Composable Heuristics for Register-Transfer Level Logic Locking Optimization

Luca Collini
luca.collini@mail.polimi.it
CSE

Supervisor: Prof. Christian Pilato

POLITECNICO
MILANO 1863

HONOURS PROGRAMME
HP-SR
in Information Technology

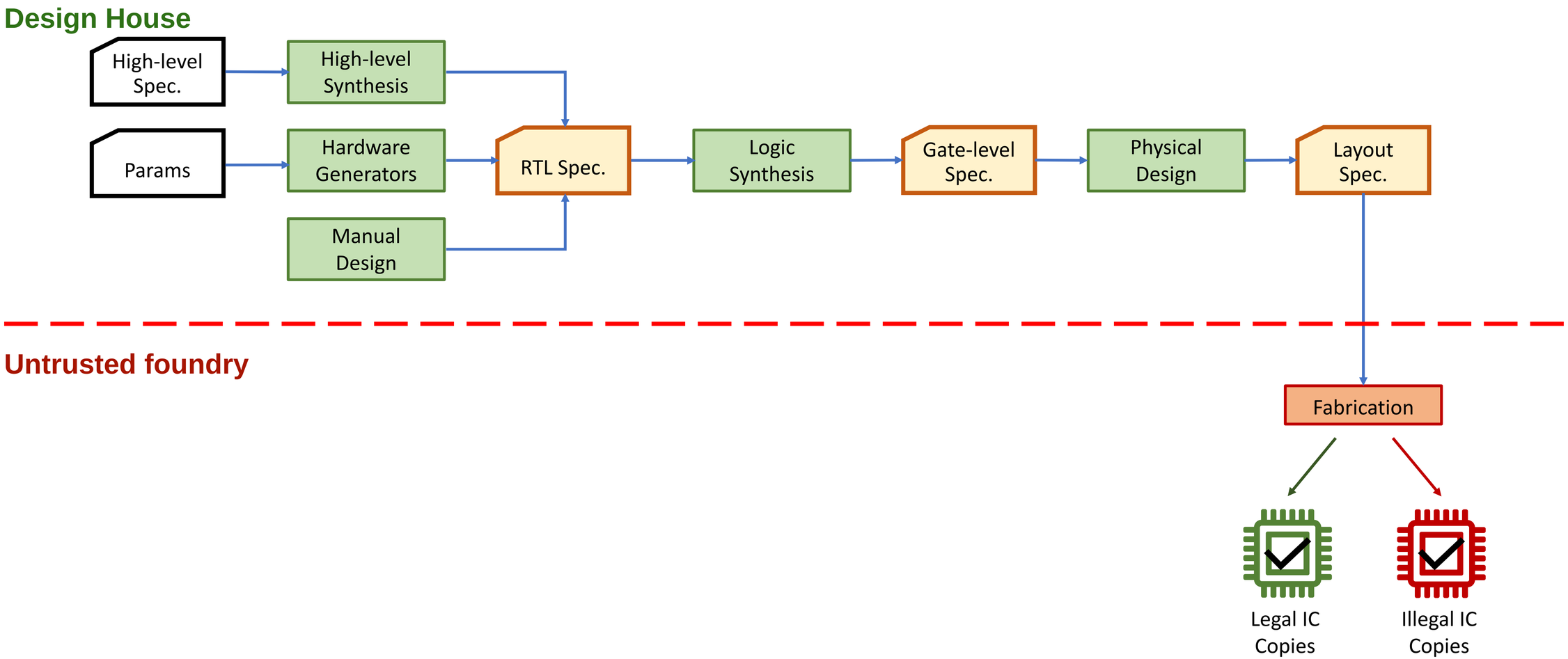
Outline

- Introduction to Hardware IP Protection
- Problem Definition
- Overview of the Approach: Heuristics for RTL Locking
- Implementation Details
- Experimental Evaluation

Outline

- **Introduction to Hardware IP Protection**
- Problem Definition
- Overview of the Approach: Heuristics for RTL Locking
- Implementation Details
- Experimental Evaluation

Globalization of the IC supply chain



Security threats

Reverse Engineering

Intellectual Property theft

total loss from IC counterfeiting was estimated to be about $169 billions in 2011

80% of reported counterfeited parts in 2019 were never reported before

Malicious modifications

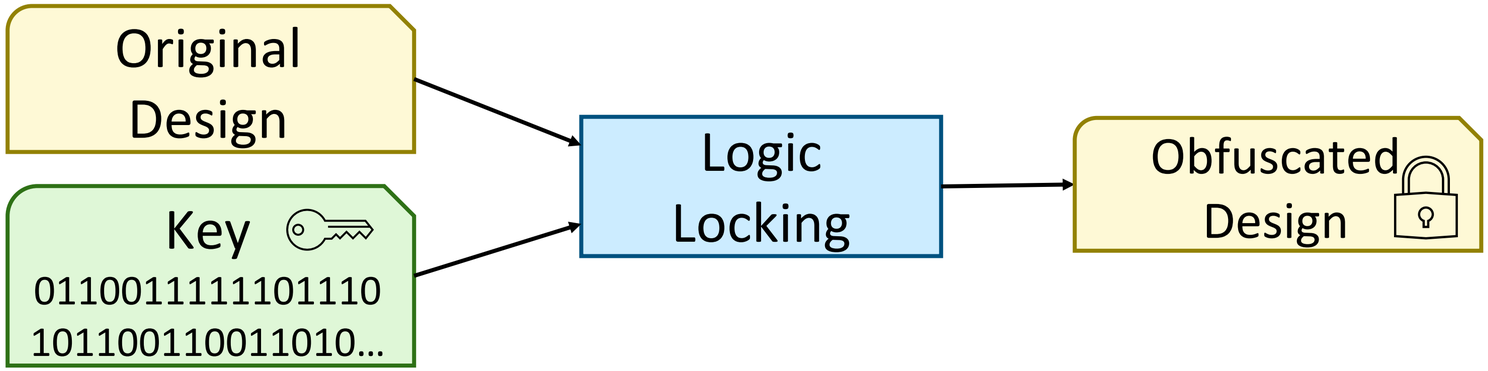
planned obsolescence trojans

backdoor insertion

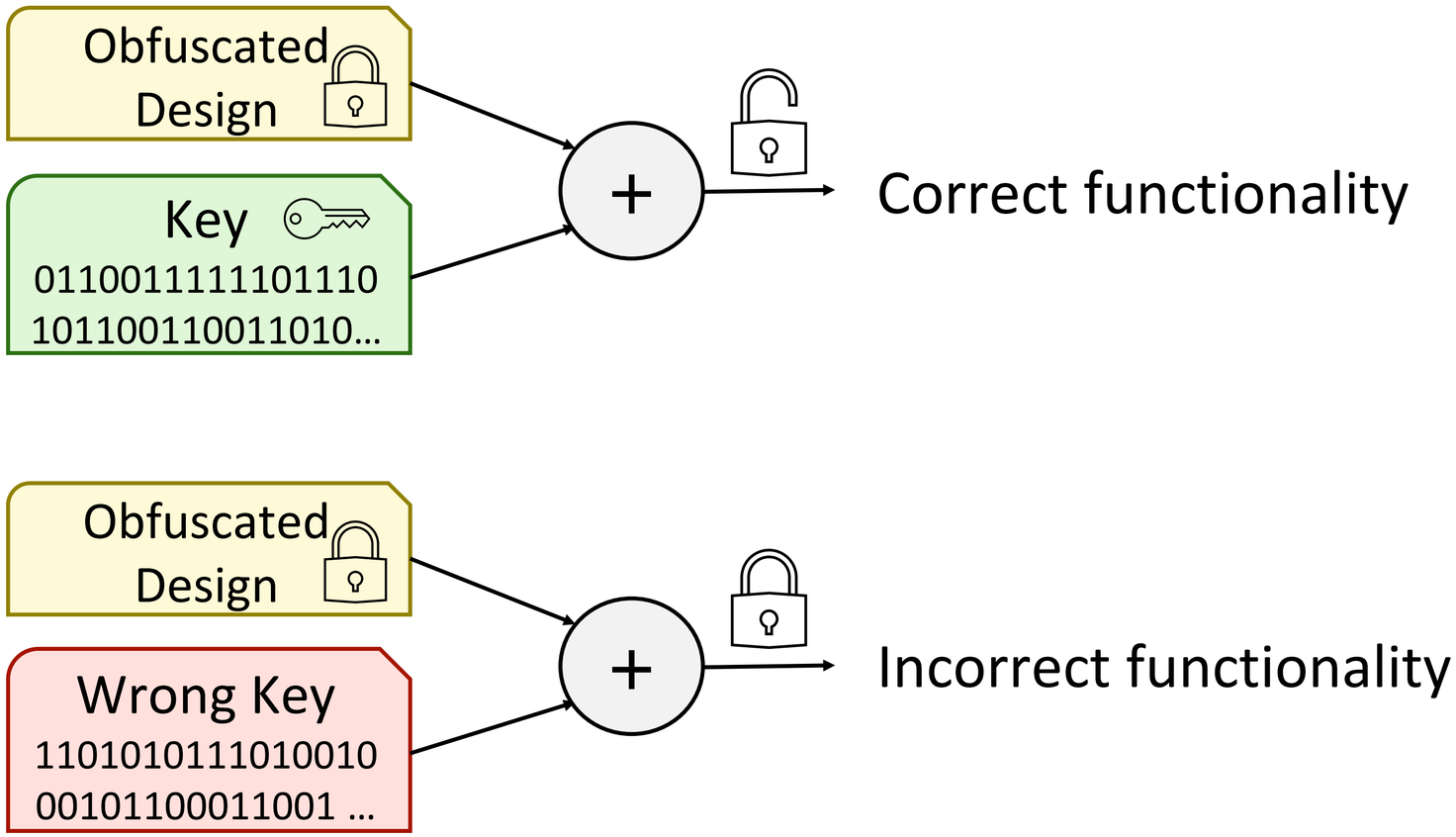
attempted trade of counterfeited Cisco equipment to the US Department of Defense

Logic Locking

Thwarting reverse engineering



Logic Locking



Outline

- Introduction to Hardware IP Protection
- **Problem Definition**
- Overview of the Approach: Heuristics for RTL Locking
- Implementation Details
- Experimental Evaluation

Threat Model

***Oracle**:* a chip that performs correct computation

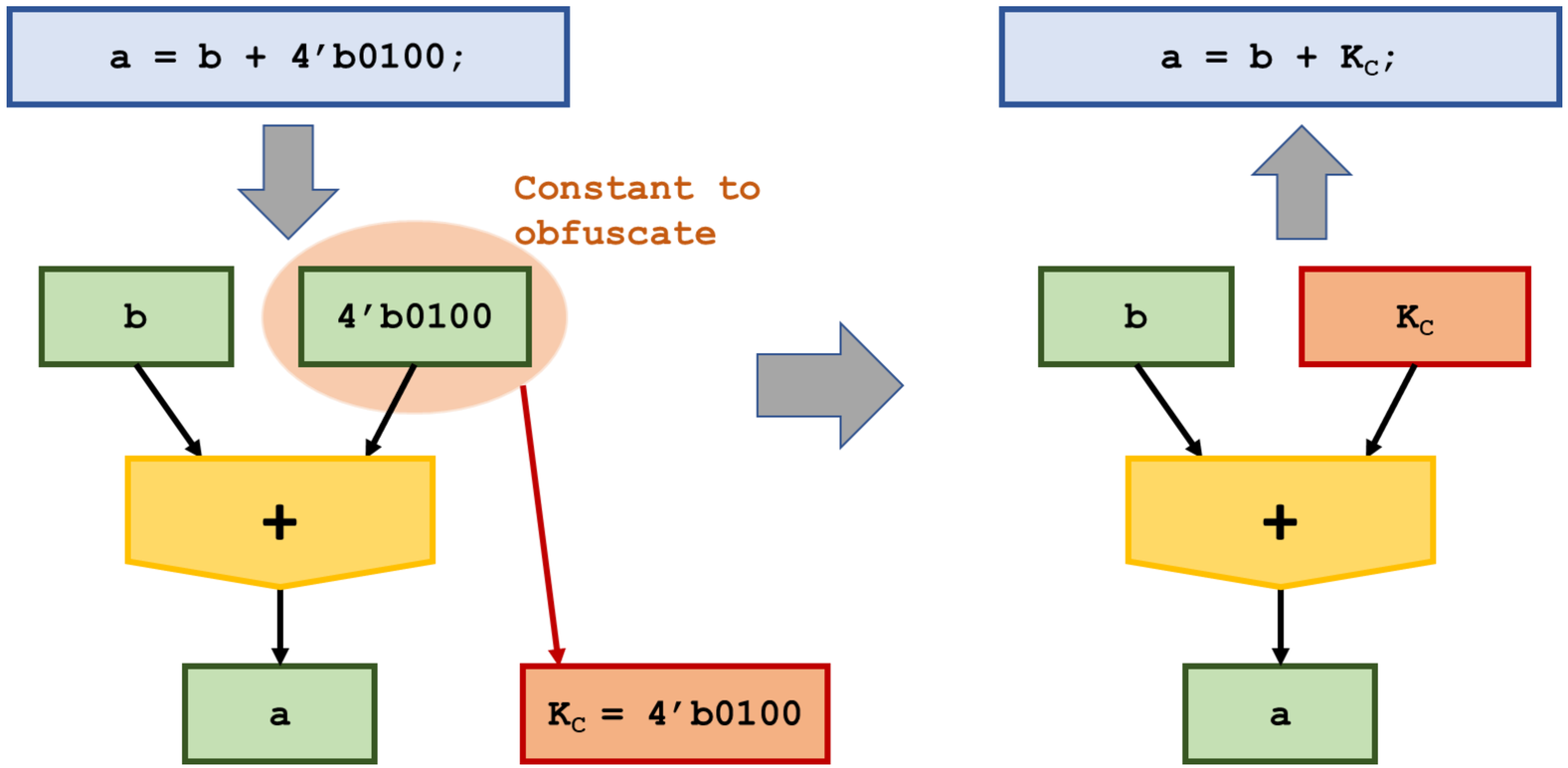
***Ambiguity**:* ability of an attacker to distinguish between primary inputs and key inputs

Oracle-less scenario with ***distinct ambiguity***

The attacker has no oracle but can distinguish data and control signals

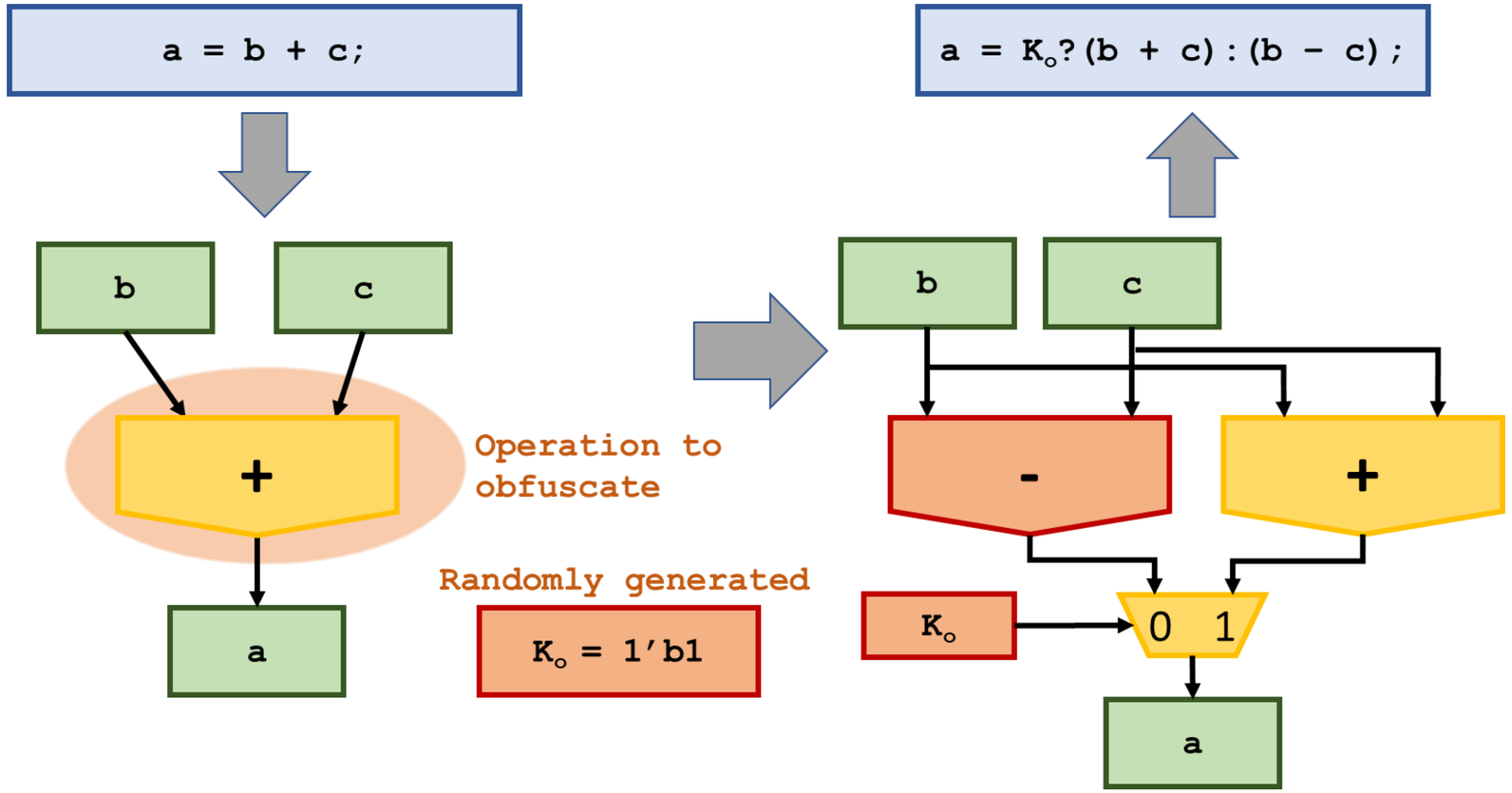
Semantic Obfuscation

Constant obfuscation



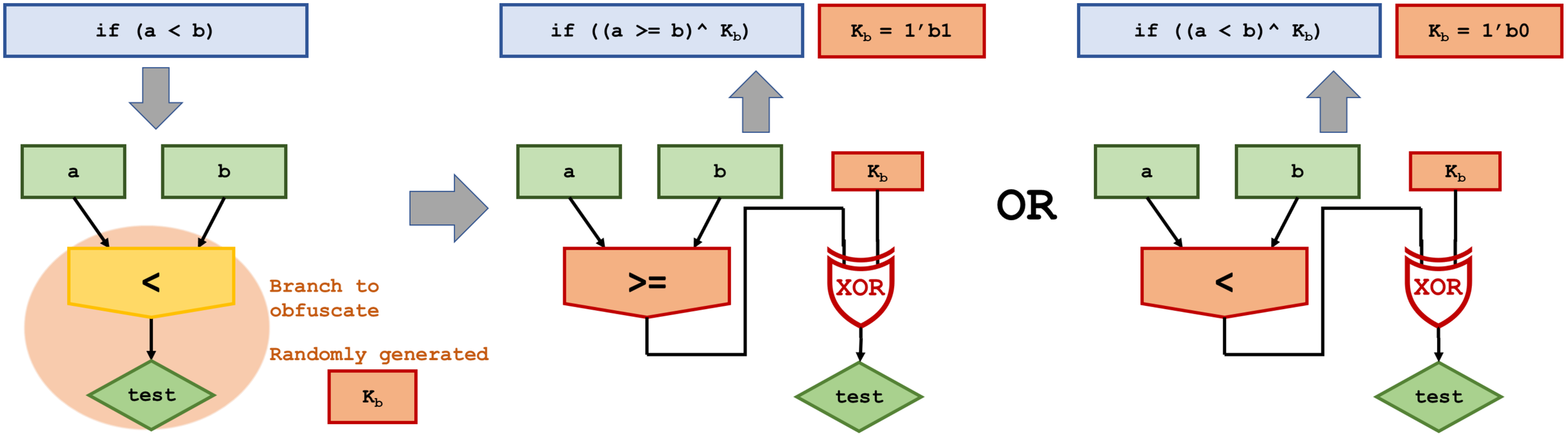
Semantic Obfuscation

Operation obfuscation



Semantic Obfuscation

Branch obfuscation



Security Evaluation

Mean Differential Entropy:

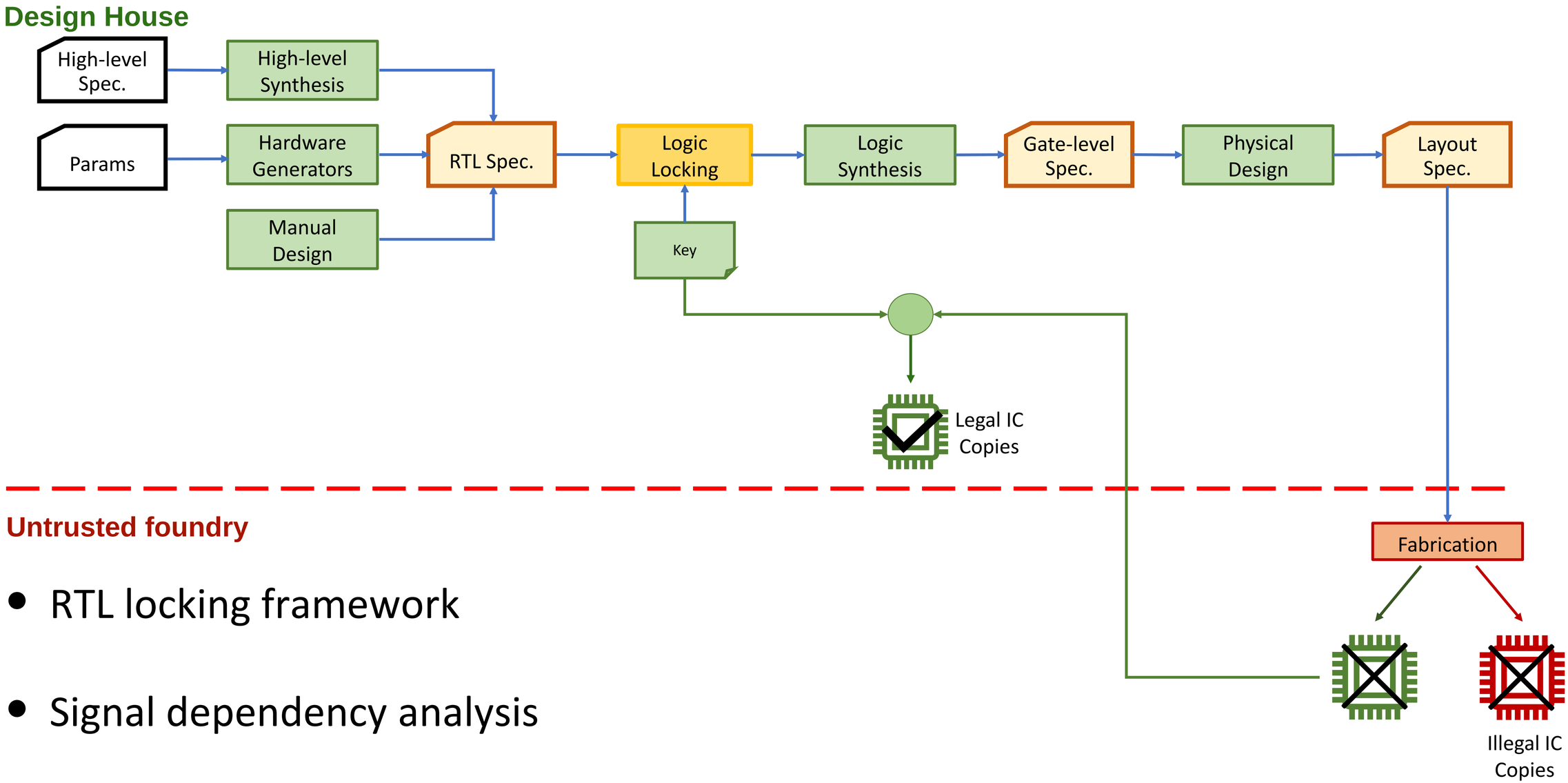
$$H = \sum_{i=1}^{N} \left(P_i \cdot log \frac{1}{P_i} + (1 - P_i) \cdot log \frac{1}{1 - P_i} \right) \cdot \frac{1}{N}$$

$$P_i = \frac{\sum_{w=1}^{N} \sum_{t=1}^{M} OUT[i]_t \bigoplus OUT[i]_{t,w}}{N \cdot M}$$

Problems of Logic Locking

- Logic locking introduces Area, Power, and Timing overheads
- Netlist Locking
  - Post-synthesis obfuscation ➡ Information embedded in the netlist
  - Obfuscation of random points ➡ May yield invalid designs
- RTL Locking
  - Obfuscation in topological order ➡ Design dependent solutions
- HLS Locking
  - DSE for obfuscation optimization ➡ High computational cost

Contributions



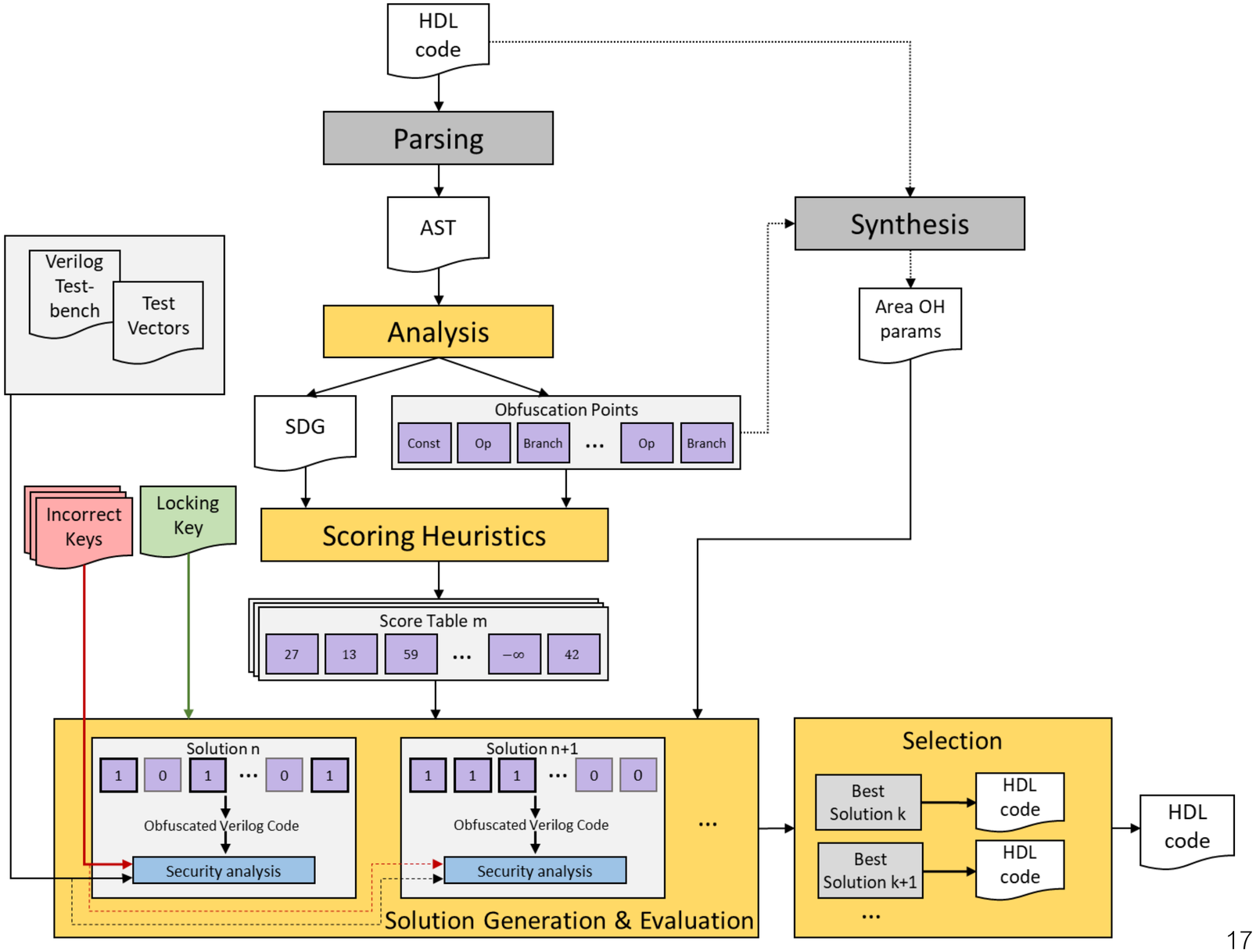
- RTL locking framework
- Signal dependency analysis
- Optimization under area and key-bit constraints

Outline

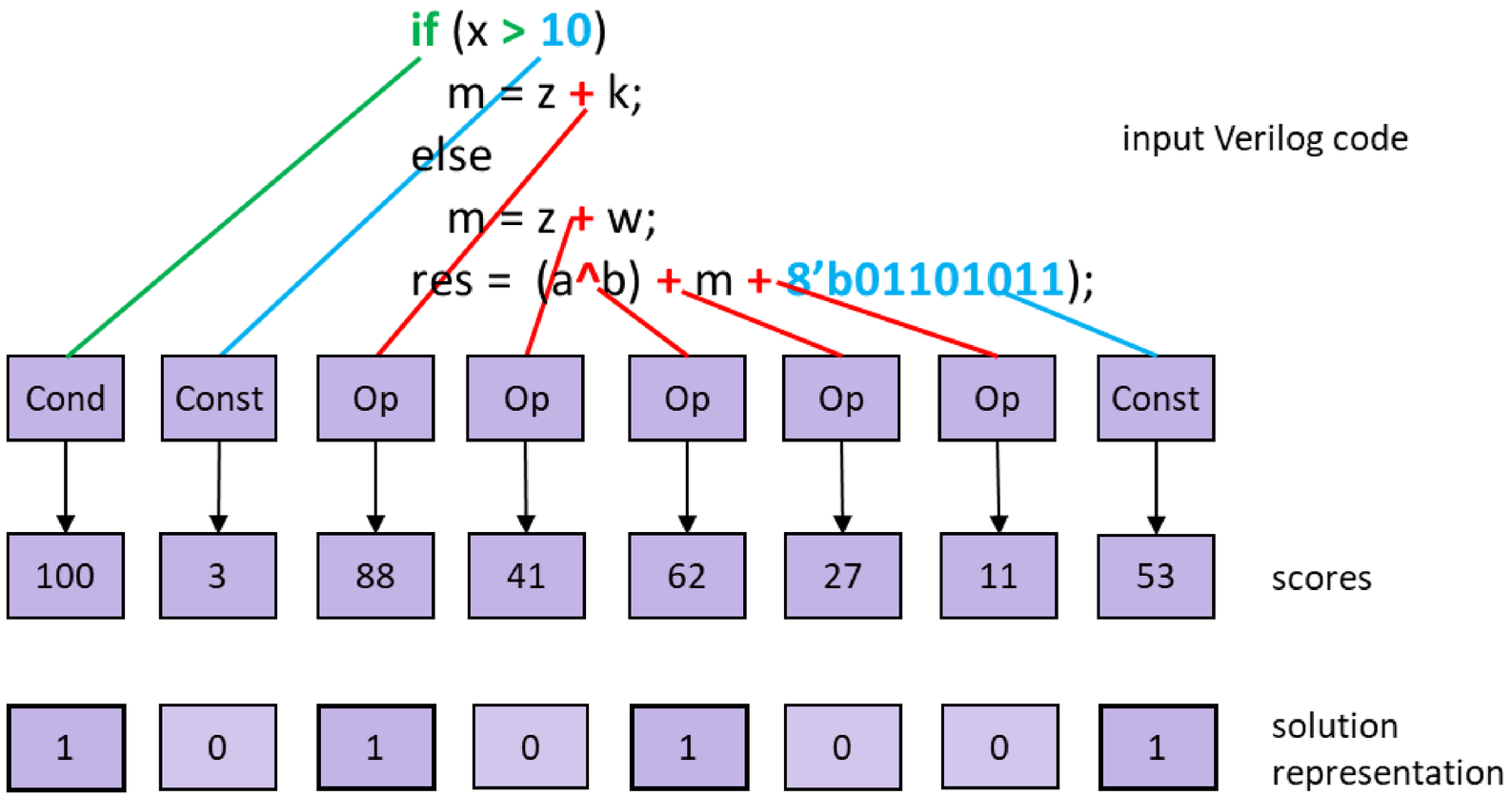
- Introduction to Hardware IP Protection
- Problem Definition
- **Overview of the Approach: Heuristics for RTL Locking**
- Implementation Details
- Experimental Evaluation

RTL Locking Framework

- Dependence analysis
- Composable scoring heuristics
- Area overhead estimations



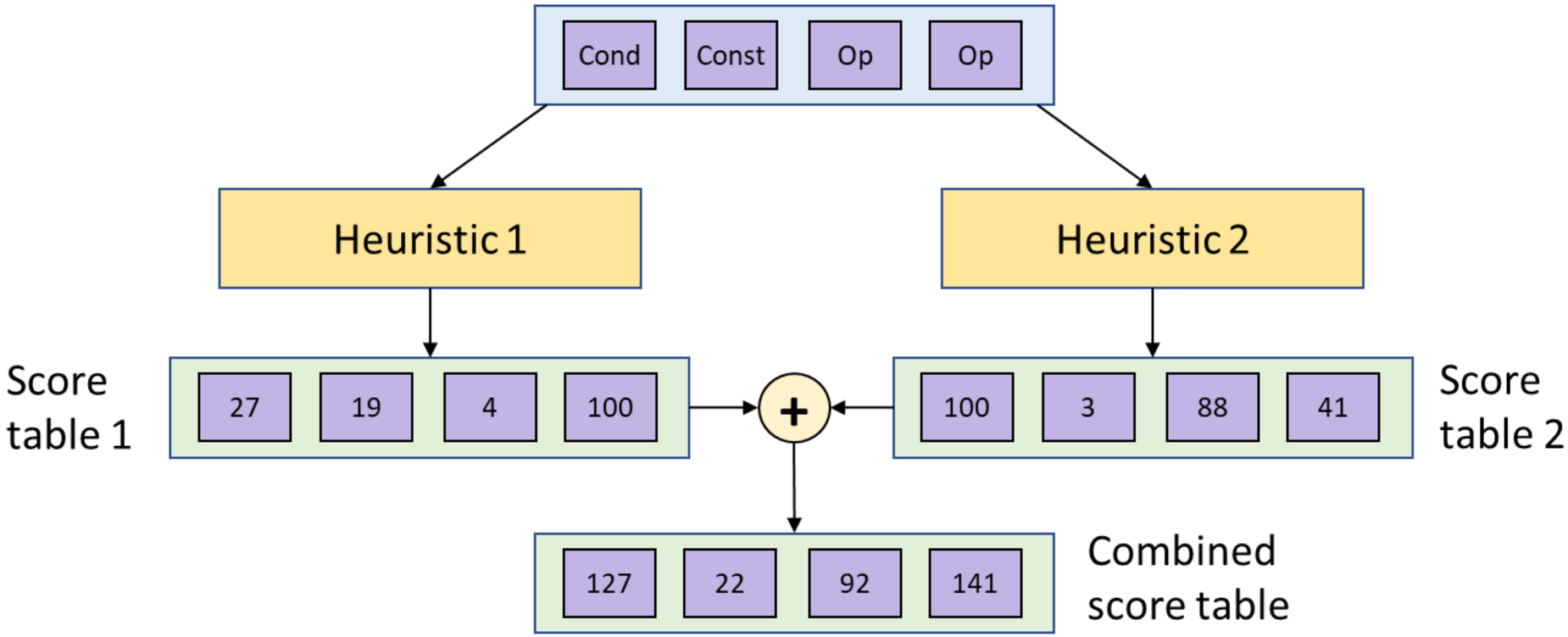
Score Table & Solution Representation




```
if ((x <= 10)^key[0]) m = key[1] ? z-k : z+k;
else m = z + w;
res =  (key[2]?(a|b) : (a^b)) + m + key[3:10];
```

obfuscated Verilog code

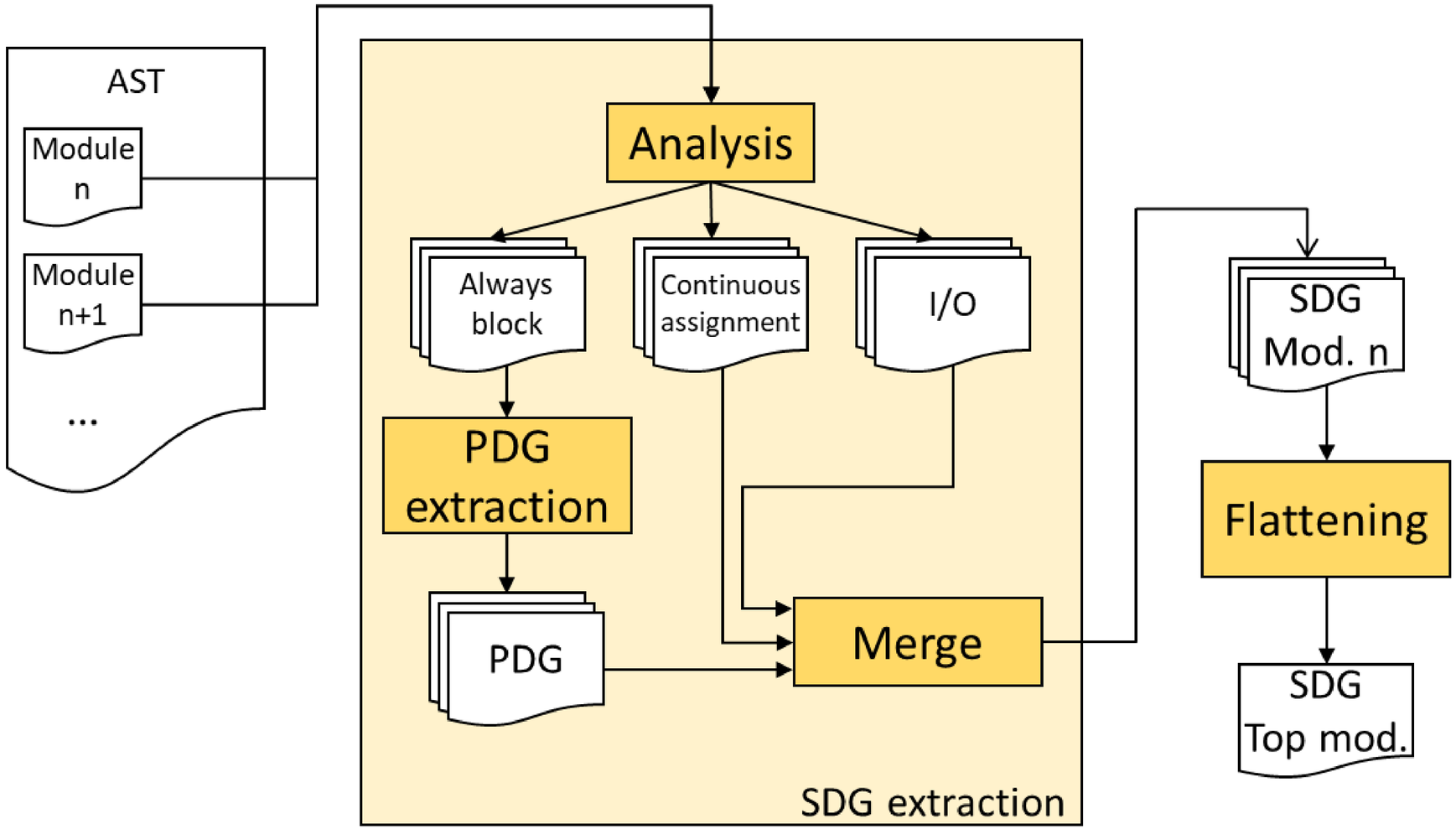
Heuristic Combination



Outline

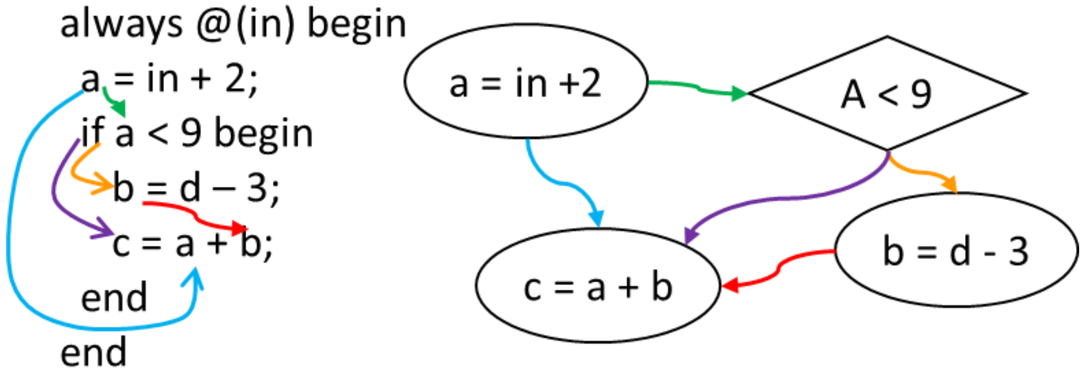
- Introduction to Hardware IP Protection
- Problem Definition
- Overview of the Approach: Heuristics for RTL Locking
- **Implementation Details**
- Experimental Evaluation

SDG extraction flow

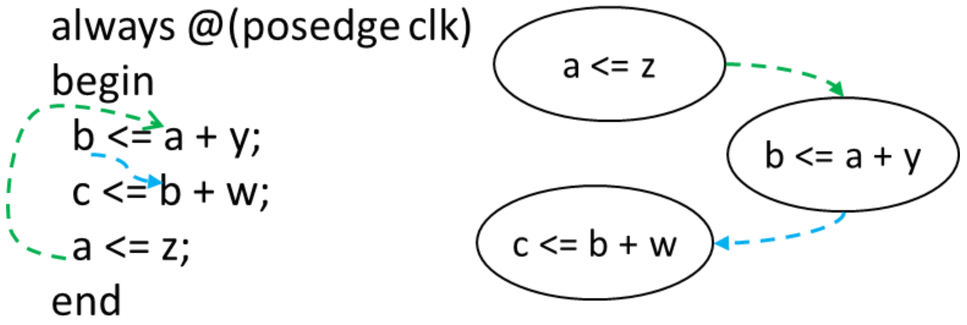


Dependencies

- Direct Dependence

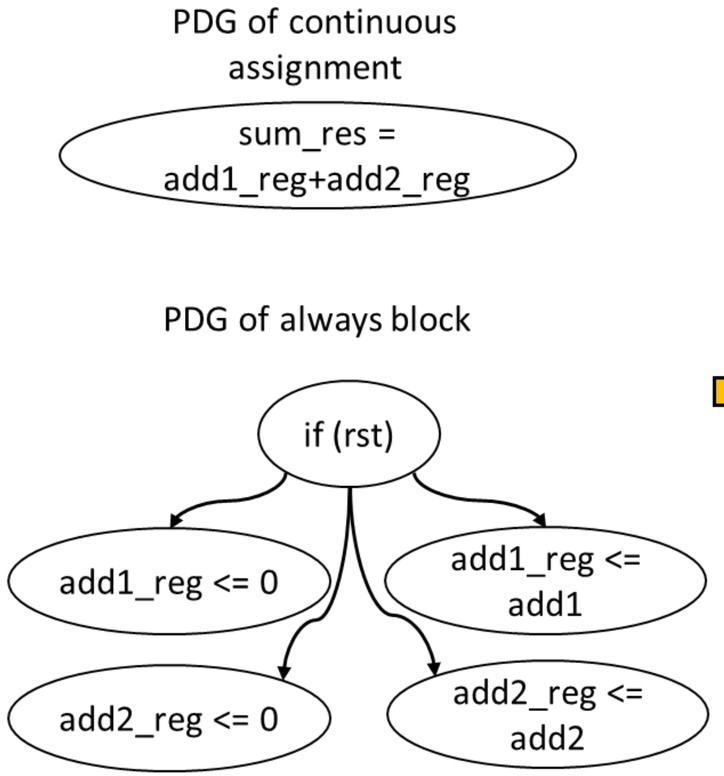


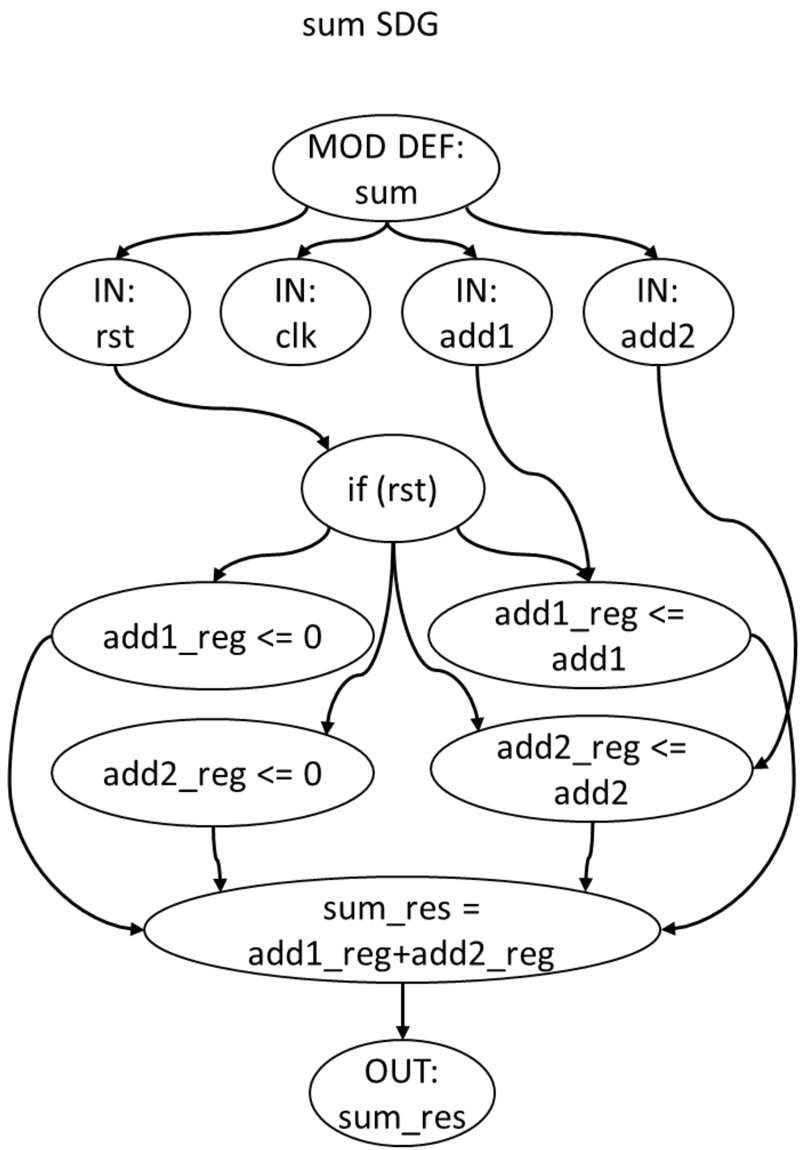
- Inter-cycle Dependence



From PDG to SDG

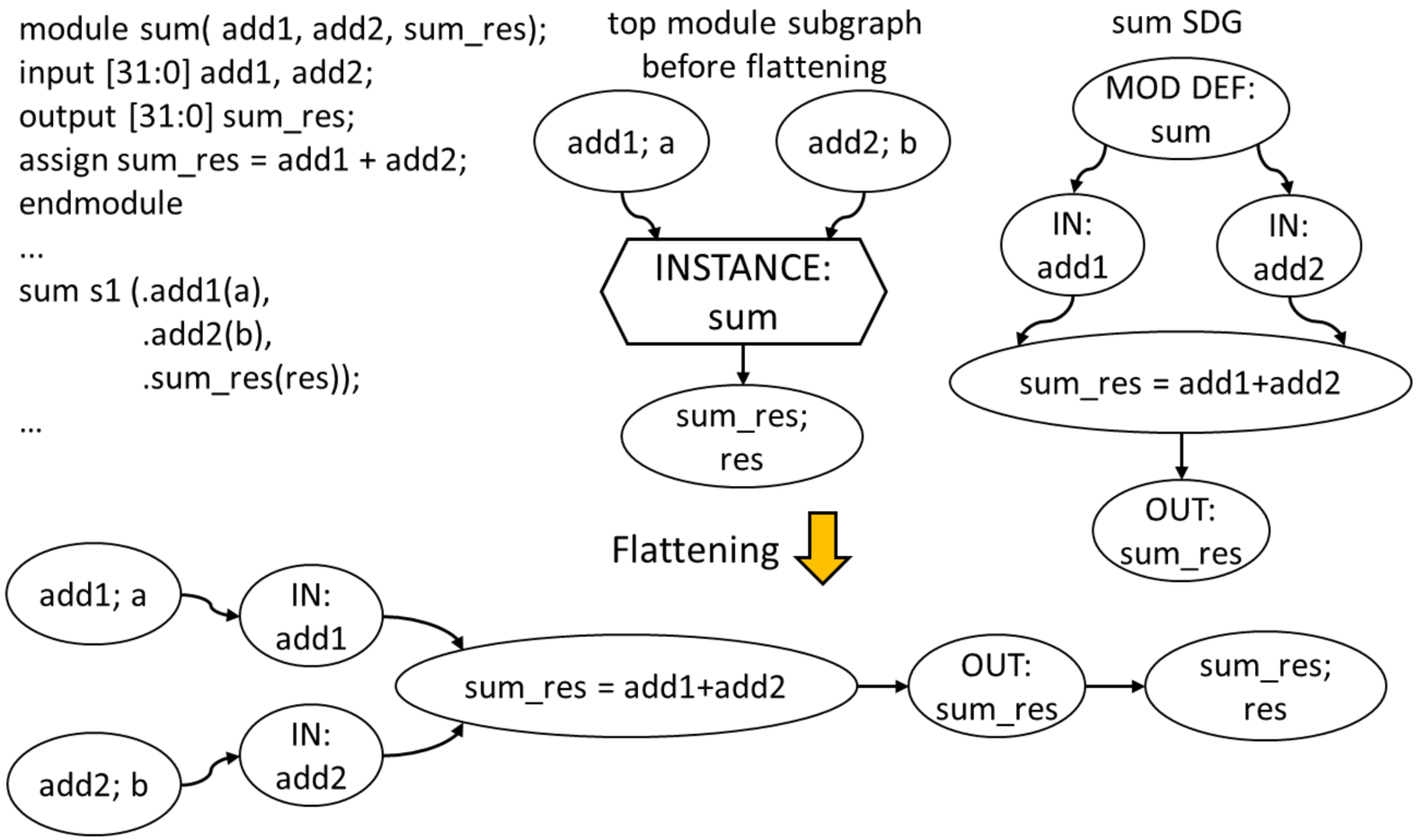
```
module sum(clk, rst, add1, add2, sum_res);
input clk, rst;
input [31:0] add1, add2;
output [31:0] sum_res;
reg [31:0] add1_reg, add2_reg;
assign sum_res = add1_reg + add2_reg;

always @(posedge clk, rst) begin
        if (rst) begin
                add1_reg <= 0;
                add2_reg <= 0;
        end
        else begin
                add1_reg <= add1;
                add2_reg <= add2;
        end
endmodule
```





Module Flattening

```
module sum( add1, add2, sum_res);
input [31:0] add1, add2;
output [31:0] sum_res;
assign sum_res = add1 + add2;
endmodule
...
sum s1 (.add1(a),
        .add2(b),
        .sum_res(res));
...
```

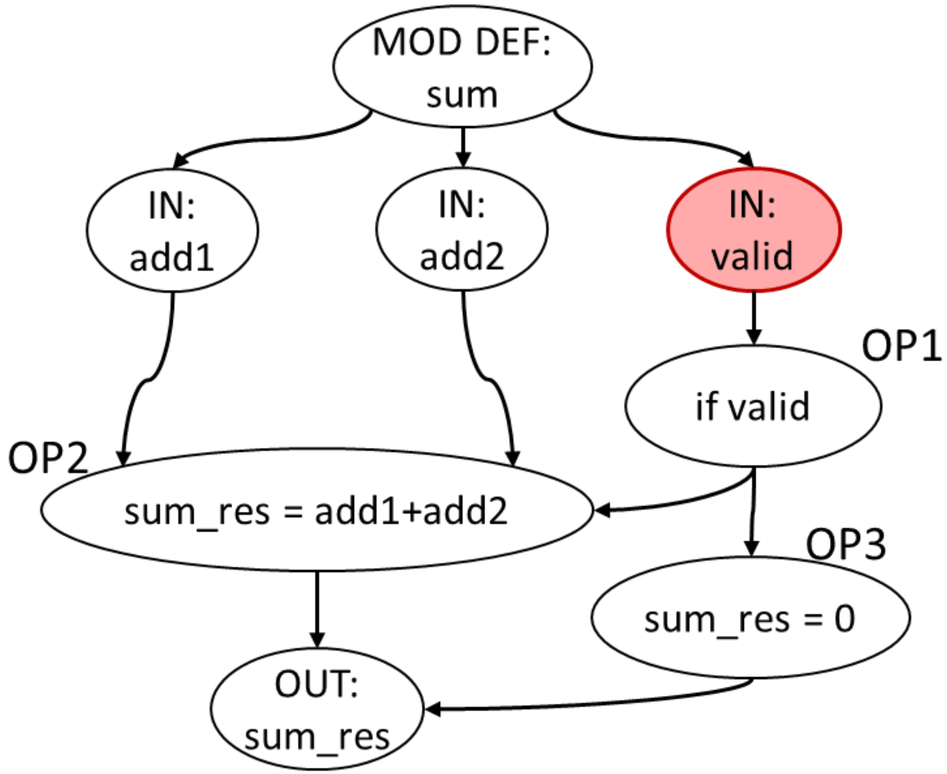


Representation of the entire design from inputs to outputs

Scoring Heuristics

Control Disabling

Avoid obfuscation of points that cause simulation failures

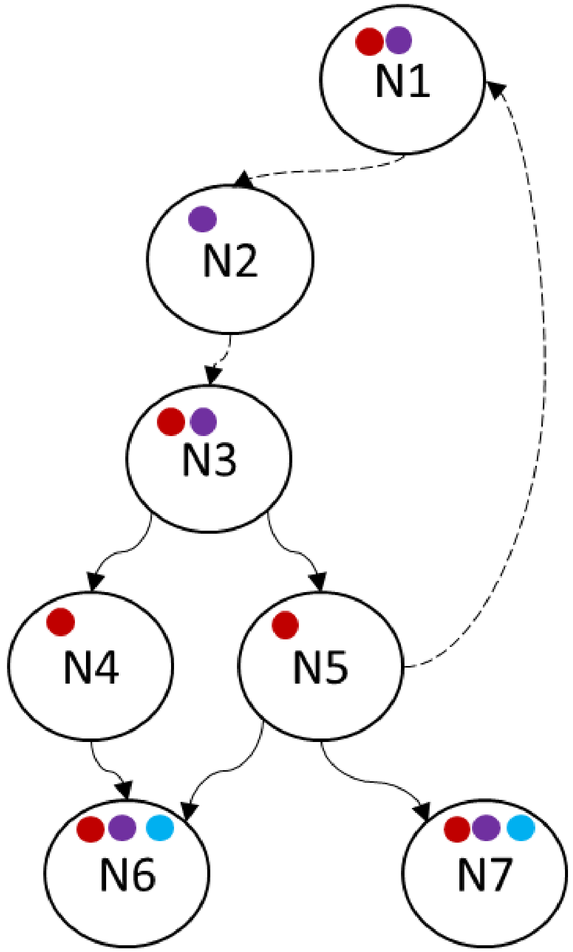


Score Table Control Disabling:
OP1 = -inf
OP2 = 0
OP3 = 0

Scoring Heuristics

Bounded (Direct) Children

Reward points that influence a big design portion



Direct dependency

Inter-cycle dependency

Assuming OPX associated with SDG node NX:

bounded_children(OP2, 3) = **6**

bounded_children(OP5, 3) = **5**

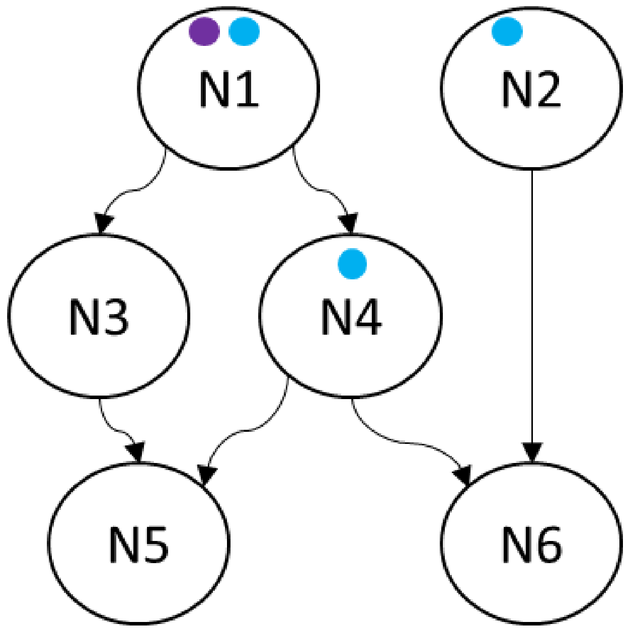
bounded_direct_children(OP2, 3) = **0**

bounded_direct_parents(OP5, 3) = **2**

Scoring Heuristics

Bounded Parents

Reward points with high convergence



Assuming OPX associated with SDG node NX:

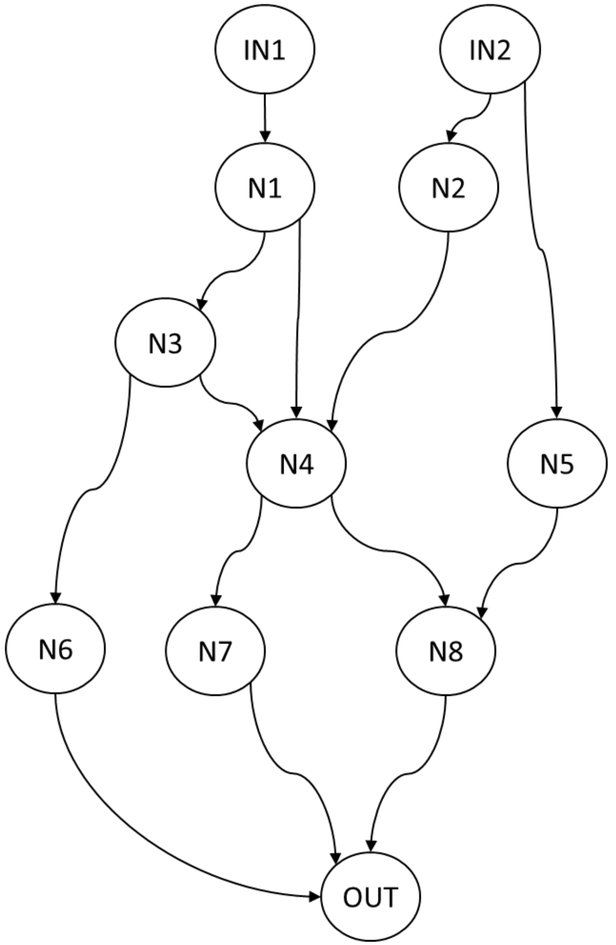
bounded_parents(OP4, 2) = **1**

bounded_parents(OP6, 2) = **3**

Scoring Heuristics

Max I/O Path Length

Reward long chains of obfuscation points



Assuming OPX associated with SDG node NX:
Paths:

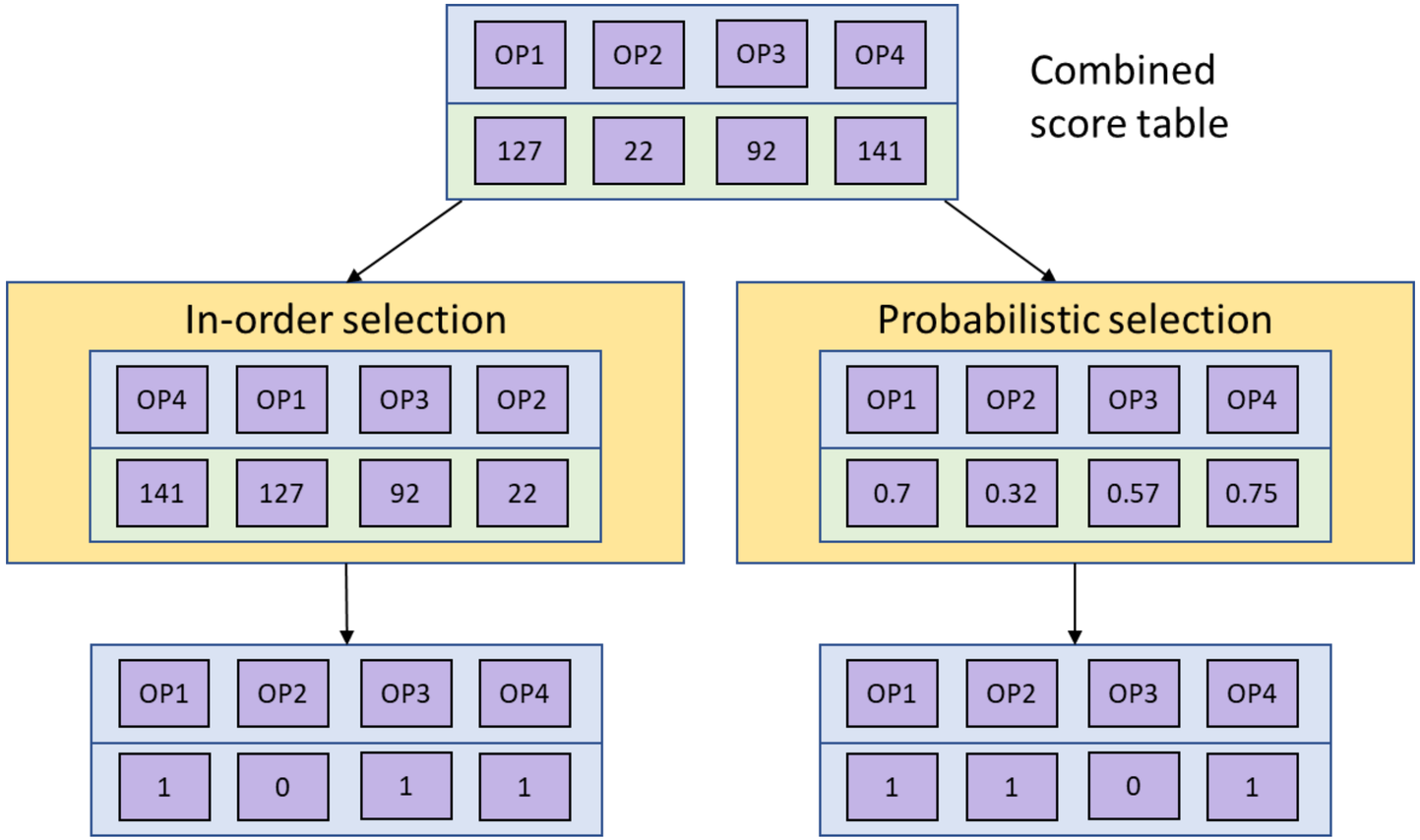
P1: IN1-N1-N3-N4-N7-OUT;	#OP = 4
P2: IN1-N1-N3-N4-N8-OUT;	#OP = 4
P3: IN1-N1-N3-N6-OUT;	#OP = 3
P4: IN1-N1-N4-N7-OUT;	#OP = 3
P5: IN1-N1-N4-N8-OUT;	#OP = 3
P6: IN2-N2-N4-N8-OUT;	#OP = 3
P7: IN2-N5-N8-OUT;	#OP = 2

For each Opx we search Nx in the paths and assign the length of the first path in which Nx is found.

Score Table:

OP1 = 4	OP5 = 2
OP2 = 3	OP6 = 3
OP3 = 4	OP7 = 4
OP4 = 4	OP8 = 4

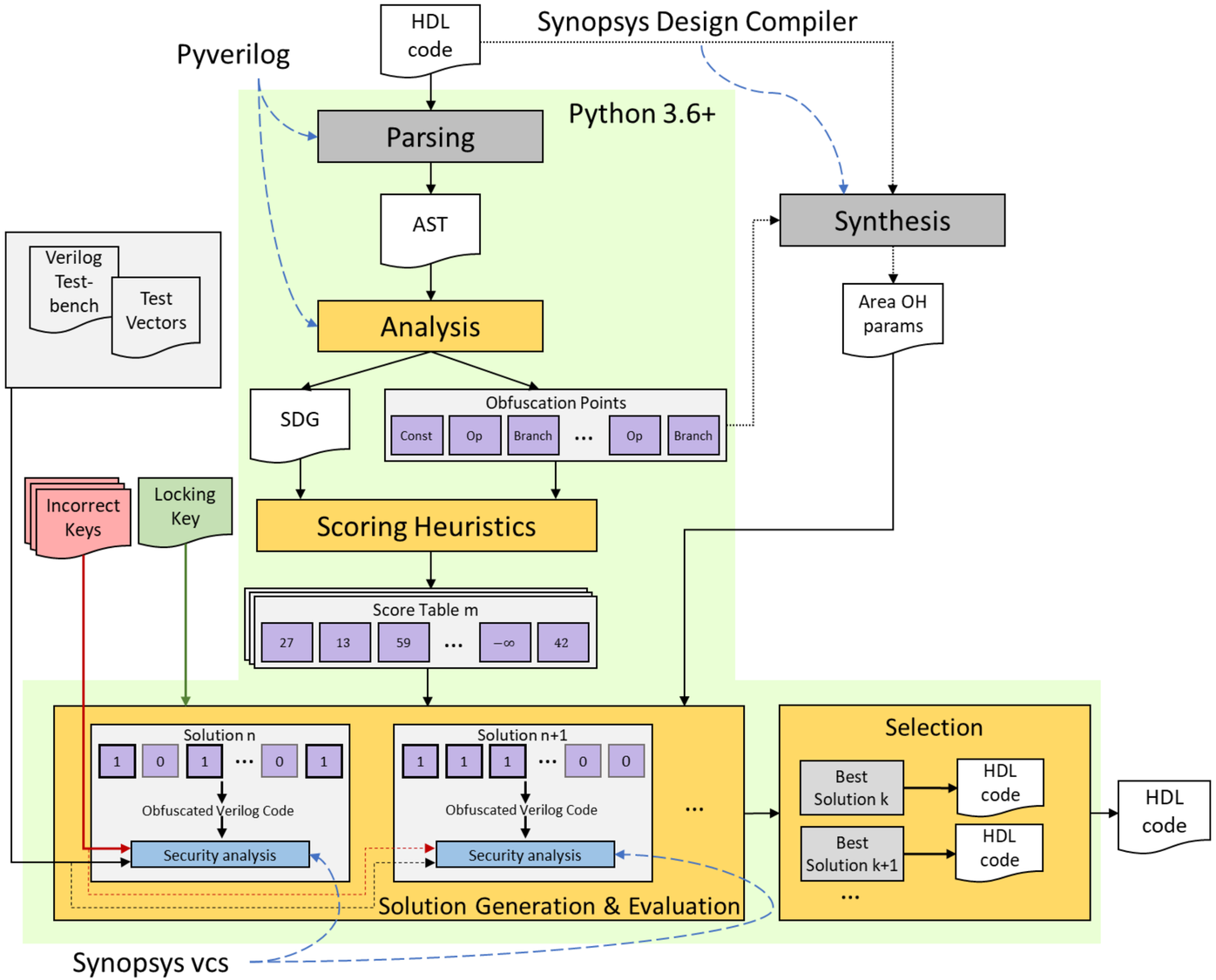
Selection Methods



Outline

- Introduction to Hardware IP Protection
- Problem Definition
- Overview of the Approach: Heuristics for RTL Locking
- Implementation Details
- **Experimental Evaluation**

Experimental Setup



Benchmarks

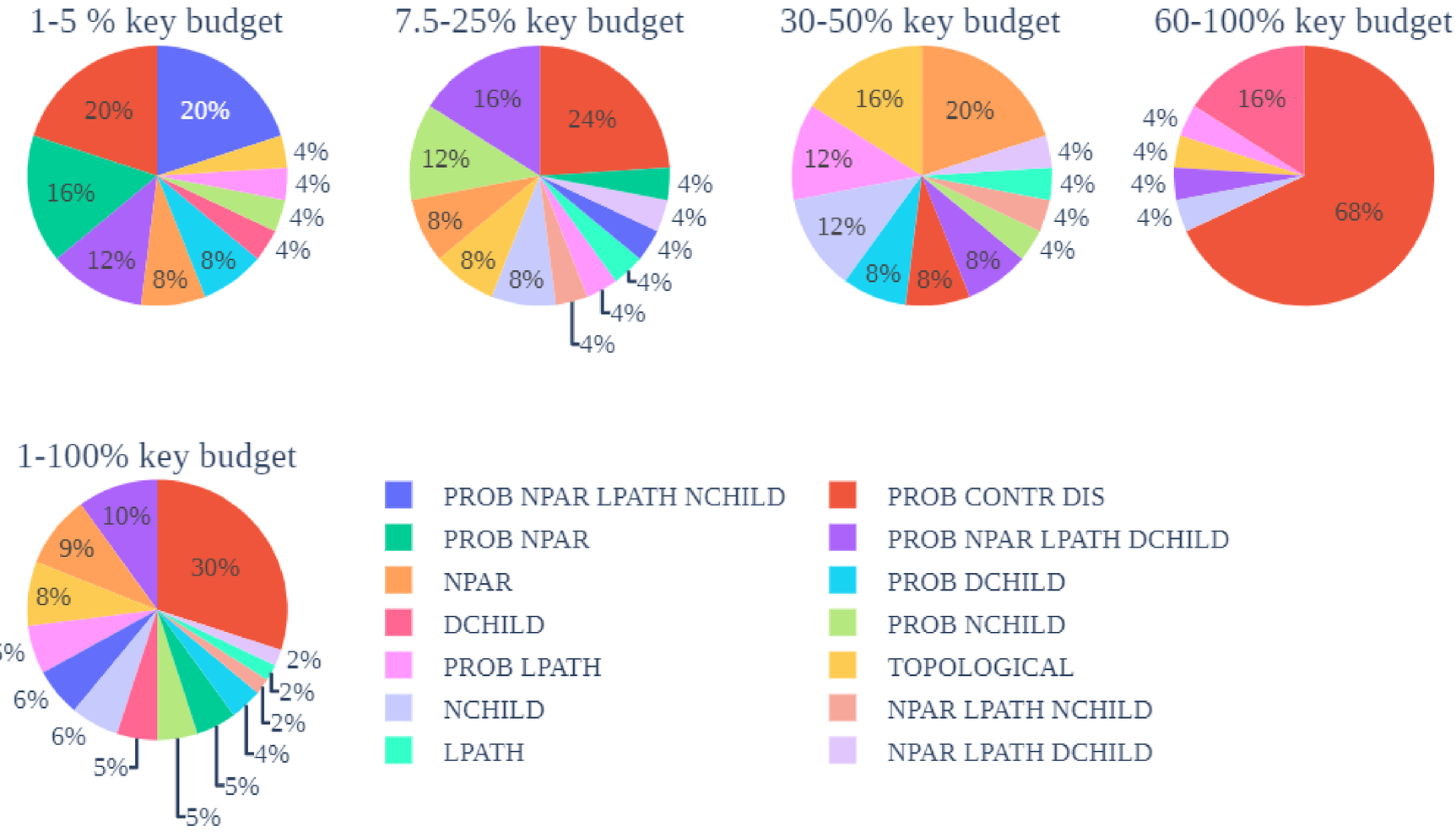
Benchmarks from CEP-MIT for evaluating security solutions

Design	Modules	Const	Ops	Branches	# Bits	SDG nodes
FIR	5	10	24	0	344	157
IIR	5	19	43	0	651	231
SHA256	3	159	36	2	4,992	619
MD5	2	150	50	1	4,533	829
DES3	11	523	3	775	2,990	3,745

1, 2, 3, 4, 5, 7.5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 70, 80, 90, 100% of key budget

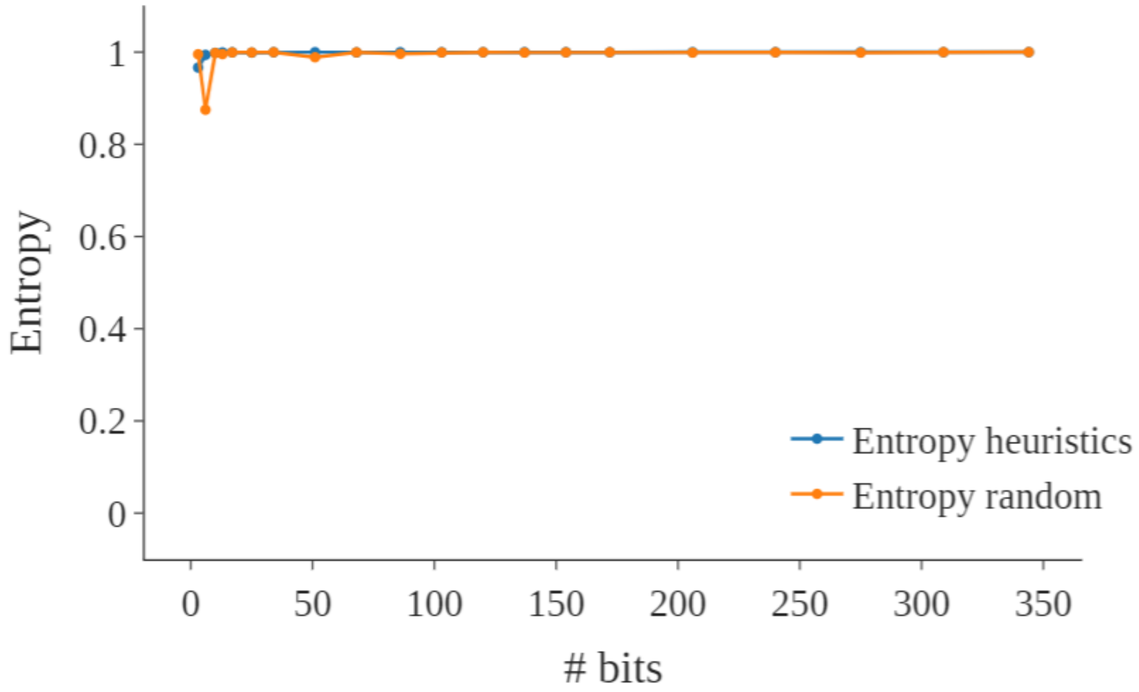
Experimental Results

- No dominant heuristic
- Better than ASSURE in 92% of the cases

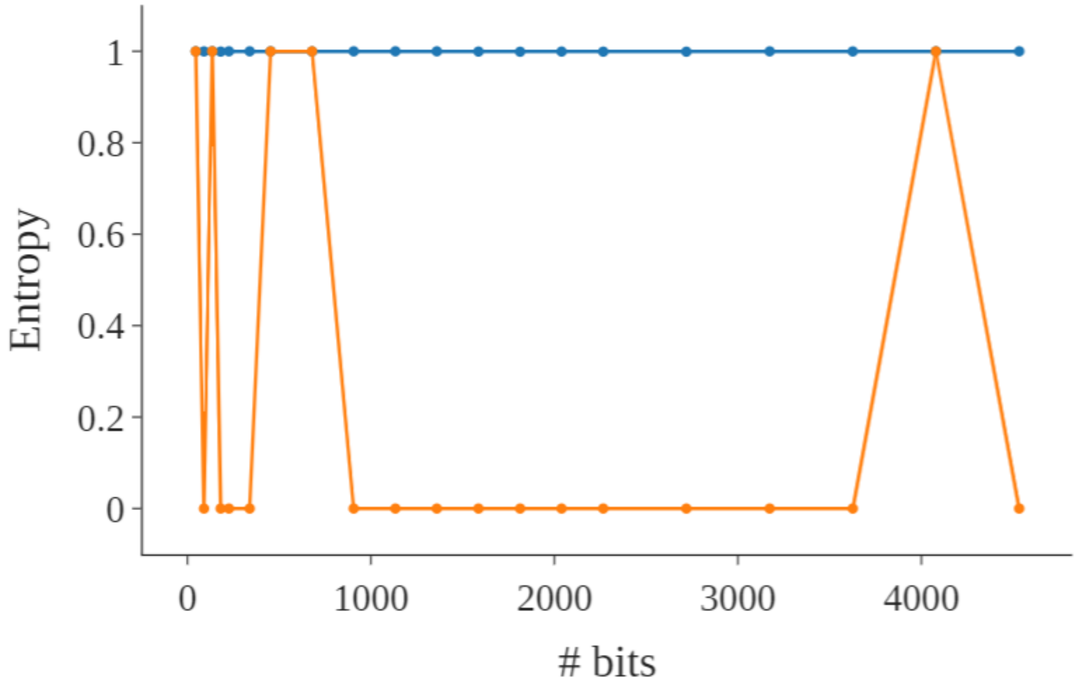


Experimental Results

- Random obfuscation selects obfuscation points that lead to invalid designs



FIR



MD5

Experimental Results

- Results comparable to DSE approach
- Heuristics are much faster (100-400 times)

Design	Composable Heuristics					DSE				
	Mean differential entropy				Time [min]	Mean differential entropy				Time [min]
	25%	50%	70%	100%		25%	50%	75%	100%	
FIR	0.999853	0.999967	0.999746	0.999935	2	0.999963	1.000000	1.000000	0.999997	240
IIR	0.999236	0.999582	0.999633	0.999842	3	0.999964	0.999999	0.999999	0.999993	360
MD5	0.999939	0.999364	0.999832	0.999832	3	0.999954	0.999952	0.999952	0.999952	450
DES3	0.999947	0.999513	0.998356	0.996788	4	0.999963	0.999957	0.999960	0.999960	600
SHA256	0.993654	0.993307	0.951003	0.951003	3	0.999540	0.999665	0.999665	0.999665	1300

Average degradation: 0.005888

Conclusion

- Best results in highly constrained scenarios
- Better than ASSURE in 92% of the cases
- Enables optimized obfuscation on more complex designs

Future work:

- New SDG analyses
- New overhead estimators
- Multi-objective optimization

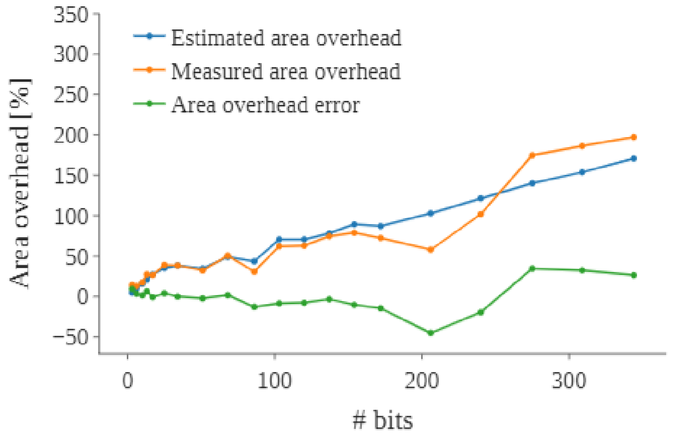
Thank you for your attention!

Luca Collini
luca.collini@mail.polimi.it

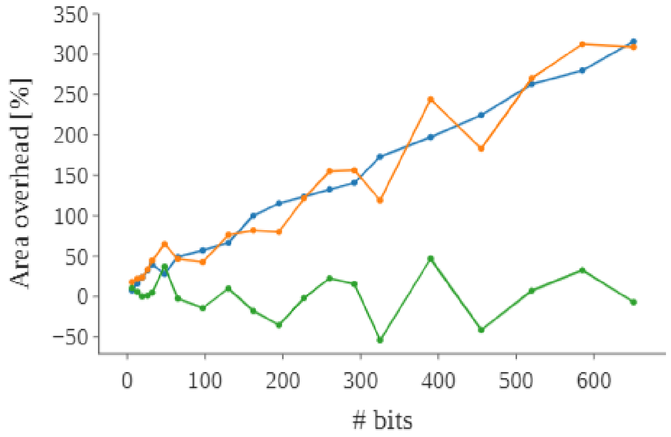
Supervisor: Prof. Christian Pilato

Additional Results

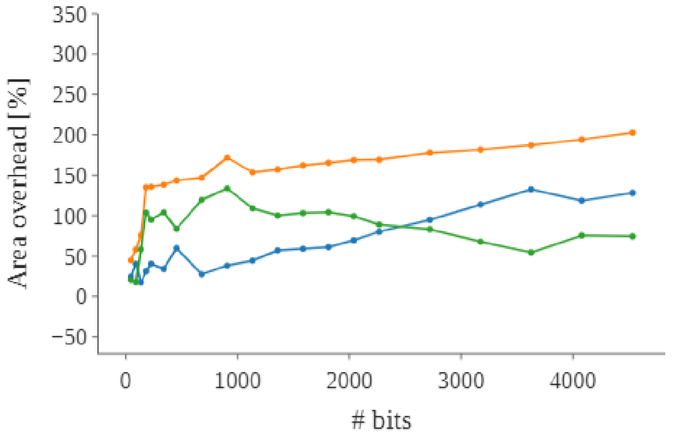
- Best accuracy in highly constrained scenarios



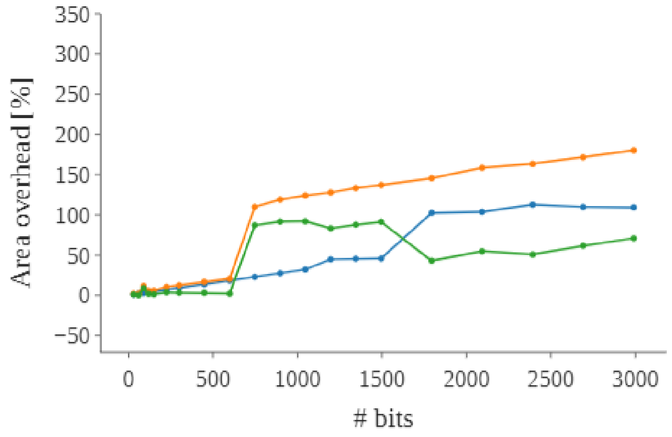
FIR



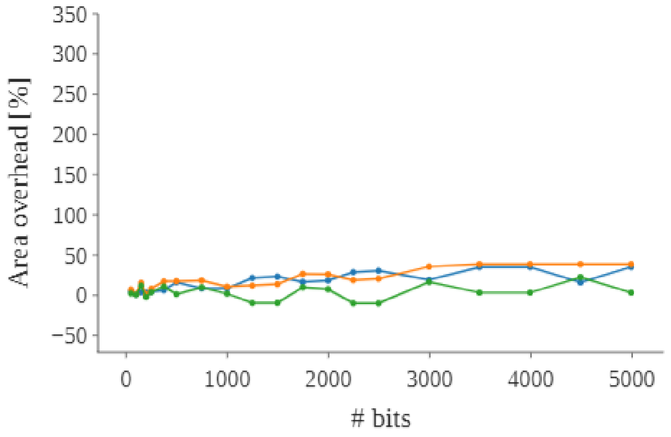
IIR



MD5

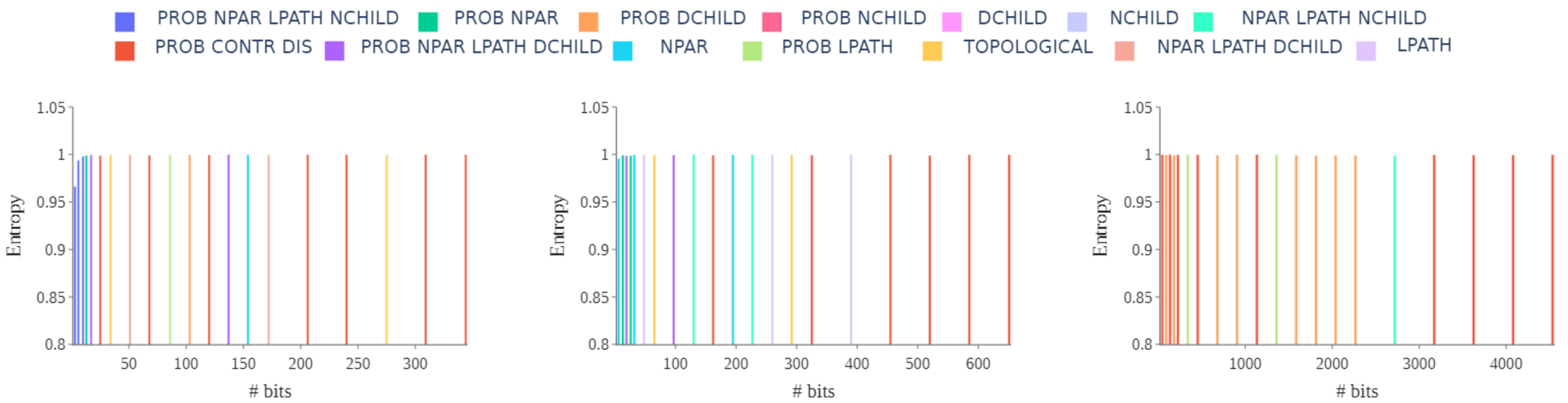


DES3



SHA256

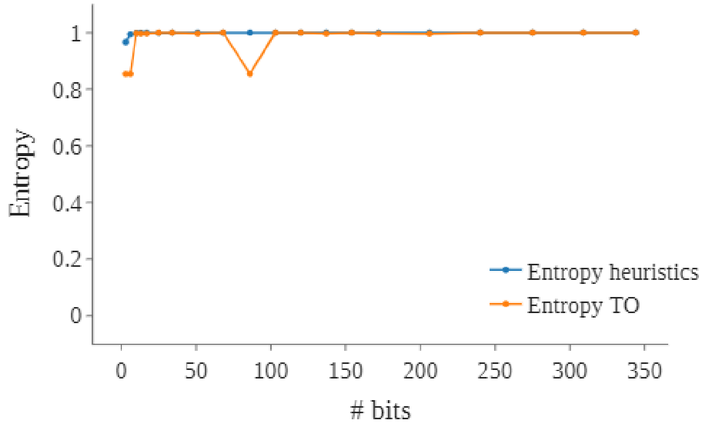
Additional Results



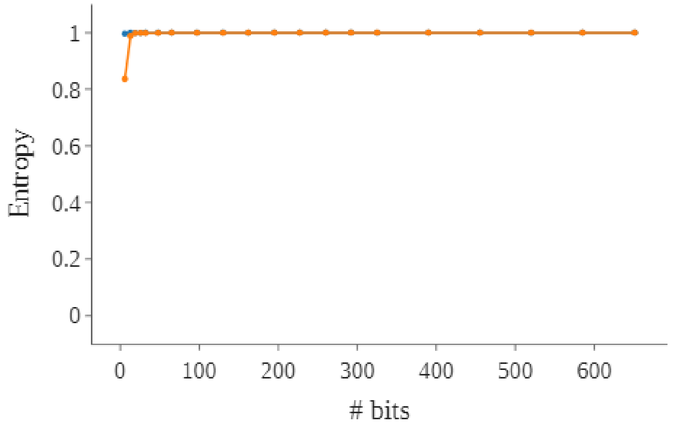
(a) FIR (b) IIR (c) MD5

(d) DES3 (e) SHA256

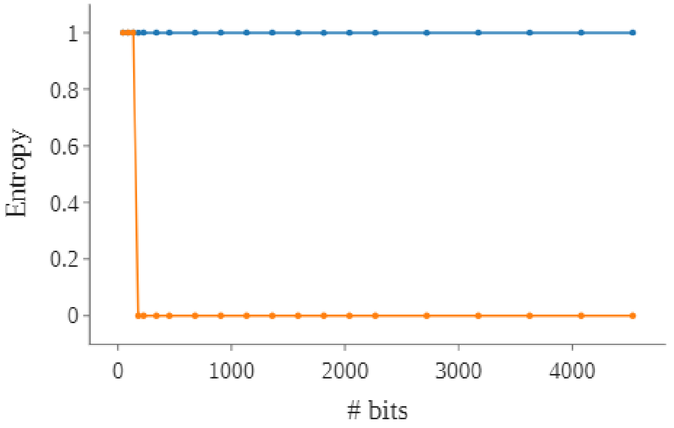
Additional Results



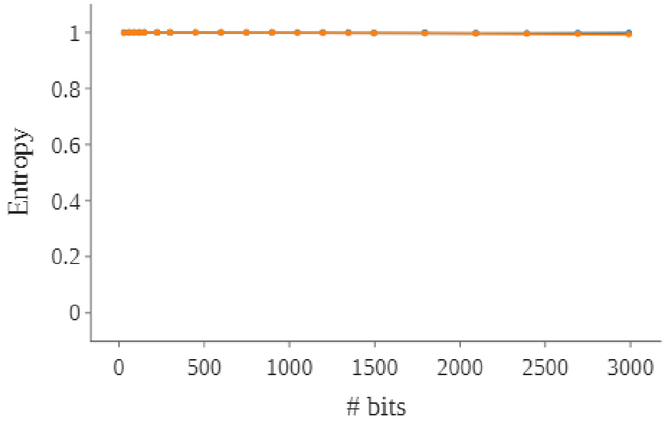
(a) FIR



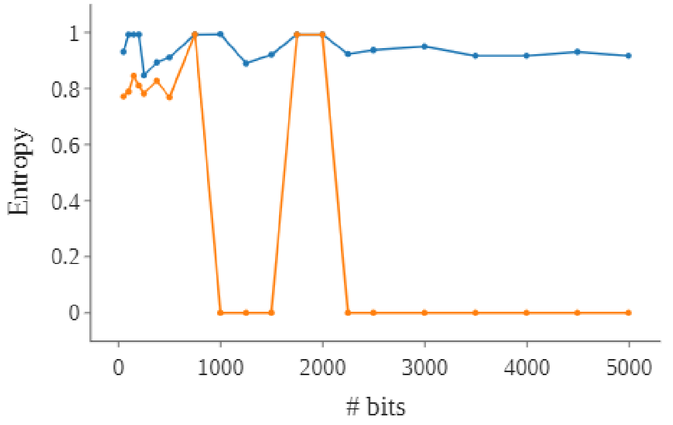
(b) IIR



(c) MD5

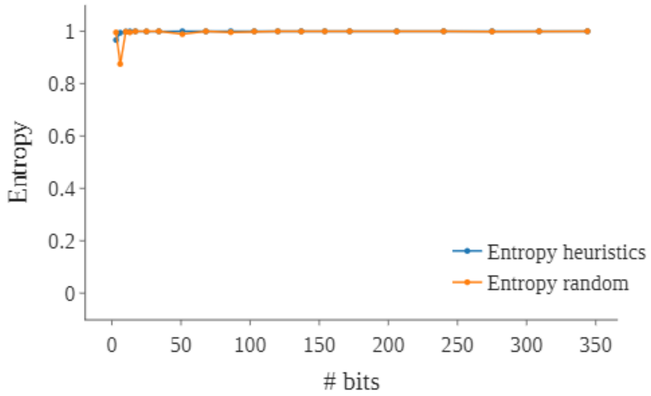


(d) DES3

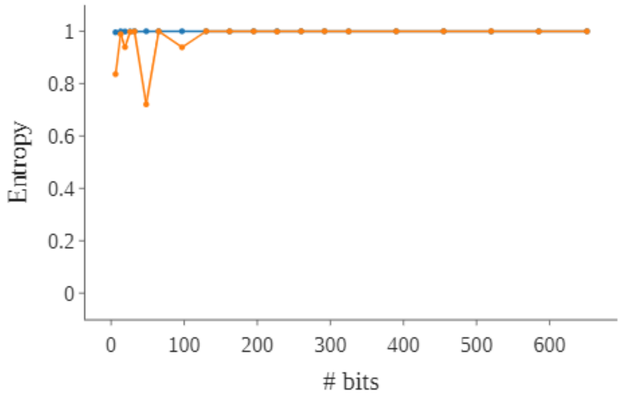


(e) SHA256

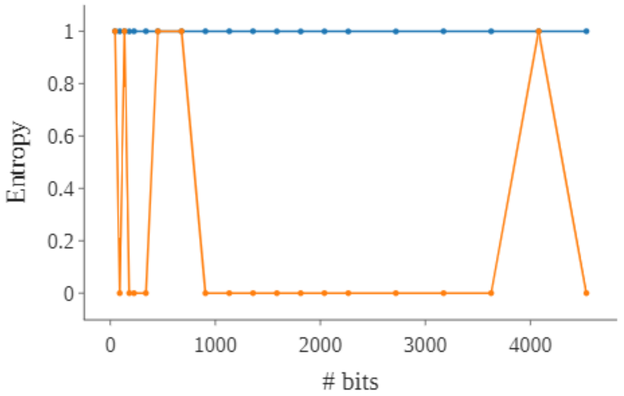
Additional Results



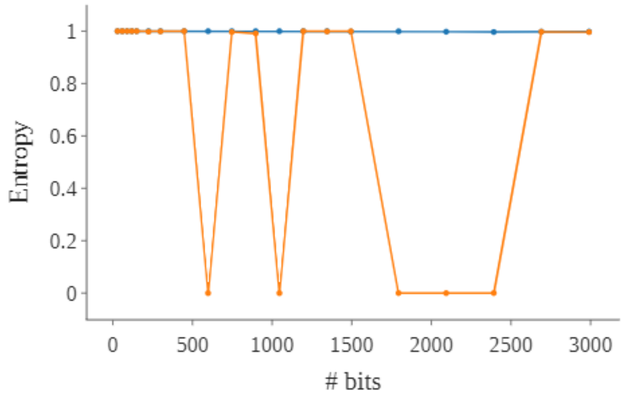
(a) FIR



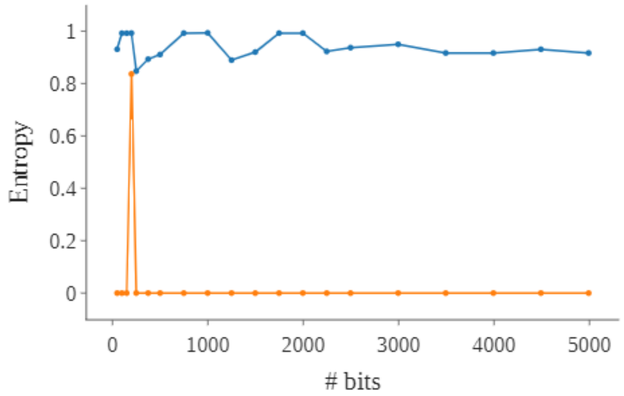
(b) IIR



(c) MD5



(d) DES3



(e) SHA256

Errata Corrige

Design	Area Overhead m.r.e. [%]
FIR	25.07
IIR	26.93
MD5	179.47
DES3	123.87
SHA256	70.36
ALL	75.20

Design	Area Overhead m.a.e. [%]
FIR	12.24
IIR	18.39
MD5	84.87
DES3	41.95
SHA256	70.36
ALL	75.20