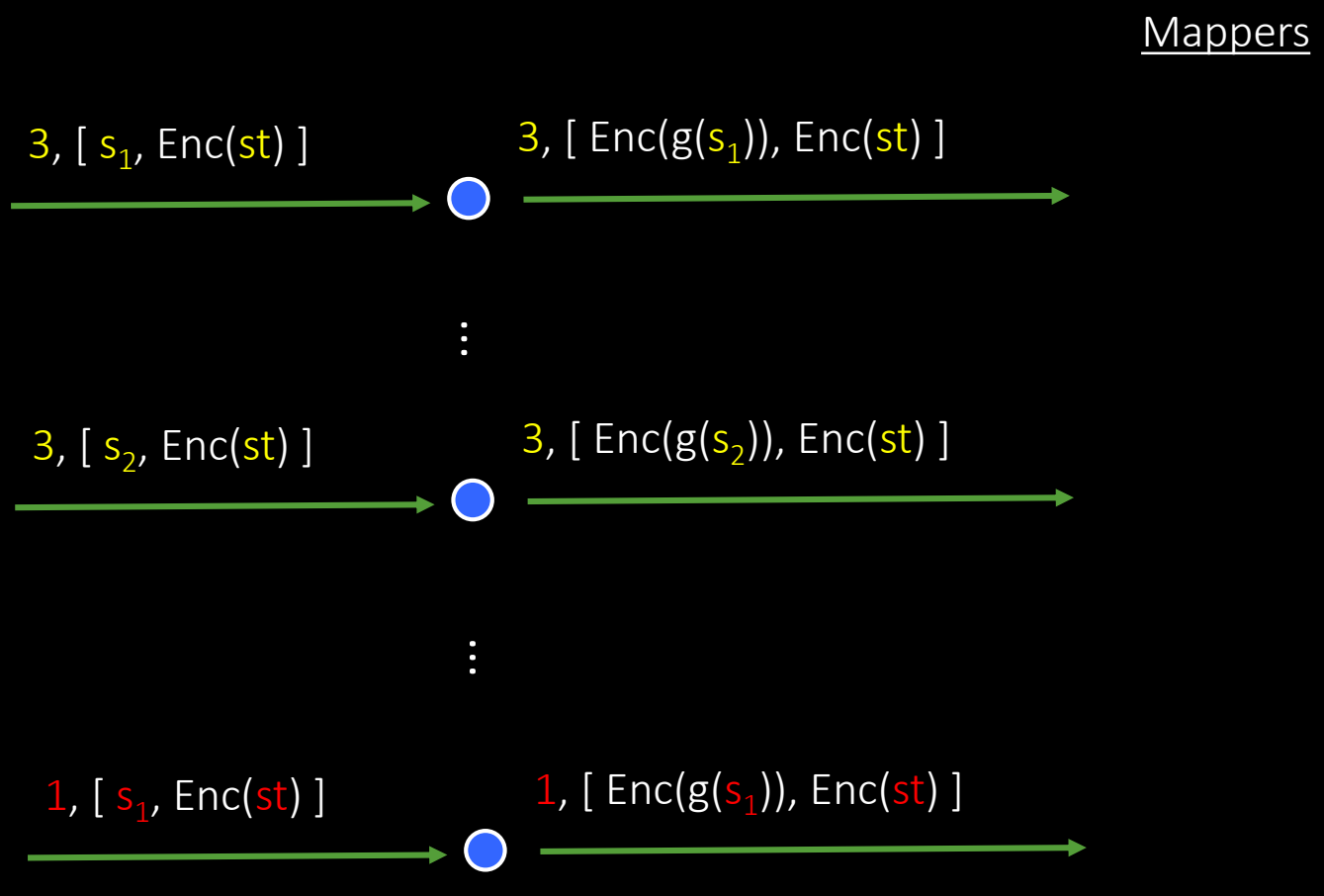
- Sharing polynomials are permutations
- Evaluation points α_i are uniform
- Shares are independent of secret

A General MR-Parallel HE Scheme

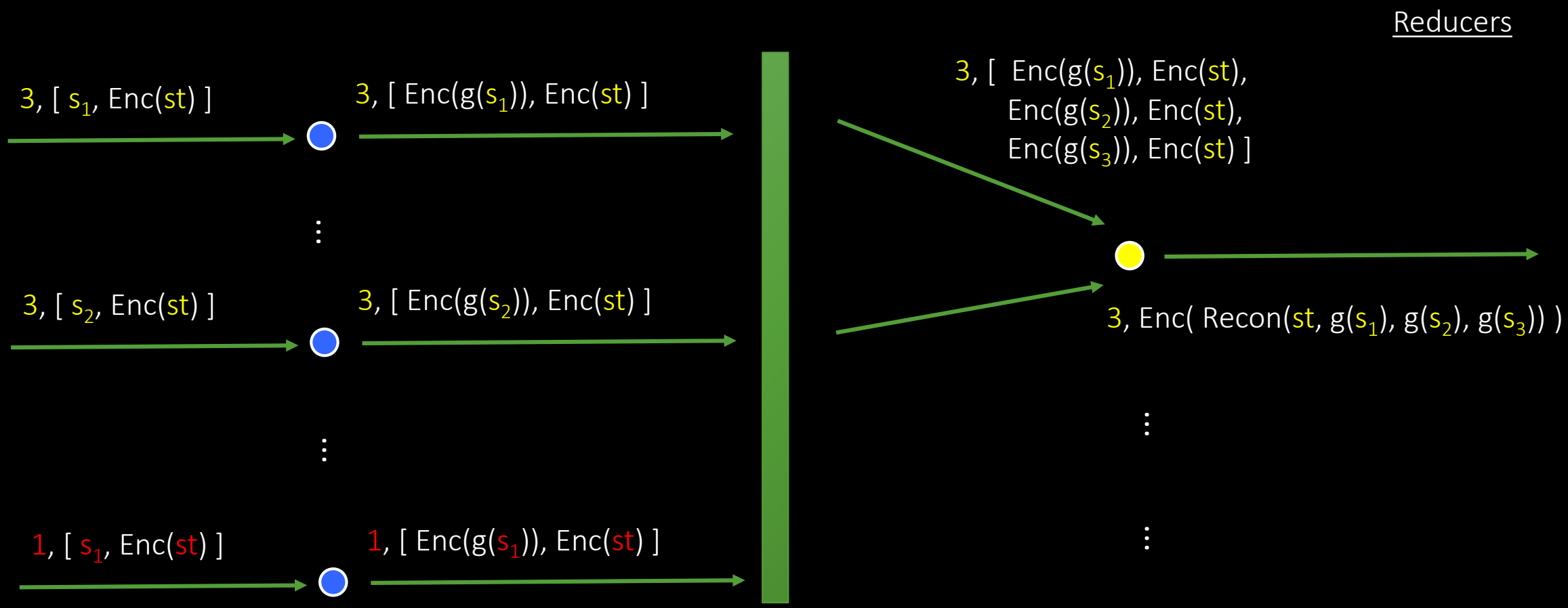


A General MR-Parallel HE Scheme





A General MR-Parallel HE Scheme



Additional Results

Randomized reduction for multivariate polynomials

- for small number of variables
- based on multi-dimensional noisy curve reconstruction assumption
  - from [Ishai-Kushilevitz-Ostrovsky-Sahai06]

More efficient direct MR-PHE constructions

- Univariate polynomials
- Multivariate polynomials

Applications

- Database search (e.g, keyword search, OR queries)

Thanks!