Centrality Indicators For Efficient And Scalable Logic Masking

Brice Colombier

—

Lilian Bossuet, David Hély

Univ Lyon, UJM-Saint-Etienne, CNRS
Laboratoire Hubert Curien UMR 5516
F-42023, SAINT-ETIENNE, France

June 19, 2017

LABORATOIRE HUBERT CURIEN
UMR • CNRS • 5516 • SAINT-ETIENNE

RhôneAlpes Région

SALWARE
French ANR Project

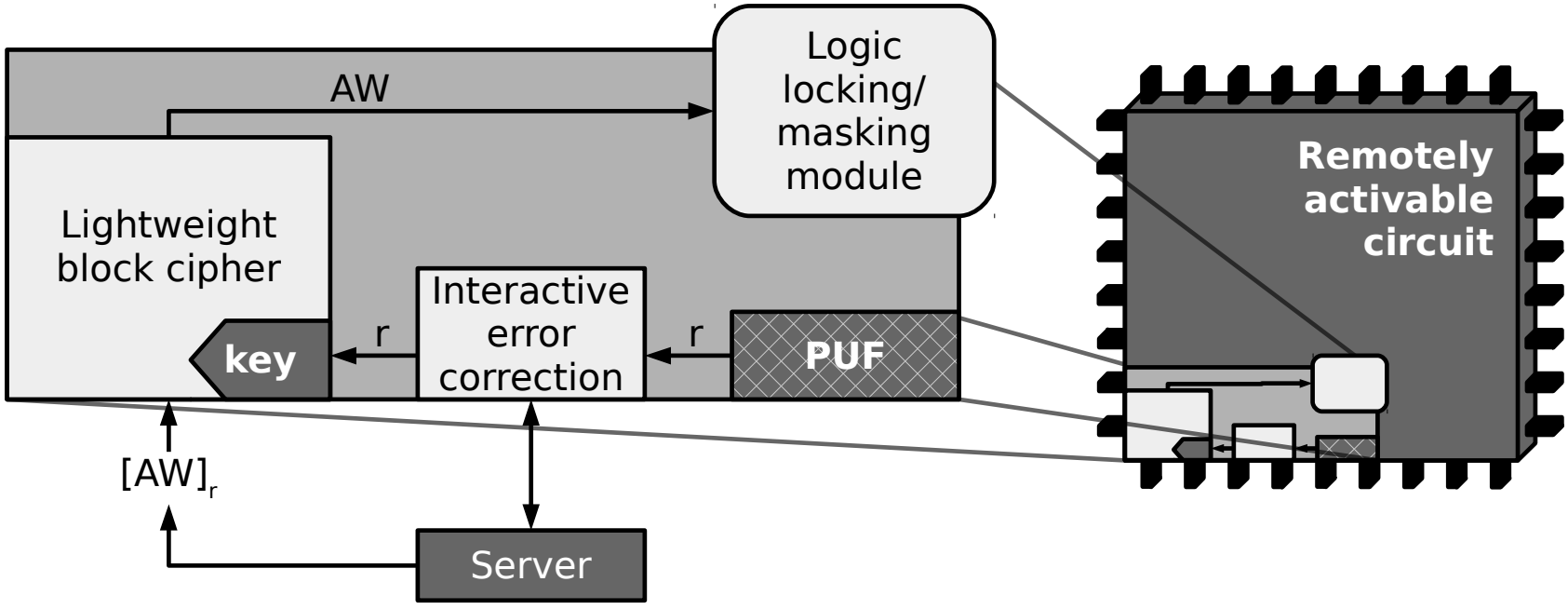
Objective:

Fight against **counterfeiting** and **illegal copying** of integrated circuits and IP cores.

Design data protection

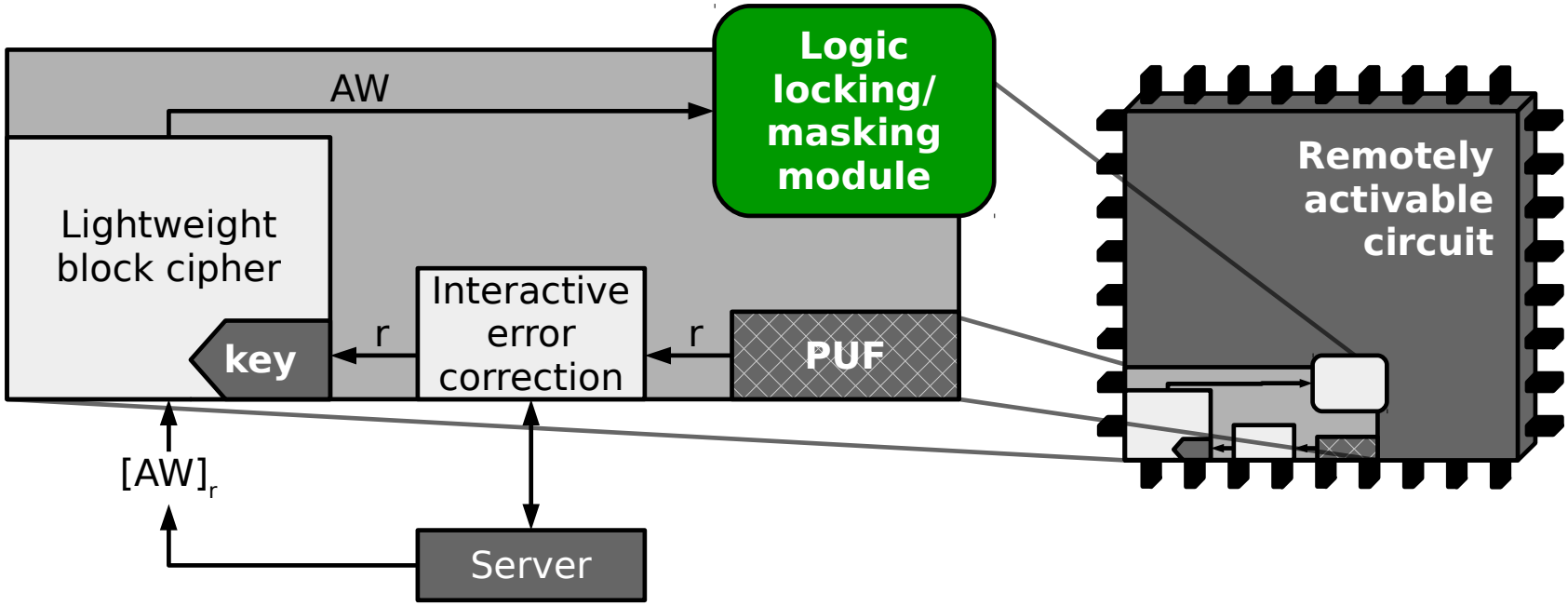
Objective:

Fight against **counterfeiting** and **illegal copying** of integrated circuits and IP cores.



Objective:

Fight against **counterfeiting** and **illegal copying** of integrated circuits and IP cores.



Proposition:

Before it has been **unlocked**, the circuit must operate as **abnormally** as possible, until the **correct activation word** is fed.

Proposition:

Before it has been **unlocked**, the circuit must operate as **abnormally** as possible, until the **correct activation word** is fed.

Solutions:

We want to be able to **controllably**:

- Force the outputs to a fixed logic value: **locking**[1].
- Alter the outputs as much as possible: **masking**[2].

---

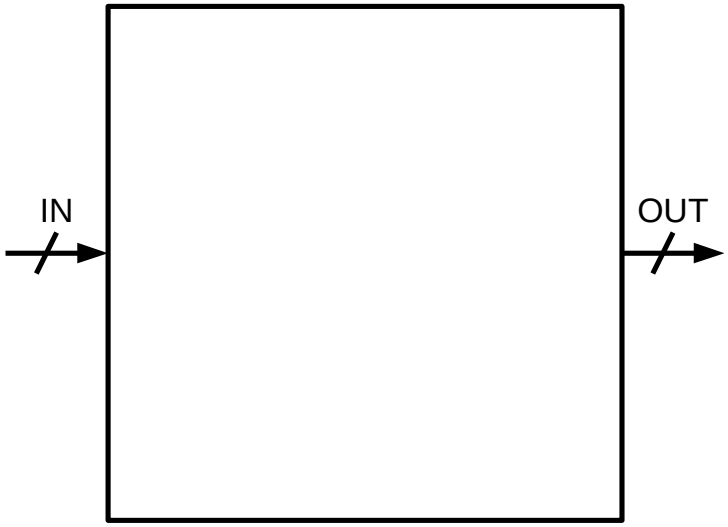
[1] B. Colombier, L. Bossuet, and D. Hély. "Reversible Denial-of-Service by Locking Gates Insertion for IP Cores Design Protection". *Cryptarchi*. 2015.

[2] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

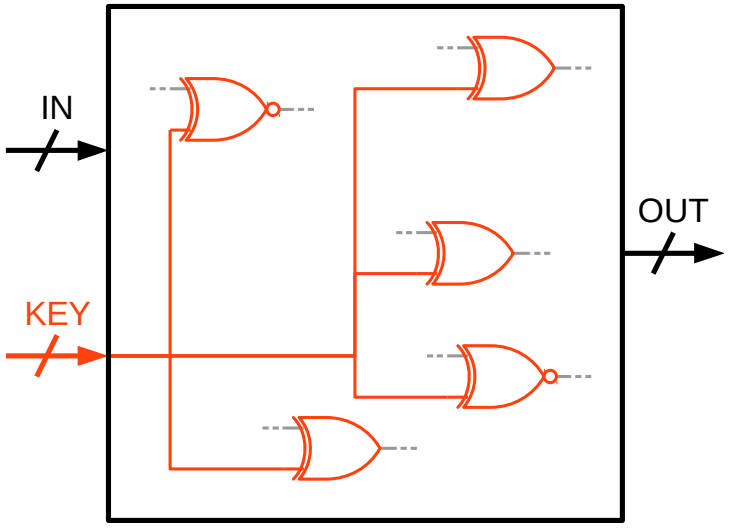
Combinational logic masking

Principle

Insert **XOR/XNOR** gates at **specific** locations in the netlist, so that internal nodes can be controllably inverted to alter the internal state if the wrong activation word is fed.



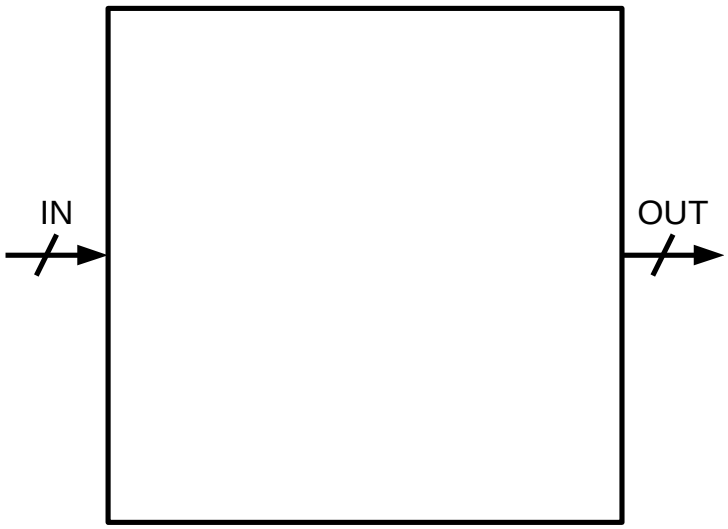
Original netlist



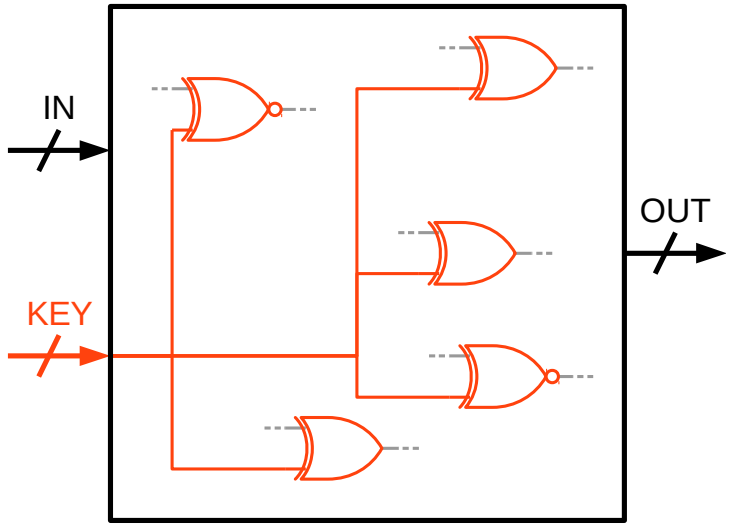
Masked netlist

Principle

Insert **XOR/XNOR** gates at **specific** locations in the netlist, so that internal nodes can be controllably inverted to alter the internal state if the wrong activation word is fed.



Original netlist



Masked netlist

Done at design time

The process should be as fast as possible.

Metrics to measure the masking efficiency

Notation:

AW: n-bit activation input.

AW_{ref}: Reference activation word.

S: Correct outputs of the netlist.

S_{mod}: Masked outputs of the netlist.

Metrics based on:

Metrics to measure the masking efficiency

Notation:

AW: n-bit activation input.

AW_{ref}: Reference activation word.

S: Correct outputs of the netlist.

S_{mod}: Masked outputs of the netlist.

Metrics based on:

- Corruptibility: $\boxed{P(S_{mod} = S | AW \neq AW_{ref}) = 0}$

Metrics to measure the masking efficiency

Notation:

AW: n-bit activation input.

AW_{ref}: Reference activation word.

S: Correct outputs of the netlist.

S_{mod}: Masked outputs of the netlist.

Metrics based on:

- Corruptibility: $\boxed{P(S_{mod} = S | AW \neq AW_{ref}) = 0}$
- Hamming distance: $\boxed{\frac{1}{2^n} \sum_{AW=0}^{2^n-1} HD(S, S_{mod}) = 50\%}$

Metrics to measure the masking efficiency

Notation:

AW: n-bit activation input.

AW_{ref}: Reference activation word.

S: Correct outputs of the netlist.

S_{mod}: Masked outputs of the netlist.

Metrics based on:

- Corruptibility: $\boxed{P(S_{mod} = S | AW \neq AW_{ref}) = 0}$
- Hamming distance: $\boxed{\frac{1}{2^n} \sum_{AW=0}^{2^n-1} HD(S, S_{mod}) = 50\%}$
- Bitwise correlation: $\boxed{\sqrt{\frac{1}{n} \sum_{i=0}^{\#\text{outputs}-1} \rho^2 \left(S[i], S_{mod}[i]\right)} = 0}$

Notation:

AW: n-bit activation input.

AW_{ref}: Reference activation word.

S: Correct outputs of the netlist.

S_{mod}: Masked outputs of the netlist.

Metrics based on:

- Corruptibility: $\boxed{P(S_{mod} = S | AW \neq AW_{ref}) = 0}$
- Hamming distance: $\boxed{\frac{1}{2^n} \sum_{AW=0}^{2^n-1} HD(S, S_{mod}) = 50\%}$
- Bitwise correlation: $\boxed{\sqrt{\frac{1}{n} \sum_{i=0}^{\#\text{outputs}-1} \rho^2 \left(S[i], S_{mod}[i]\right)} = 0}$

How to **optimise** these metrics ?

Several heuristics exist to select the place of insertion:

Selection heuristic
Random[1]

[1] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

[2] R. S. Chakraborty and S. Bhunia. "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection". *IEEE Trans. on CAD of IC and Syst.* (2009).

[3] J. Rajendran et al. "Security analysis of logic obfuscation". *DAC*. 2012.

[4] J. Rajendran et al. "Security analysis of integrated circuit camouflaging". *ACM Conference on Computer & communications security*. 2013.

[5] J. Rajendran et al. "Fault Analysis-Based Logic Encryption". *IEEE Transactions on Computers* (2015).

Several heuristics exist to select the place of insertion:

Selection heuristic
Random[1]
Fan-in/fan-out cones[2]

[1] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

[2] R. S. Chakraborty and S. Bhunia. "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection". *IEEE Trans. on CAD of IC and Syst.* (2009).

[3] J. Rajendran et al. "Security analysis of logic obfuscation". *DAC*. 2012.

[4] J. Rajendran et al. "Security analysis of integrated circuit camouflaging". *ACM Conference on Computer & communications security*. 2013.

[5] J. Rajendran et al. "Fault Analysis-Based Logic Encryption". *IEEE Transactions on Computers* (2015).

Nodes selection heuristics

Several heuristics exist to select the place of insertion:

Selection heuristic
Random[1]
Fan-in/fan-out cones[2]
Random + interf. graph[3]

[1] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

[2] R. S. Chakraborty and S. Bhunia. "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection". *IEEE Trans. on CAD of IC and Syst.* (2009).

[3] J. Rajendran et al. "Security analysis of logic obfuscation". *DAC*. 2012.

[4] J. Rajendran et al. "Security analysis of integrated circuit camouflaging". *ACM Conference on Computer & communications security*. 2013.

[5] J. Rajendran et al. "Fault Analysis-Based Logic Encryption". *IEEE Transactions on Computers* (2015).

Nodes selection heuristics

Several heuristics exist to select the place of insertion:

Selection heuristic
Random[1]
Fan-in/fan-out cones[2]
Random + interf. graph[3]
Random + interf. graph + corrup.[4]

[1] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

[2] R. S. Chakraborty and S. Bhunia. "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection". *IEEE Trans. on CAD of IC and Syst.* (2009).

[3] J. Rajendran et al. "Security analysis of logic obfuscation". *DAC*. 2012.

[4] J. Rajendran et al. "Security analysis of integrated circuit camouflaging". *ACM Conference on Computer & communications security*. 2013.

[5] J. Rajendran et al. "Fault Analysis-Based Logic Encryption". *IEEE Transactions on Computers* (2015).

Nodes selection heuristics

Several heuristics exist to select the place of insertion:

Selection heuristic
Random[1]
Fan-in/fan-out cones[2]
Random + interf. graph[3]
Random + interf. graph + corrup.[4]
Fault analysis[5]

[1] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

[2] R. S. Chakraborty and S. Bhunia. "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection". *IEEE Trans. on CAD of IC and Syst.* (2009).

[3] J. Rajendran et al. "Security analysis of logic obfuscation". *DAC*. 2012.

[4] J. Rajendran et al. "Security analysis of integrated circuit camouflaging". *ACM Conference on Computer & communications security*. 2013.

[5] J. Rajendran et al. "Fault Analysis-Based Logic Encryption". *IEEE Transactions on Computers* (2015).

Nodes selection heuristics

Several heuristics exist to select the place of insertion:

Selection heuristic	Masking efficiency
Random[1]	✗
Fan-in/fan-out cones[2]	✗
Random + interf. graph[3]	✗
Random + interf. graph + corrup.[4]	✗
Fault analysis[5]	✓

[1] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

[2] R. S. Chakraborty and S. Bhunia. "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection". *IEEE Trans. on CAD of IC and Syst.* (2009).

[3] J. Rajendran et al. "Security analysis of logic obfuscation". *DAC*. 2012.

[4] J. Rajendran et al. "Security analysis of integrated circuit camouflaging". *ACM Conference on Computer & communications security*. 2013.

[5] J. Rajendran et al. "Fault Analysis-Based Logic Encryption". *IEEE Transactions on Computers* (2015).

Nodes selection heuristics

Several heuristics exist to select the place of insertion:

Selection heuristic	Masking efficiency	Computational complexity
Random[1]	✗	✓
Fan-in/fan-out cones[2]	✗	✓
Random + interf. graph[3]	✗	✓
Random + interf. graph + corrup.[4]	✗	✓
Fault analysis[5]	✓	✗

[1] J. A. Roy, F. Koushanfar, and I. Markov. "EPIC: Ending Piracy of Integrated Circuits". *DATE*. 2008.

[2] R. S. Chakraborty and S. Bhunia. "HARPOON: An Obfuscation-Based SoC Design Methodology for Hardware Protection". *IEEE Trans. on CAD of IC and Syst.* (2009).

[3] J. Rajendran et al. "Security analysis of logic obfuscation". *DAC*. 2012.

[4] J. Rajendran et al. "Security analysis of integrated circuit camouflaging". *ACM Conference on Computer & communications security*. 2013.

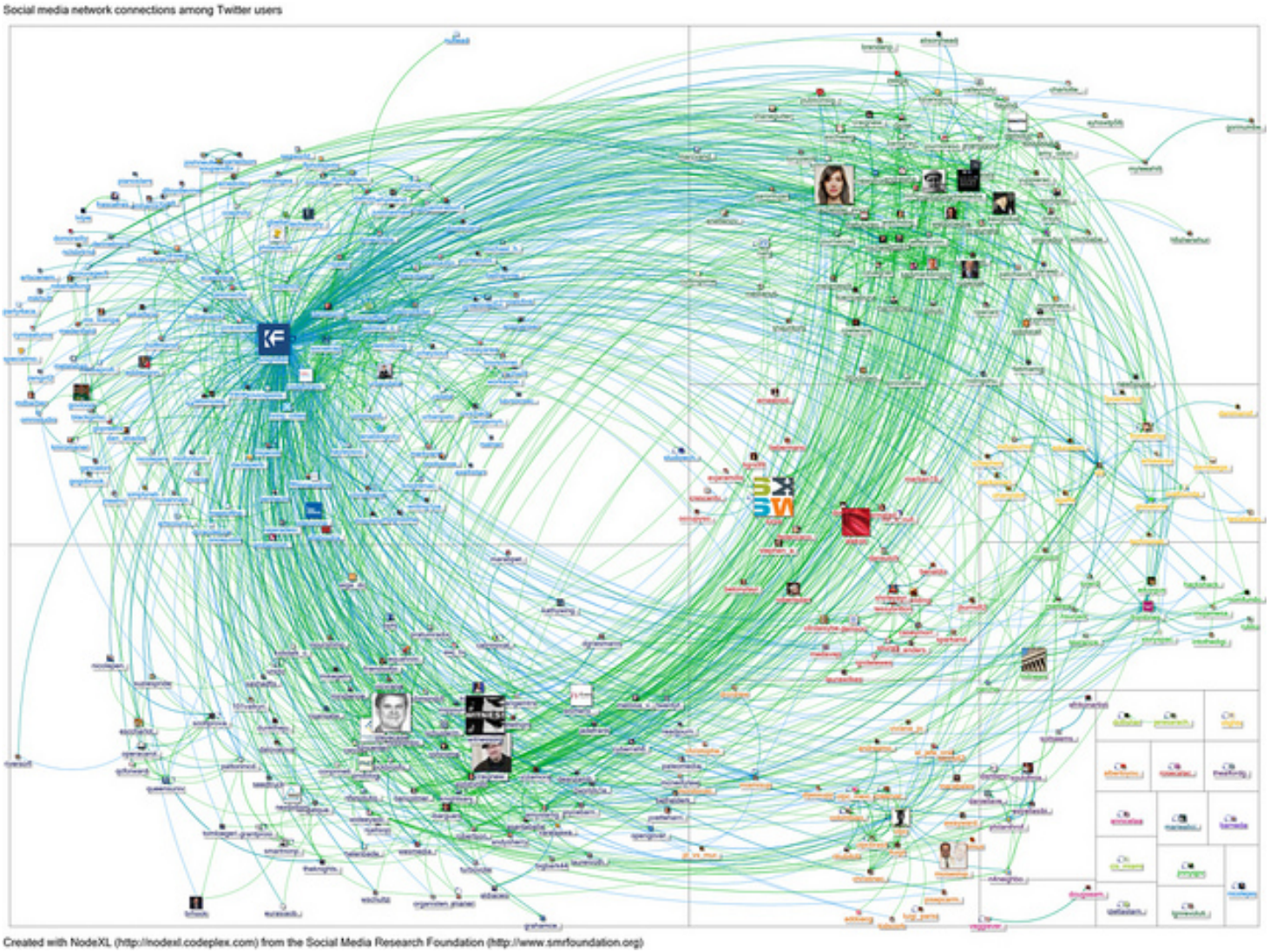
[5] J. Rajendran et al. "Fault Analysis-Based Logic Encryption". *IEEE Transactions on Computers* (2015).

Towards a better trafe-off between masking efficiency and computational complexity

Centrality: **Importance** of a given vertex inside a graph.

Centrality indicators

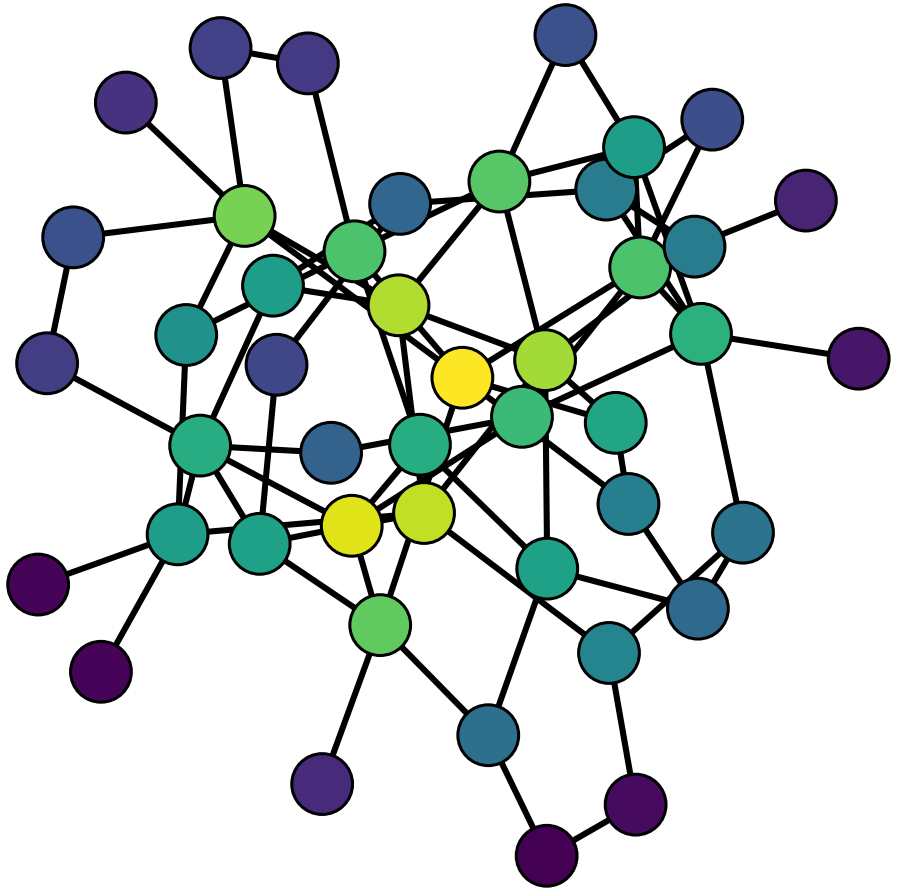
Centrality: **Importance** of a given vertex inside a graph.



Centrality: **Importance** of a given vertex inside a graph.

Centrality:

- Closeness,



Definition:

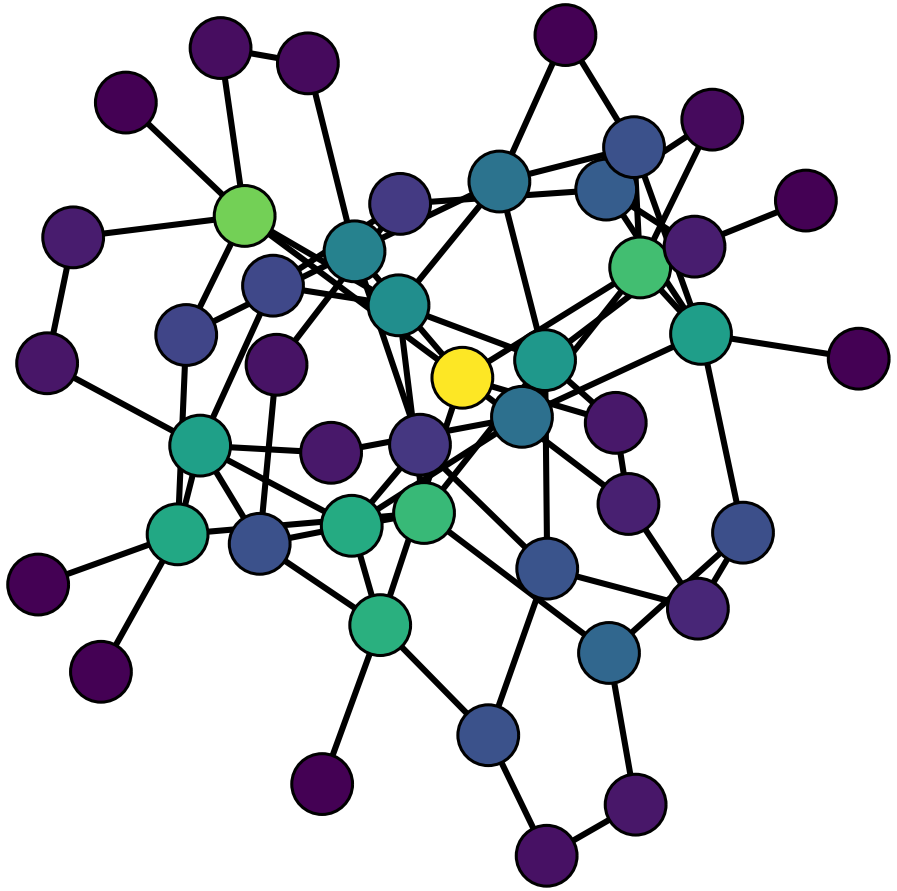
Inverse of the sum of distances to all the other vertices.

$$C(v) = \frac{1}{\sum_{y:y \in V} d(v,y)}$$

Centrality: **Importance** of a given vertex inside a graph.

Centrality:

- Closeness,
- Betweenness,



Definition:

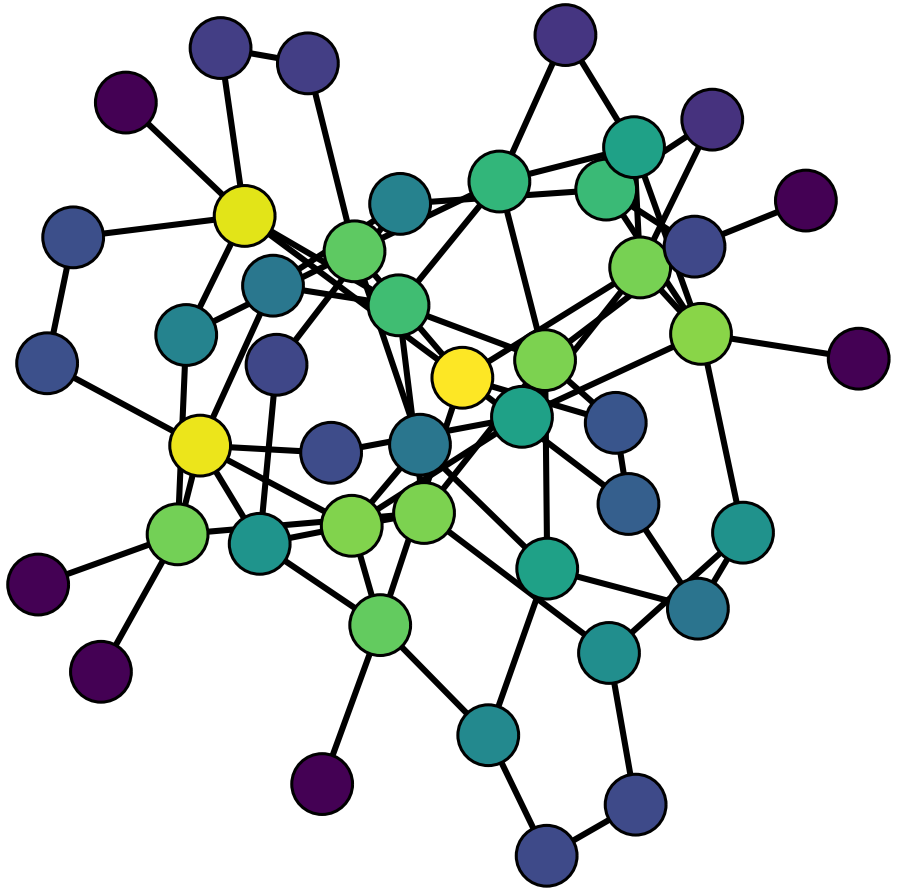
Ratio of shortest paths going through this vertex.

$$C(v) = \sum_{s \neq t \neq v : \{s,t,v\} \in V} \frac{\sigma_{svt}}{\sigma_{st}}$$

Centrality: **Importance** of a given vertex inside a graph.

Centrality:

- Closeness,
- Betweenness,
- Current-flow betweenness,



Definition:

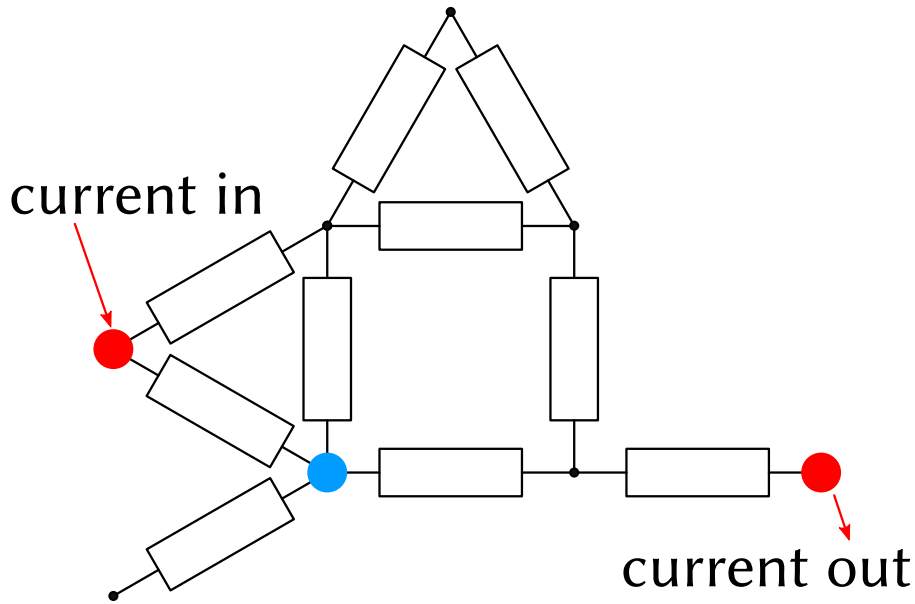
Current flowing through this vertex with others as source and sink.

$$C(v) = \sum_{s \neq t : \{s,t\} \in V} I_v^{(st)}$$

Centrality: **Importance** of a given vertex inside a graph.

Centrality:

- Closeness,
- Betweenness,
- Current-flow betweenness,



Definition:

Current flowing through this vertex with others as source and sink.

$$C(v) = \sum_{s \neq t : \{s,t\} \in V} I_v^{(st)}$$

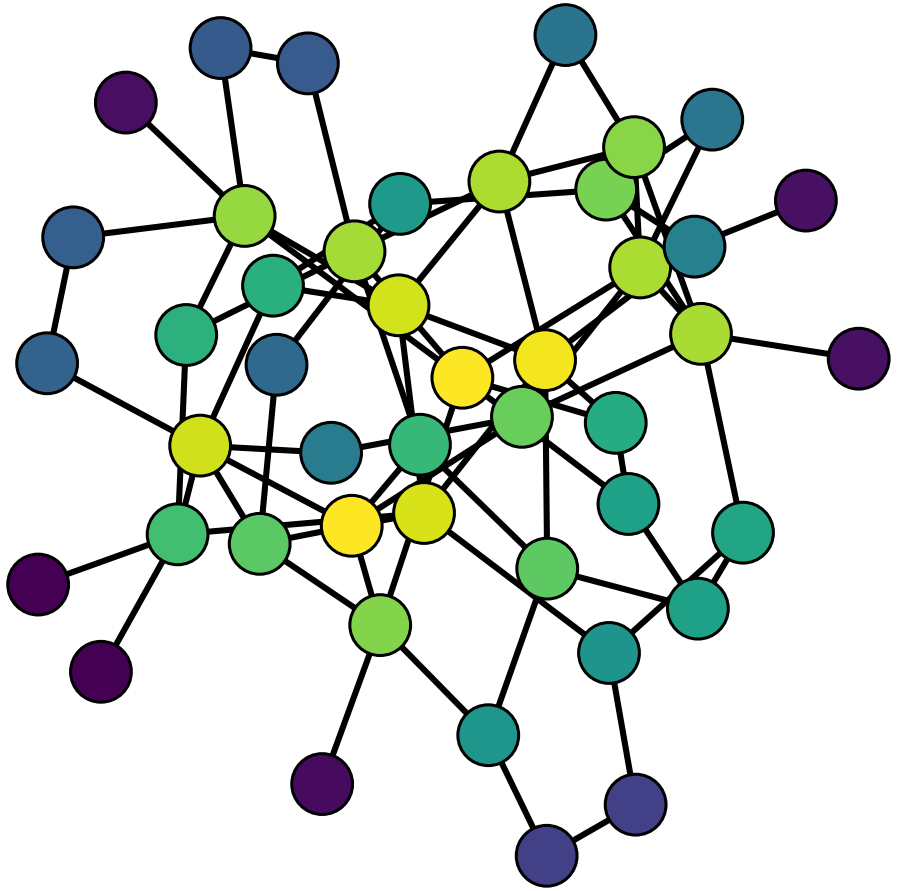
Centrality: **Importance** of a given vertex inside a graph.

Centrality:

- Closeness,
- Betweenness,
- Current-flow betweenness,
- Current-flow closeness,

Definition:

Sum of effective resistances to all other vertices.



$$C(v) = \frac{1}{\sum_{y:y\in V} p(v)-p(y)} = \frac{1}{\sum_{y:y\in V} R_{eq}(v,y)}$$

Indicators that only account for shortest paths

Closeness, betweenness.

Comparison of centrality indicators

Indicators that only account for shortest paths

Closeness, betweenness.

Indicators that weigh paths according to their length

Current-flow betweenness and closeness centralities.

Indicators that only account for shortest paths

Closeness, betweenness.

Indicators that weigh paths according to their length

Current-flow betweenness and closeness centralities.

Increasing computational complexity

Closeness

↓

Betweenness

↓

Current-flow closeness centrality

↓

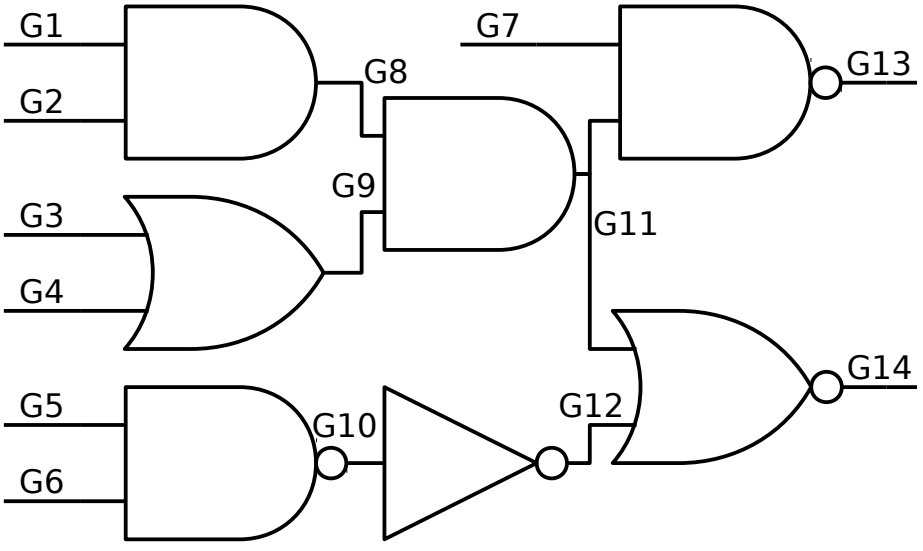
Approximated current-flow betweenness centrality

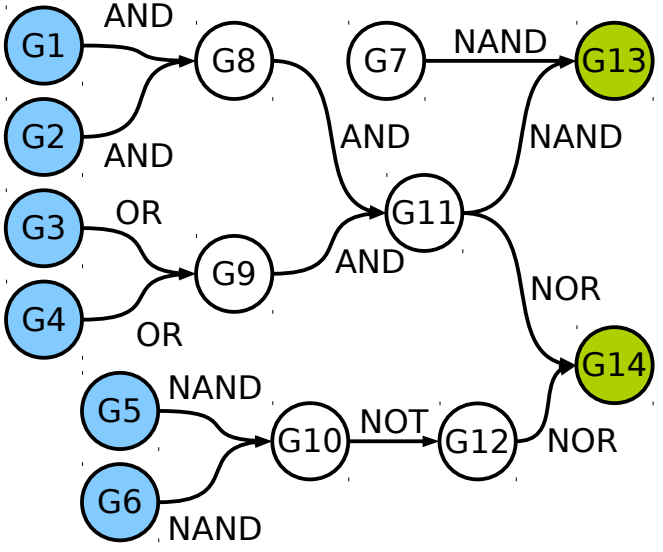
↓

Current-flow betweenness centrality

Convert the netlist into a graph

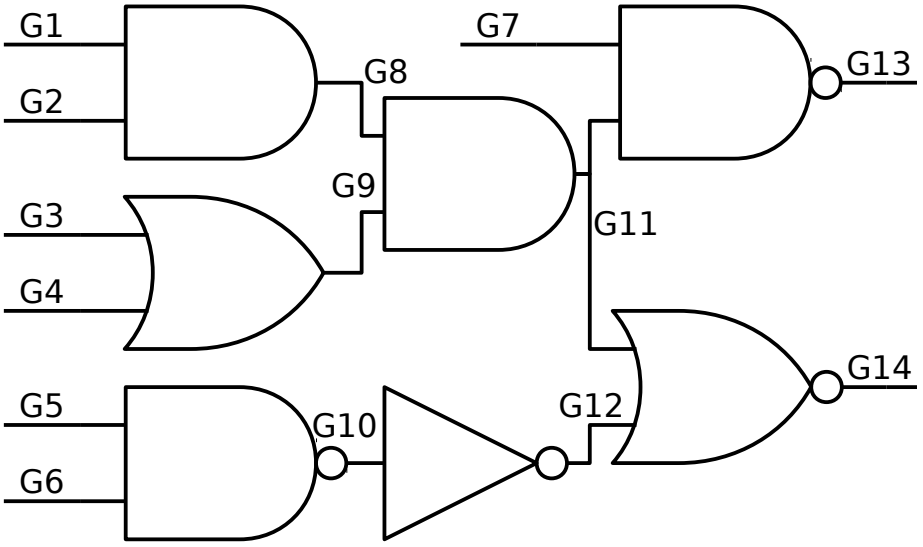
Nodes $\rightarrow$ vertices
Gates $\rightarrow$ edges

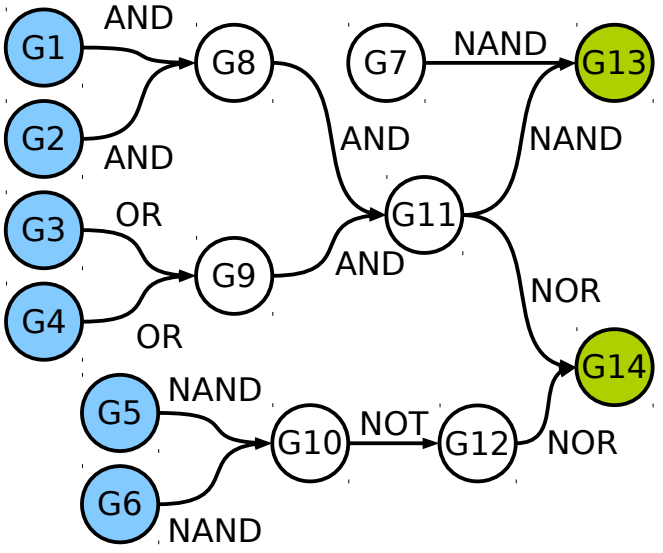




Convert the netlist into a graph

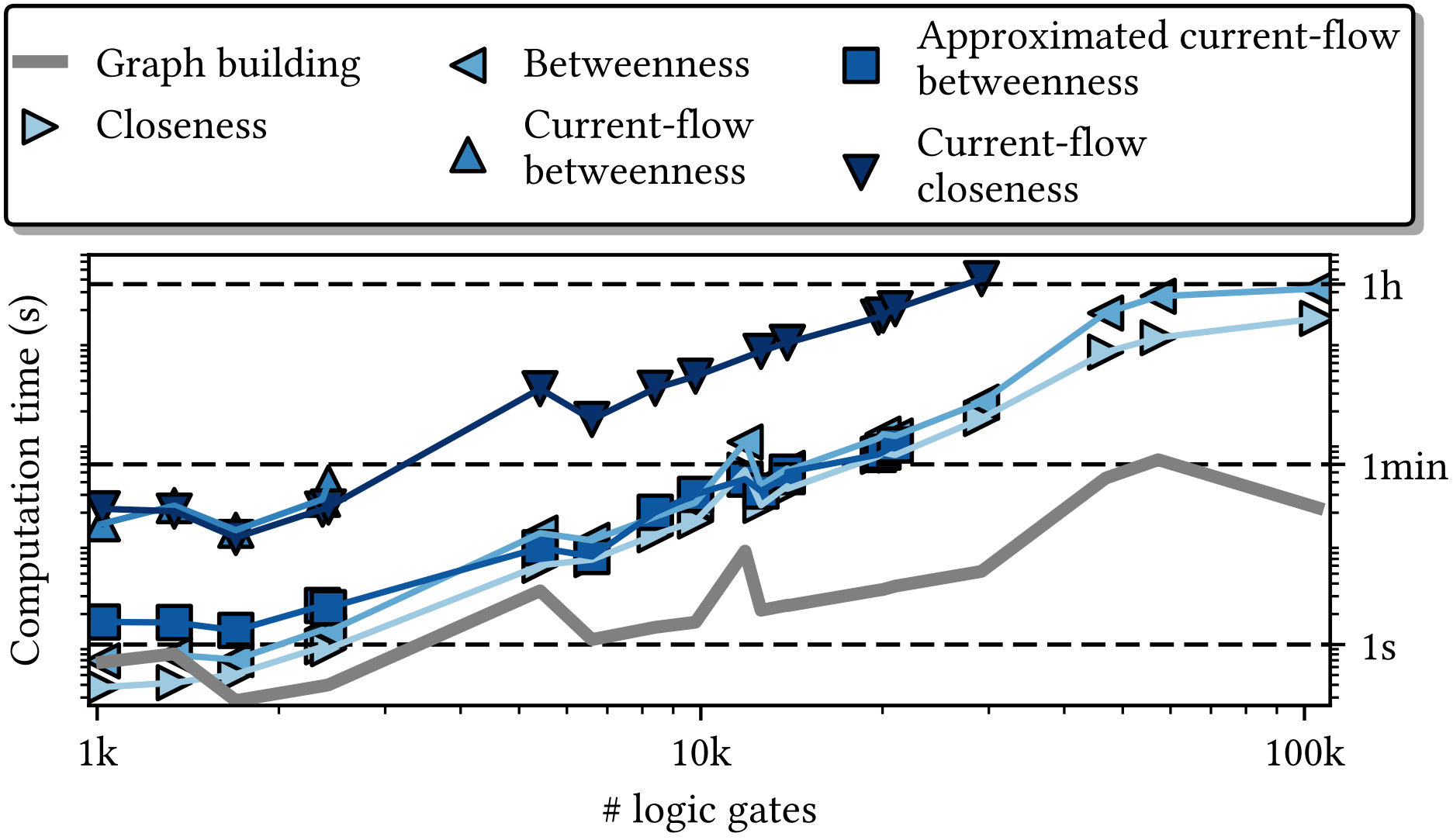
Nodes $\rightarrow$ vertices
Gates $\rightarrow$ edges

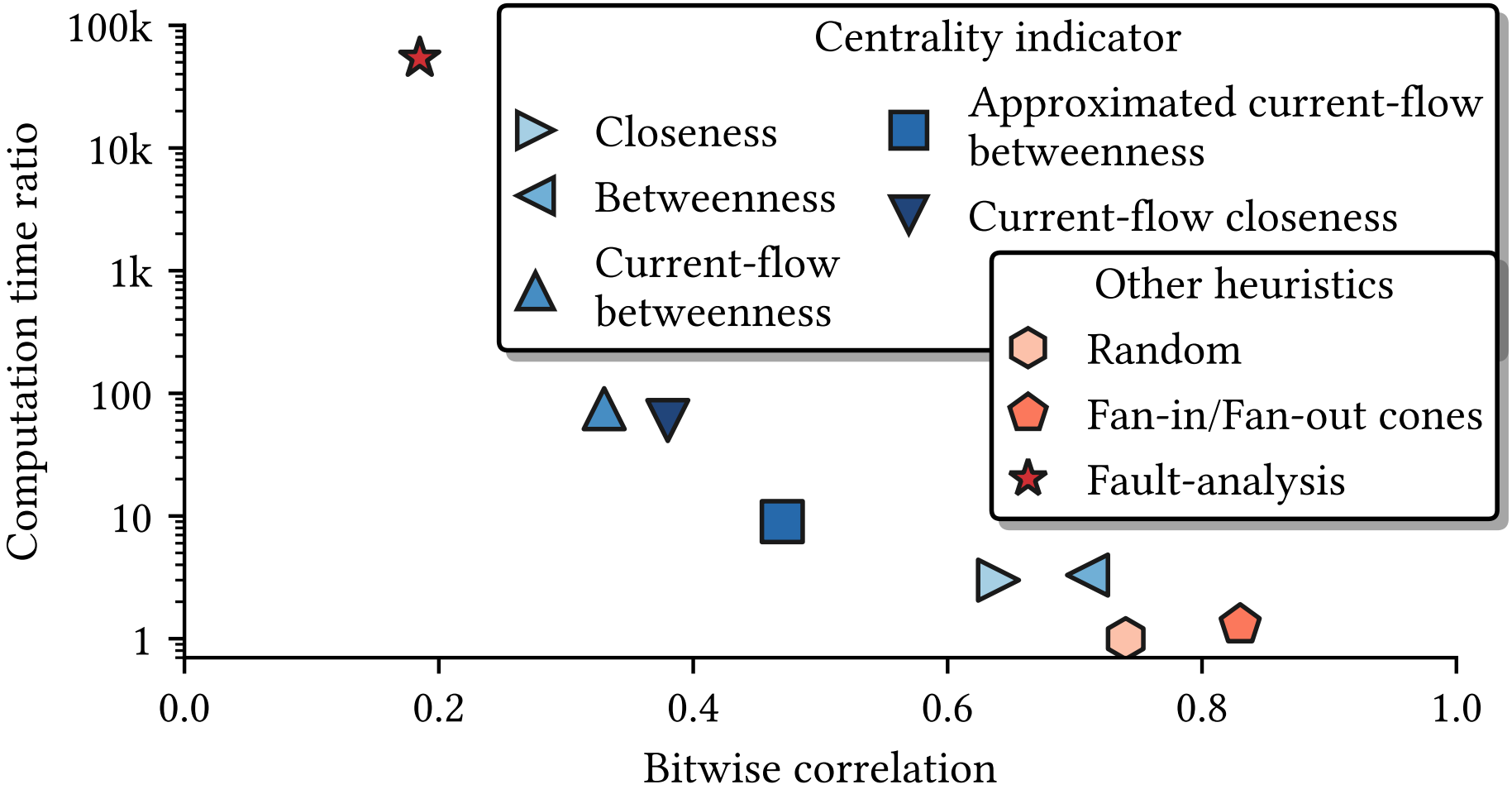


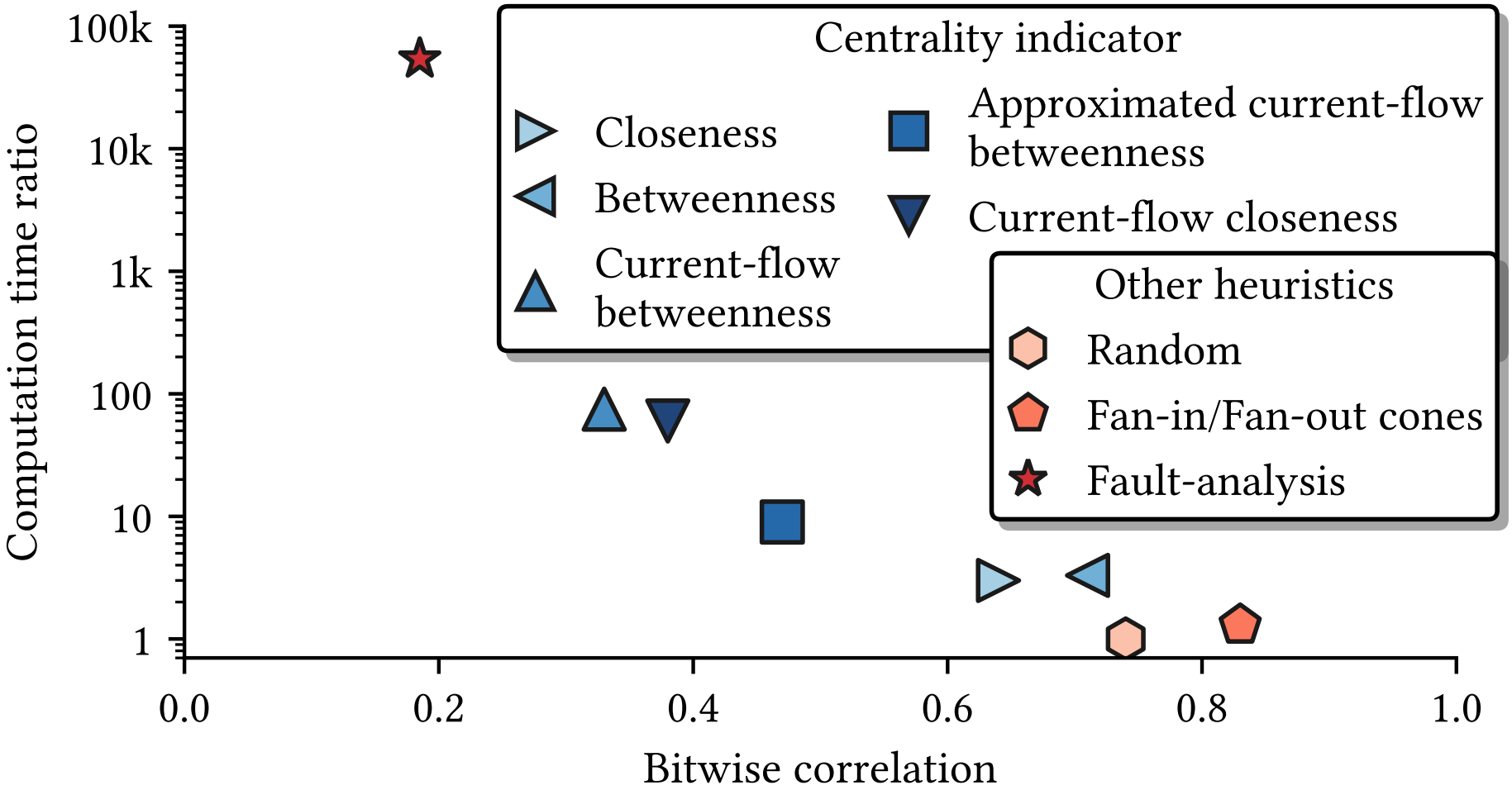


XOR/XNOR gates are inserted on the vertices for which the chosen centrality indicator is the **highest**.

Computation time




Centrality indicator
Closeness
Betweenness
Current-flow betweenness
Approximated current-flow betweenness
Current-flow closeness
Other heuristics
Random
Fan-in/Fan-out cones
Fault-analysis
Computation time ratio
100k
10k
1k
100
10
1
0.0
0.2
0.4
0.6
0.8
1.0
Bitwise correlation



Conclusion

Bitwise correlation is **almost as low** as for the fault analysis-based heuristic, for a run-time **1000x shorter**.

Centrality indicators, in particular current-flow closeness centrality, as node selection heuristic for logic masking:

- allow to **disturb** the outputs **efficiently**,
- have a **manageable** computational **complexity**,
- offer a **better trade-off** than existing heuristics,
- could be **parallelized**...

Centrality indicators, in particular current-flow closeness centrality, as node selection heuristic for logic masking:

- allow to **disturb** the outputs **efficiently**,
- have a **manageable** computational **complexity**,
- offer a **better trade-off** than existing heuristics,
- could be **parallelized**...

— Questions ? —