



University of Siena,  
Italy



National Research Council  
of Italy, Rome

## Parallel Implementation of GC-Based MPC Protocols in the Semi-Honest Setting

Barni, Bernaschi, Lazzeretti, Pignata, Sabellico

# Outline

---

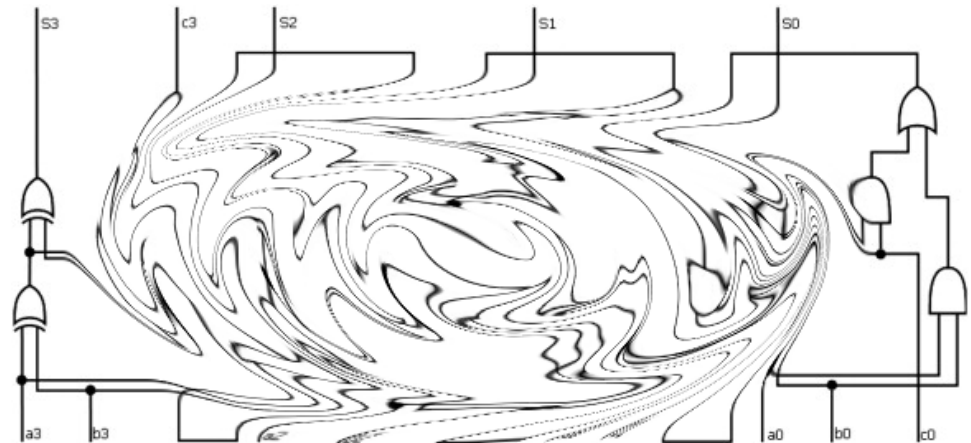
- ▶ Introduction
- ▶ GC parallelization
- ▶ Two different implementations
  - ▶ Fine-grained parallelization
  - ▶ Coarse-grained parallelization
- ▶ Application examples
- ▶ Results



# Garbled Circuits [Yao86]

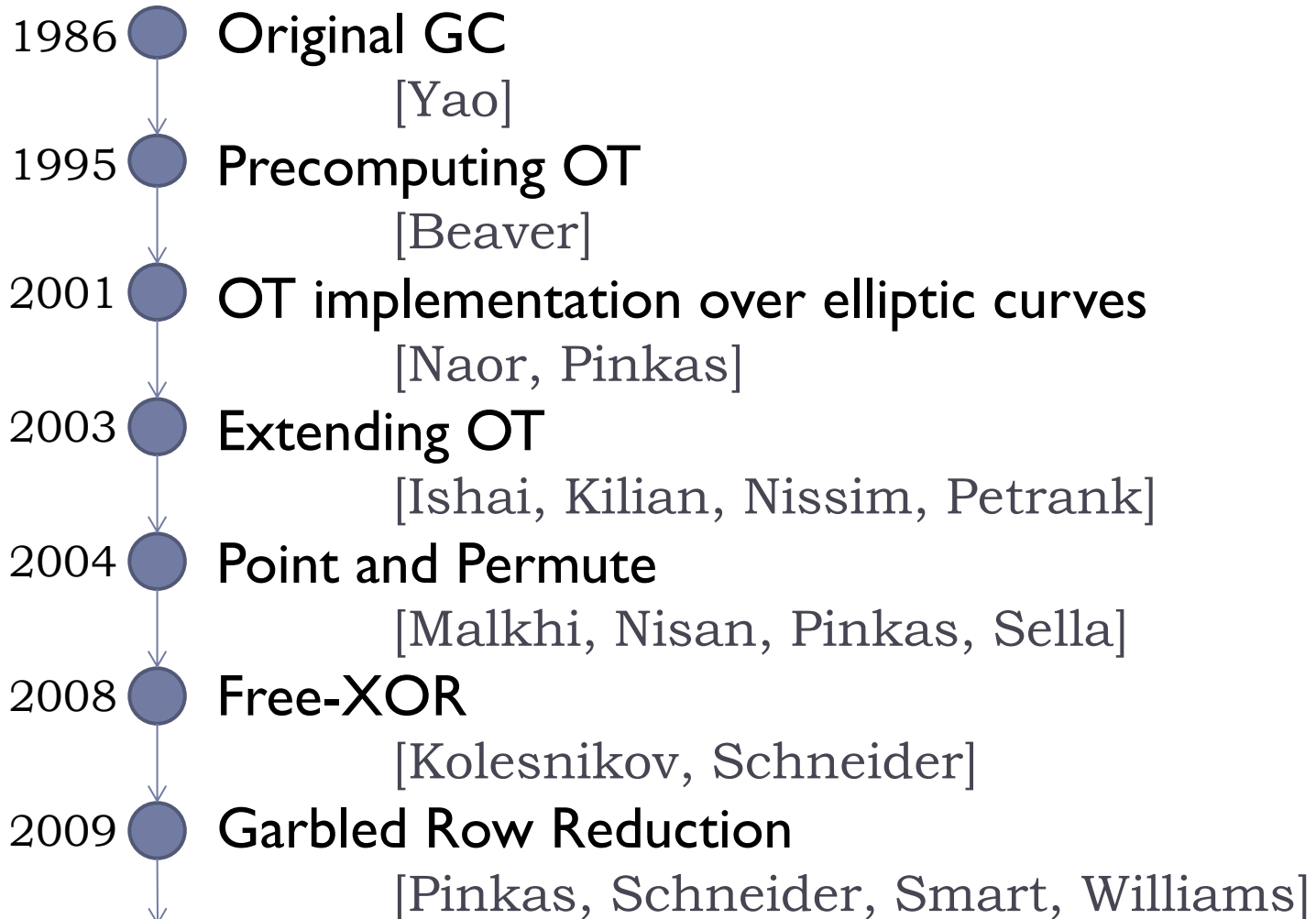
---

- ▶ Powerful MPC tool
- ▶ Permits to evaluate any  $f(x,y)$ , represented by a boolean circuit, on private inputs
  
- ▶ Applied to
  - ▶ Auctions
  - ▶ Medical scenarios
  - ▶ Biometric identification
  - ▶ ...



# Previous GC improvements

---



# Motivation

---

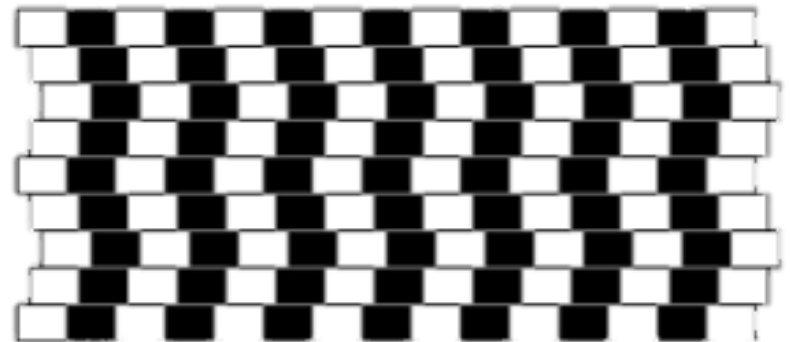
- ▶ Boolean circuits have a lot of gates that can be evaluated in parallel
- ▶ Many actual systems are suitable for parallel computation
  - ▶ Multi-core CPUs
  - ▶ Graphic Processing Units
  - ▶ Multi-processors servers
- ▶ Other works
  - ▶ Parallel implementation of particular operation  
[*Pu, Duan, Liu 2011*]
  - ▶ GPUs for malicious setting  
[*Frederiksen, Nielsen, 2013*]
- ▶ Our contribution:
  - ▶ Two parallel implementations of GC
  - ▶ Analysis



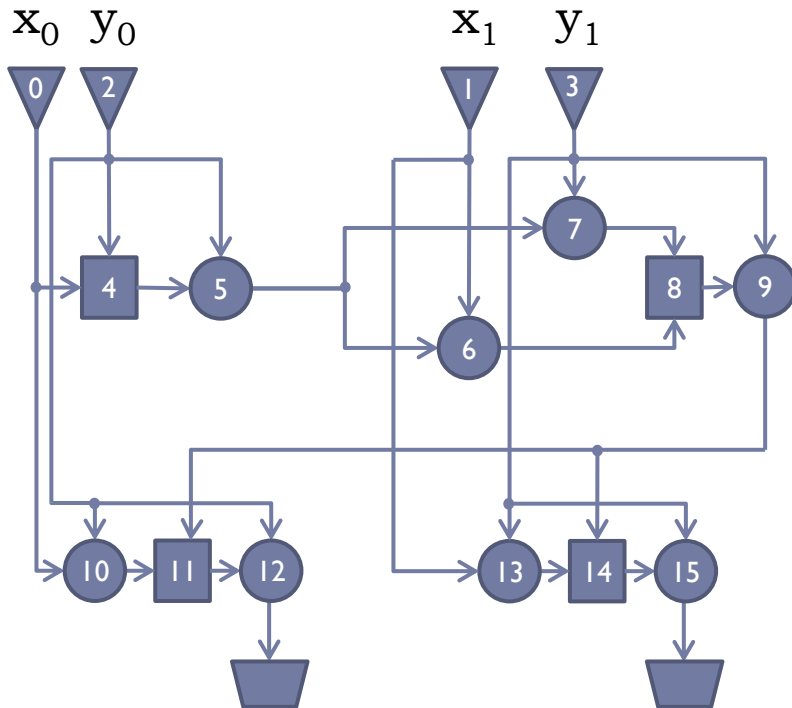
# Fine grained parallelization

---

- ▶ Parallelization of single gates
- ▶ Can be applied to any circuit
- ▶ No special attention during circuit design
  
- ▶ Circuit gates subdivided in layers
- ▶ Parallelization performed by a parser
  - ▶ Parallelized circuit
    - ▶ Sorted gates
    - ▶ Can be also evaluated sequentially
  - ▶ Additional information
    - ▶ Number of gates in each layer

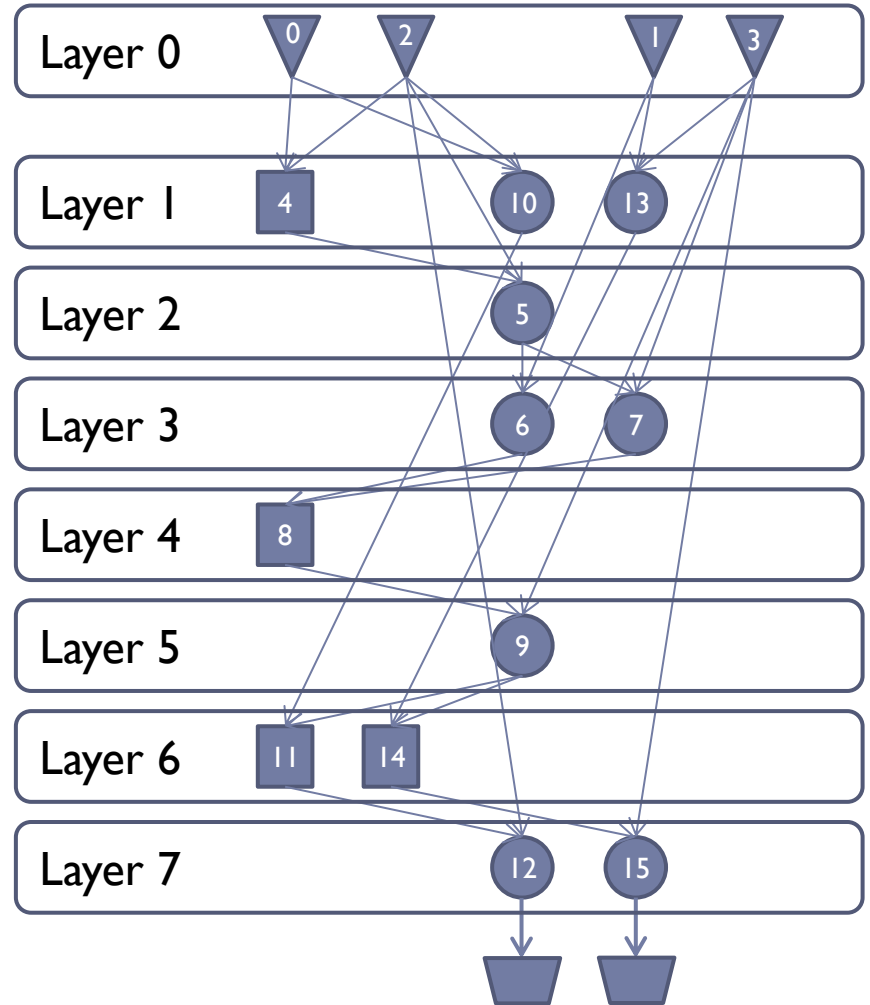


# Circuit parallelization



**General rule:**

A gate having inputs coming from gates respectively in layers  $i$  and  $j$  is placed in layer  $\max(i, j) + 1$



# Parser outputs

## Sorted circuit

```
520 5 440 1|
v u "distance_0" 16 0 1 2 3 4 5 6 7 8 9 10
v u "distance_1" 16 16 17 18 19 20 21 22 23
v u "distance_2" 16 32 33 34 35 36 37 38 39
v u "distance_3" 16 48 49 50 51 52 53 54 55
v u "distance_4" 16 64 65 66 67 68 69 70 71
x n 113 47 63
x n 112 46 62
x n 111 45 61
x n 110 44 60
x n 109 43 59
x n 108 42 58
x n 107 41 57
x n 106 40 56
x n 105 39 55
x n 104 38 54
x n 103 37 53
x n 102 36 52
x n 101 35 51
x n 100 34 50
x n 99 33 49
x n 98 32 48
x n 96 15 31
x n 95 14 30
x n 94 13 29
x n 93 12 28
x n 92 11 27
```


## Additional information

```
147 2
1 1 : 0 32 2
1 2 : 0 2 0
1 3 : 0 4 0
1 4 : 0 0 2
1 5 : 0 2 0
1 6 : 0 4 0
1 7 : 0 0 2
1 8 : 0 2 0
1 9 : 0 4 0
1 10 : 0 0 2
1 11 : 0 2 0
1 12 : 0 4 0
1 13 : 0 0 2
1 14 : 0 2 0
1 15 : 0 4 0
1 16 : 0 0 2
1 17 : 0 2 0
1 18 : 0 4 0
1 19 : 0 0 2
1 20 : 0 2 0
1 21 : 0 4 0
1 22 : 0 0 2
1 23 : 0 2 0
1 24 : 0 4 0
1 25 : 0 0 2
1 26 : 0 2 0
```



# Fine-grained execution

---

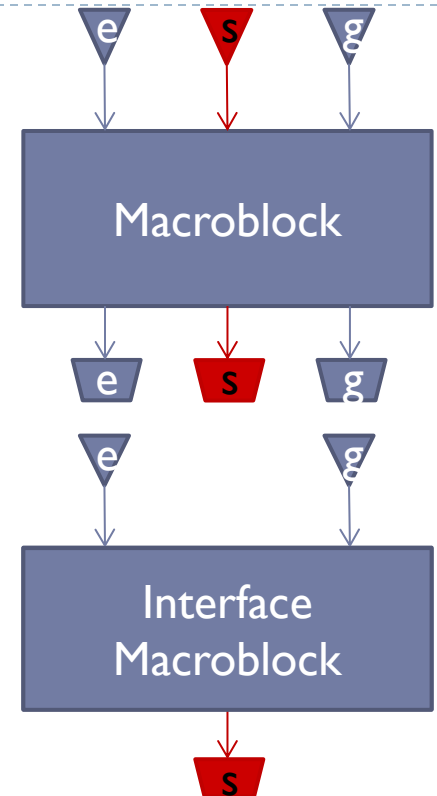
- ▶ Gates in the same layer are assigned to different threads
- ▶ New layer processed when previous one is completely elaborated
- ▶ Separate management for NOT, XOR and non-XOR gates
  - ▶ XOR gates have low complexity
  - ▶ Circuits usually composed by ~75% of them
    - ▶  High benefits from XOR parallelization
- ▶ High overhead introduced by thread management

# Coarse-Grained Parallelization

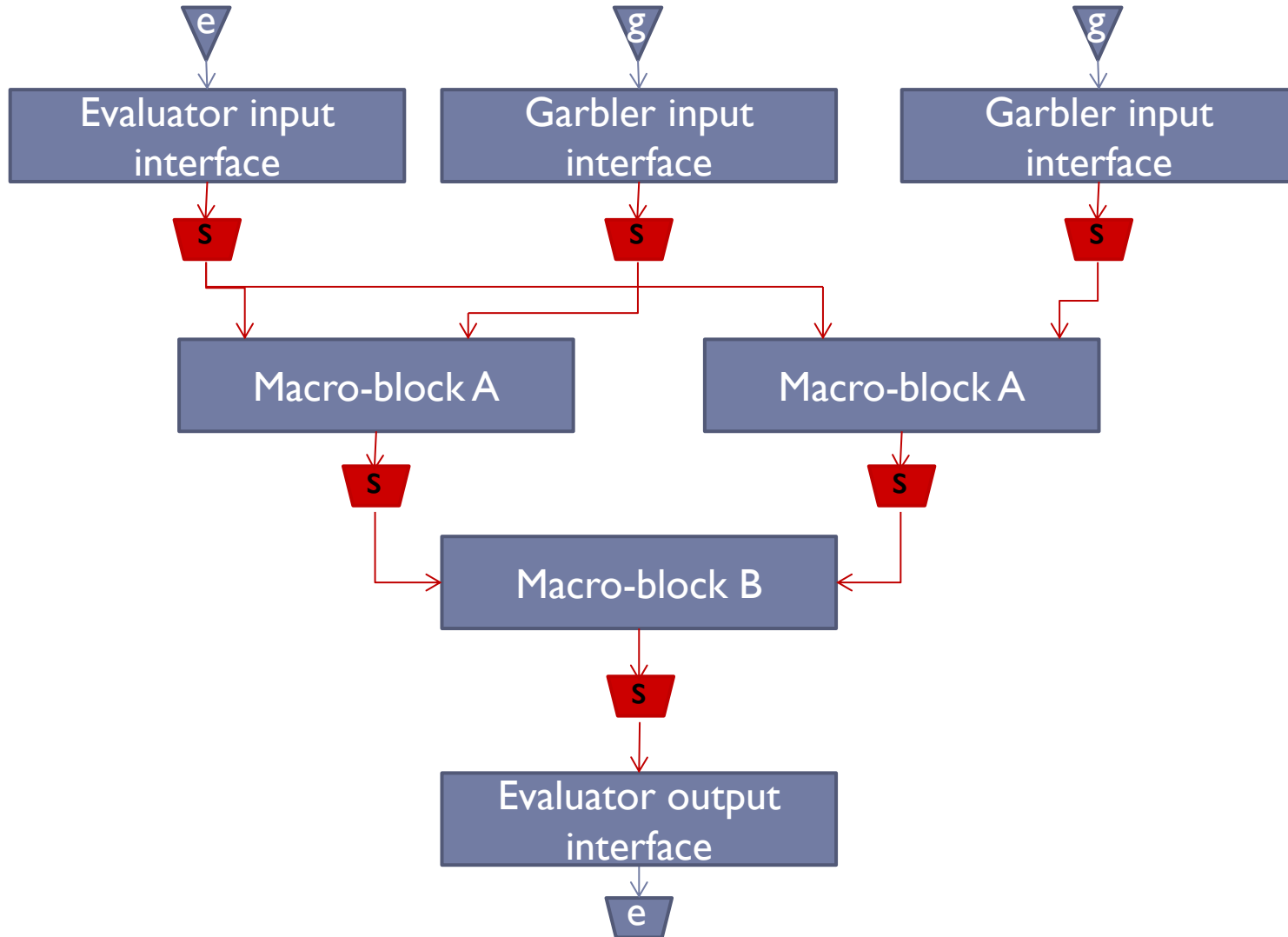
- ▶ Parallelization of macro-blocks
- ▶ Different design strategy
  - ▶ A file for each macro-block
  - ▶ Easier circuit design
  - ▶ Interface between macro-blocks needed
    - ▶ New secret type for input and output

- ▶ Suggestion:

- ▶ Use of macroblocks also for input and output management
- ▶ Conversion of plain inputs into associated secrets implemented by one or more macroblocks



# Composition of macroblocks



# Execution

---

- ▶ **Garbling**
  - ▶ Same  $\Delta = s_0 \oplus s_1$  used in all the circuits
  - ▶ Secret input pairs are not randomly generated
    - ▶ Forced to be equal to secret output pairs obtained by previous blocks
- ▶ **Evaluation**
  - ▶ Secrets obtained as output are stored to be used later
  - ▶ Secrets used inside the block can be erased
- ▶ Different instances of the same block garbled/evaluated independently in parallel
- ▶ Garbling/evaluation of instances of the same block can be driven together
- ▶ Time saved for loading circuit description
  - ▶ One file reading for all the instances of the same block
  - ▶ Reduced circuit description size
- ▶ Single macro-blocks can be processed by using fine-grained parallelization

# Security

- ▶ Semi-honest model
- ▶ Provided by GC protocol
- ▶ Fine-grained implementation
  - ▶ Gates are only permuted
  - ▶ Evaluator and Garbler view identical to sequential implementation
- ▶ Coarse-grained implementation
  - ▶ Evaluator and Garbler view is equal to the one provided by a single circuit obtained composing the macro-blocks



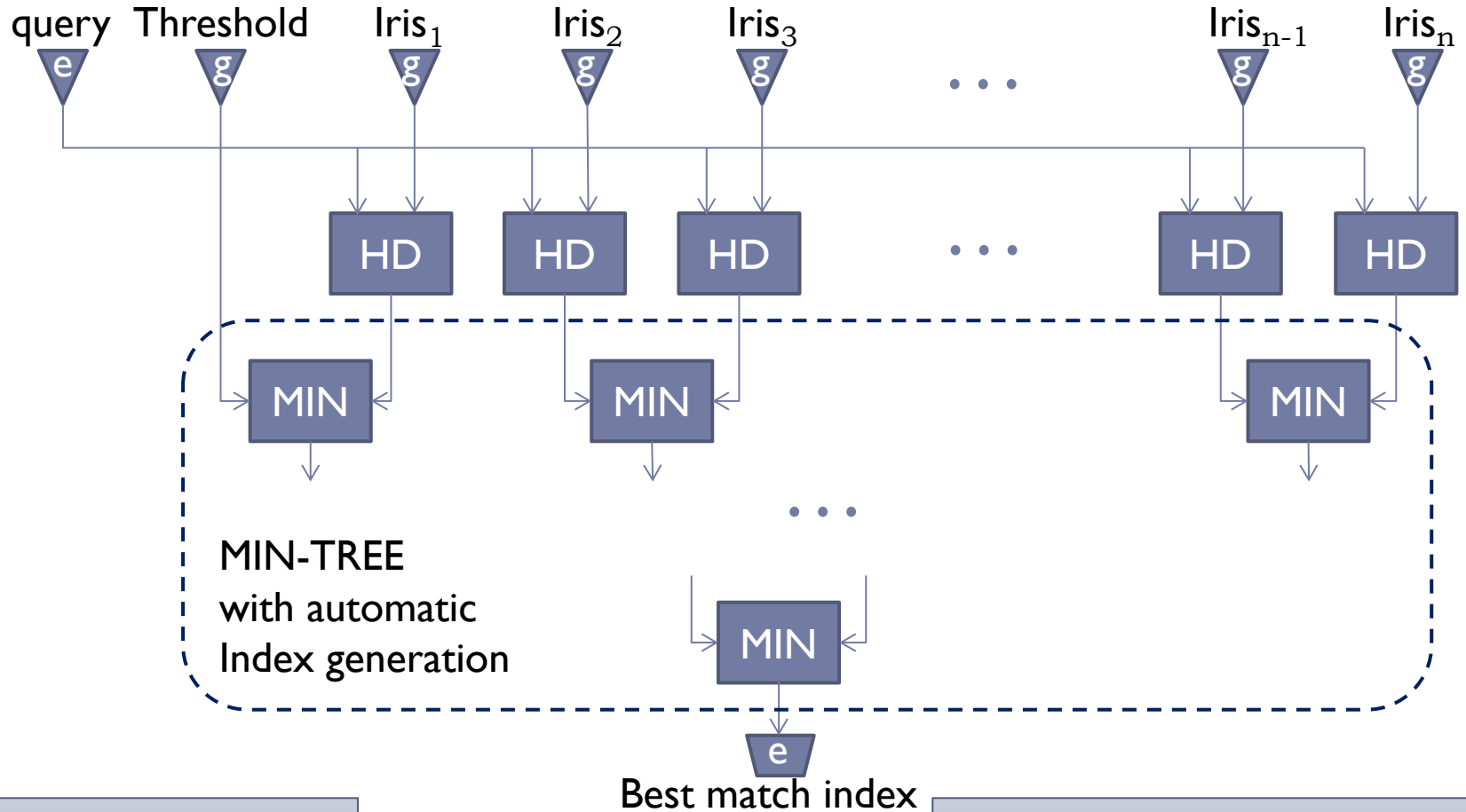
# Performance analysis

---

- ▶ Two application scenarios
  - ▶ Iris Identification
    - ▶ High parallel nature
    - ▶ Output: index of the best match, if exceeding a given threshold
  - ▶ AES encryption
    - ▶ Comparison with previous works
    - ▶ Multiple parallel AES encryption
- ▶ System configuration
  - ▶ Two Intel Xeon E5-2609@2.4GHz
    - ▶ 10Mb cache
    - ▶ 4 cores each
  - ▶ 16 GB RAM
  - ▶ Connected to 100Mb/s lan
- ▶ OT precomputation performed independently from the application
  - ▶ 1 million OTs precomputed in 5 seconds



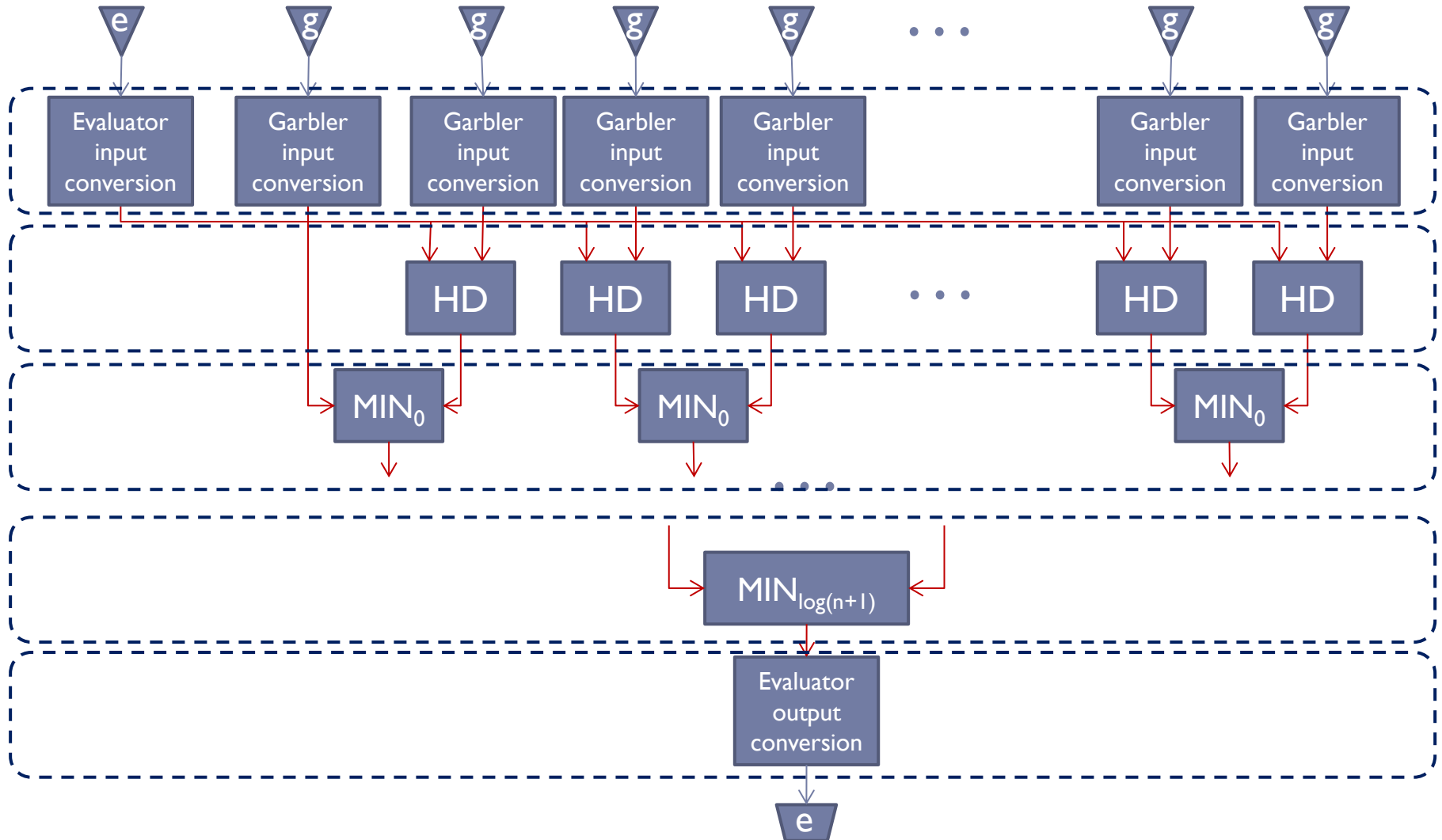
# Iris identification



Parameters:  
1023 irises in the DB  
2048 bits for each iris

Single circuit:  
6.3 M gates (1M non-XOR gates)  
parallelizable in 356 layers

# Iris identification (macroblocks)



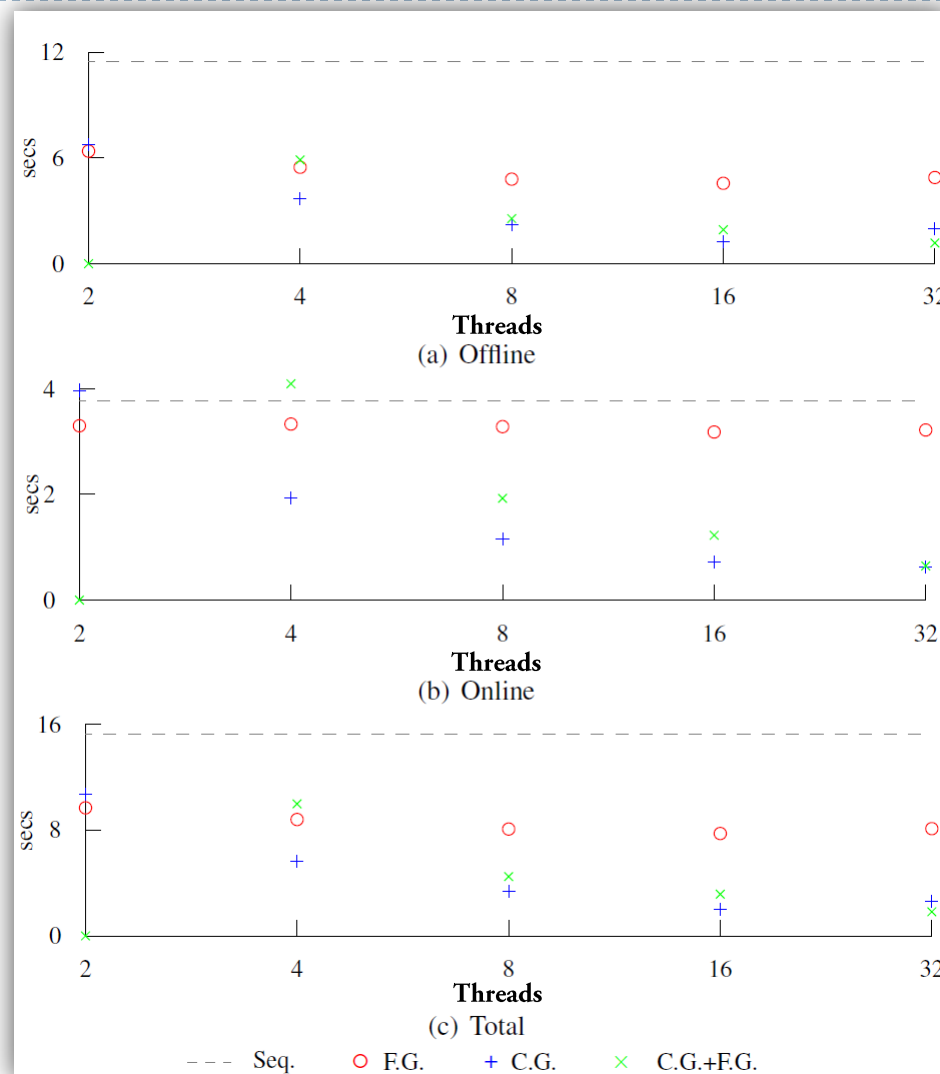


# Iris Identification performance (8 threads)

---

	Phase	Sequential	Fine-Grained	Coarse-Grained	Fine Grained + Coarse Grained
Offline	Garbling	9.772	3.475	2.175	1.860
	OT precomputation	0.010	0.010	0.010	0.010
	Garbled tables transmission	1.701	1.314	0.036	0.690
Online	Garbler's secret transmission	0.338	0.378	0.130	0.158
	Evaluator's secret transmission	0.002	0.003	0.002	0.002
	Evaluation	3.437	2.899	1.019	1.765

# Iris Identification performance (8 threads)



# Oblivious AES Encryption

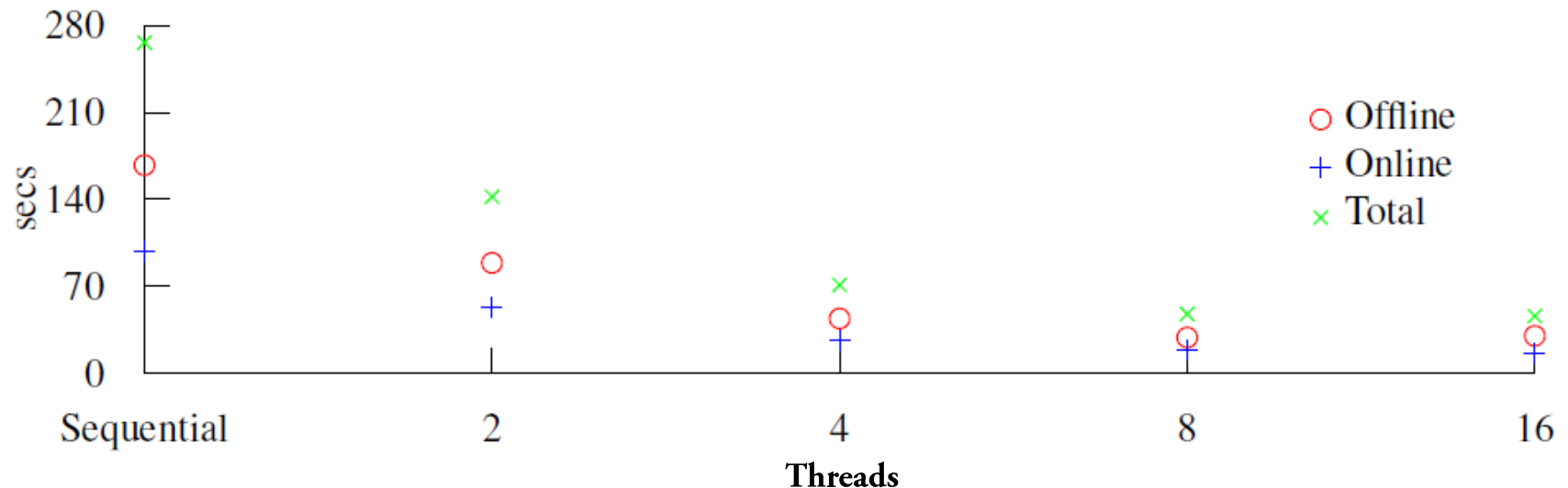
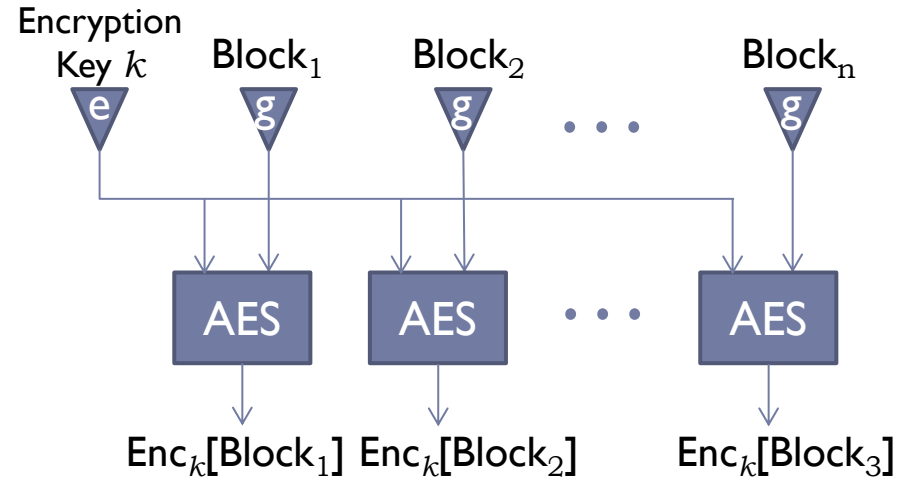
- ▶ Encryption of 128 bits
- ▶ Data owned by Garbler
- ▶ Encryption key owned by Evaluator
- ▶ Circuit kindly provided by Schneider
  - ▶ 38366 gates parallelizable in 327 layers
- ▶ Comparison with the most efficient sequential implementation

[Huang, Evans, Katz, Malka, 2011]

	Phase	Sequential	Fine-Grained	Huang et al.
Offline	Garbling	0.001	0.001	1.438
	OT precomputation	0.133	0.082	
	Garbled tables transmission	0.039	0.044	
Online	Garbler's secret transmission	0.000	0.000	0.038
	Evaluator's secret transmission	0.013	0.002	0.086
	Evaluation	0.066	0.017	0.311

# Parallel AES Encryption

- ▶ Encryption of greyscale 256x256 pixels image
  - ▶ 4096 blocks evaluated in parallel



# Conclusions

---

- ▶ Addressed an analysis of parallel implementation of GC
- ▶ Two different parallelization techniques
  - ▶ Fine-grained (gate)
  - ▶ Coarse-grained (macroblocks)
- ▶ Tests performed on two different scenarios
  - ▶ Both the solutions improve performances
  - ▶ Coarse-grained is preferable, when applicable
  - ▶ Optimum solutions for multi-core systems
  
- ▶ Future works:
  - ▶ Study on circuit design for efficient parallelization
  - ▶ Implementation and tests on GPUs
  - ▶ Malicious setting analysis

