An EDA Flow for Reliability Driven Logic Synthesis

Mr. Satish Grandhi

3rd year PhD Student

School of Electrical Engineering

ucc

Coláiste na hOllscoile Corcaigh, Éire

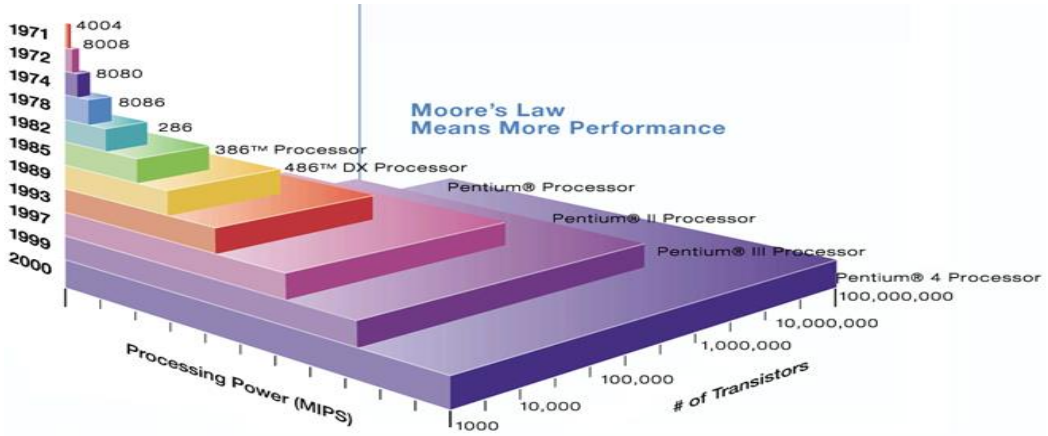
University College Cork, Ireland

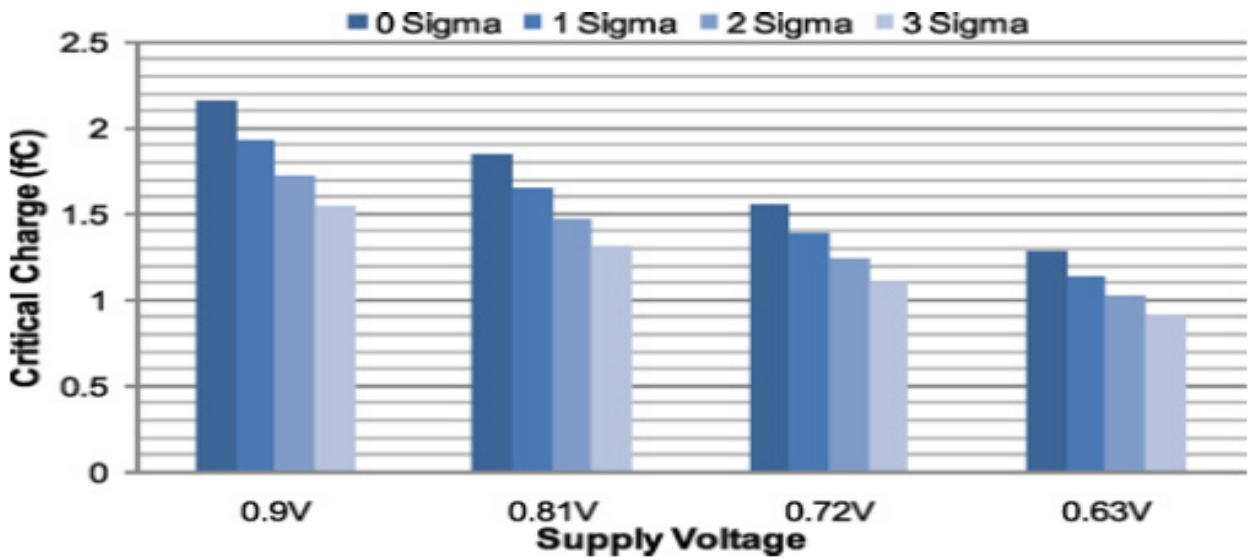
Contents

- Introduction
- Reliability Estimation
  - Gate Error Model
  - CPEP Algorithm
- Reliability Optimization
  - Rule based Rewriting
  - Cut Based Rewriting
- Results & Conclusion

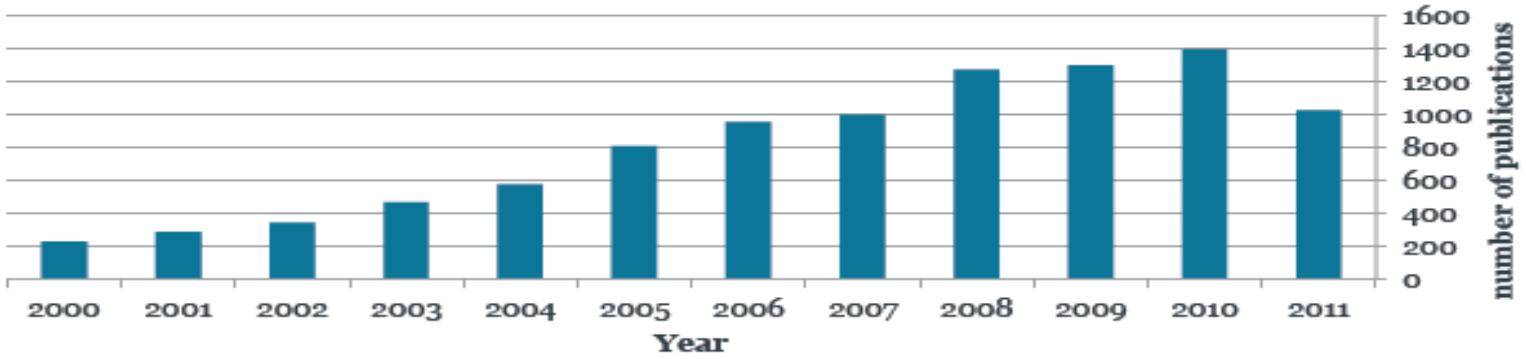
Introduction

- Shrinking CMOS technology.
- Lower power supplies.
- Voltage scaling and process variation degrades reliability.
- Promising post silicon devices like CNFET's are inherently unreliable due to statistical variations.

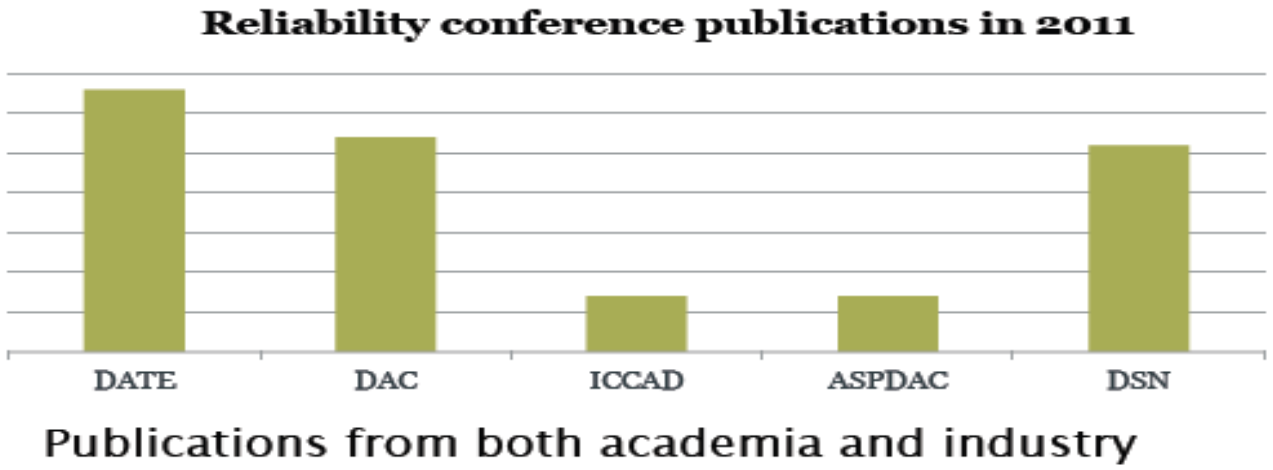




Reliability Publications



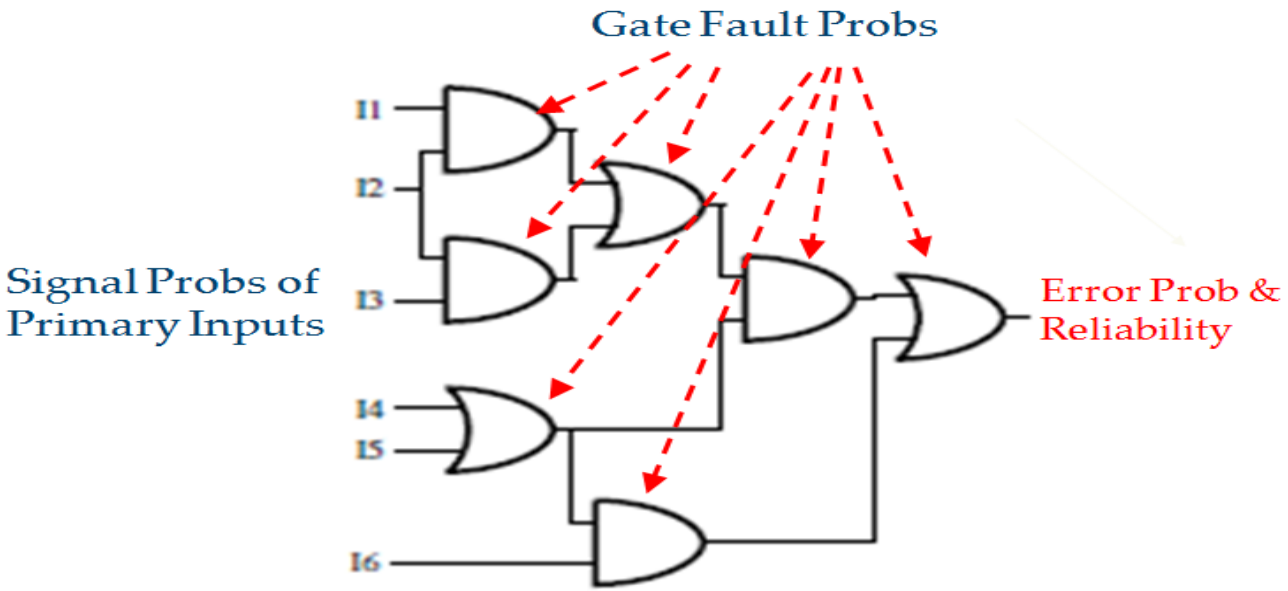
9000+ publications over the past 12 years



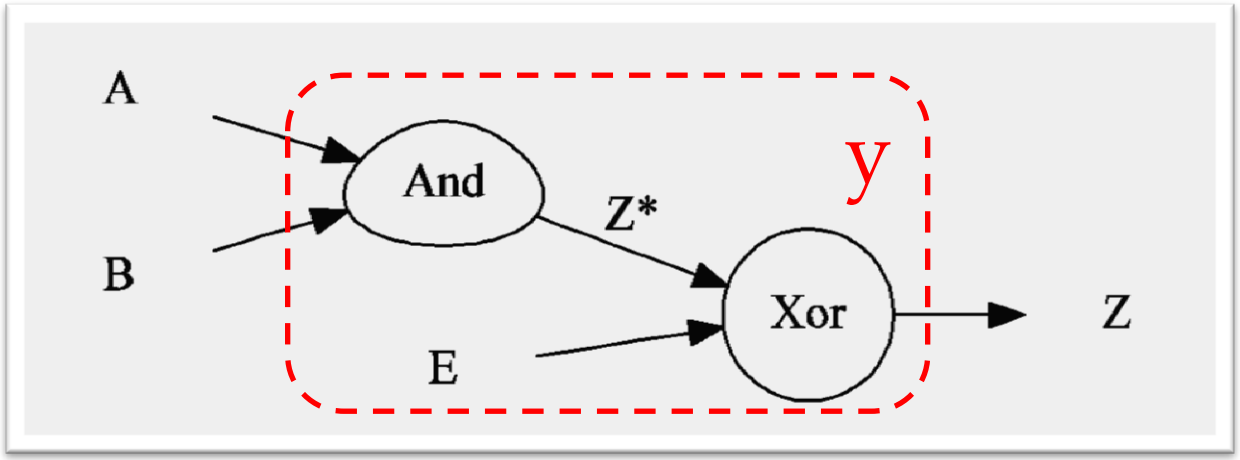
Ref : Bashir M. Al-Hashimi, “Hardware Reliability of Embedded Systems: Are We There Yet?”, Designing with Uncertainty - Opportunities & Challenges workshop York, UK, 17-19th March 2014.

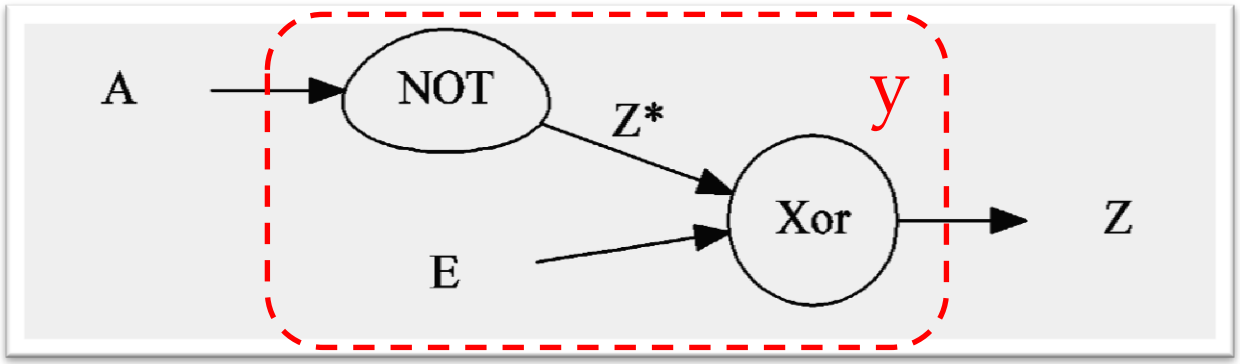
Introduction -- Errors and Propagation

- 3 masking effects in combinational logic to see if soft errors are propagated or not.
  - **Logical Masking** : Off path gate inputs prevent logical transition on output node {Major focus of my research}.
  - **Temporal Masking :** An erroneous pulse is masked if it does not occur close enough to latch trigger point {Trying to add this to the tool}.
  - **Electrical Masking** : If signal is attenuated by electrical properties of the gate. {Not studied}



Probabilistic Gate Model





- eXOR performs the stochastic operations of Errors
- SPZ* = P[Z* = 1] -- Static Probability
- EPZ* = P[Z* output is incorrect]
- Pf(y) = P[gate y is faulty] = SPE
- EP(y) = EPZ = f(EPZ*, Pf(y))

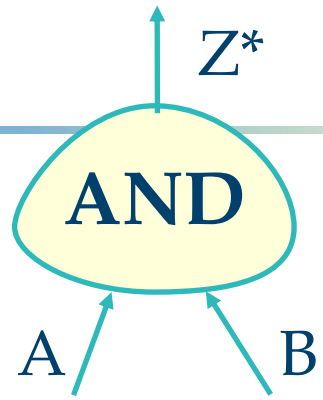
Where,

- Z* --- Ideal Output
- Z --- Error-prone Output
- E --- Error Injection Input

E = 1 -- Gate is faulty

E = 0 -- Gate is correct

Probabilistic Gate Model



- Error-prone Version of Truth Table

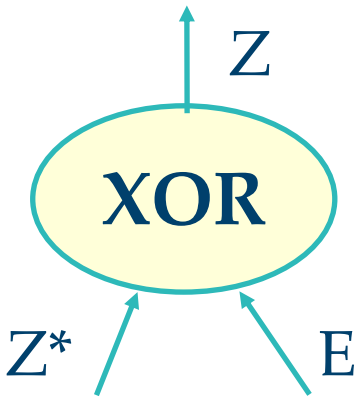
Actual Value of A	State/ Correct Value of A		Actual Value of B	State/ Correct Value of B		Actual Value of Z*	State/ Correct Value of Z*		Error Terms
0	c	0	0	c	0	0	c	0	
	c	0		ε	1		c	0	
	ε	1		c	0		c	0	
	ε	1		ε	1		ε	1	P[A0, Aε, B0, Bε]
	c	0	1	c	1	0	c	0	
	c	0		ε	0		c	0	
	ε	1		c	1		ε	1	P[A0, Aε, B1, Bc]
	ε	1		ε	0		c	0	
1	c	1	0	c	0	0	c	0	
	c	1		ε	1		ε	1	P[A1, Ac, B0, Bε]
	ε	0		c	0		c	0	
	ε	0		ε	1		c	0	
	c	1	1	c	1	1	c	1	
	c	1		ε	0		ε	0	P[A1, Ac, B1, Bε]
	ε	0		c	1		ε	0	P[A1, Aε, B1, Bc]
	ε	0		ε	0		ε	0	P[A1, Aε, B1, Bε]

- SPZ*: SPZ* = P[A1, B1]
- Joint Probs:
  - CPZ*0 = P[Z*0, Z*c] = P[Z*0] – (t1+t2+t3)
  - CPZ*1 = P[Z*0, Z*ε] = t1+t2+t3
  - CPZ*2 = P[Z*1, Z*c] = P[A1, Ac, B1, Bc]
  - CPZ*3 = P[Z*1, Z*ε] = t4+t5+t6
- Therefore,
  - EPZ* = P[Z*0, Z*ε] + P[Z*1, Z*ε]
    = CPZ1 + CPZ3
    = t1+t2+t3+t4+t5+t6

Probabilistic Gate Model

- Error-prone Version of Truth Table

Actual value of Z*	State of Z*	Gate Fault	Value of Z	correct value of Z	State of Z	Prob.
0	c	c	0	0	c	
	c	f	1	0	ε	← $P[Z^*=0, Z^*_c, P_F]$
	ε	c	0	1	ε	← $P[Z^*=0, Z^*_\epsilon, P_c]$
	ε	f	1	1	c	
1	c	c	1	1	c	
	c	f	0	1	ε	← $P[Z^*=1, Z^*_c, P_F]$
	ε	c	1	0	ε	← $P[Z^*=1, Z^*_\epsilon, P_c]$
	ε	f	0	0	c	

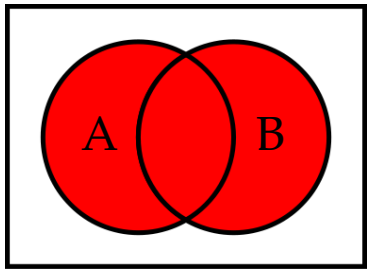


- Ec -- E1 -- Gate is faulty
- Eε -- E0 -- Gate is correct

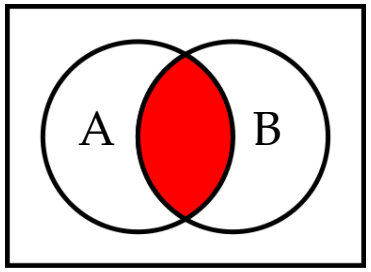
- SPz = P[Z*1, Ec] + P[Z*0, Eε]
- Joint Probs:
  - CPZ0 = P[Z0, Zc] = P[Z*0, Z*c, Ec] + P[Z*1, Z*e, Eε]
  - CPZ1 = P[Z0, Zε] = P[Z*0, Z*e, Ec] + P[Z*1, Z*c, Eε]
  - CPZ2 = P[Z1, Zc] = P[Z*0, Z*e, Eε] + P[Z*1, Z*c, Ec]
  - CPZ3 = P[Z1, Zε] = P[Z*0, Z*c, Eε] + P[Z*1, Z*e, Ec]
- Therefore,
  - EPZ = P[Z0, Zε] + P[Z1, Zε]
    = CPZ1 + CPZ3

Lists

- All PI List (List Disjunction)
  - ALL_PI(y) = ALL_PI(A) ∨ ALL_PI(B)



- Common PI List (List Conjunction)
  - COM_PI(y) = All_PI(A) ∧ ALL_PI(B)



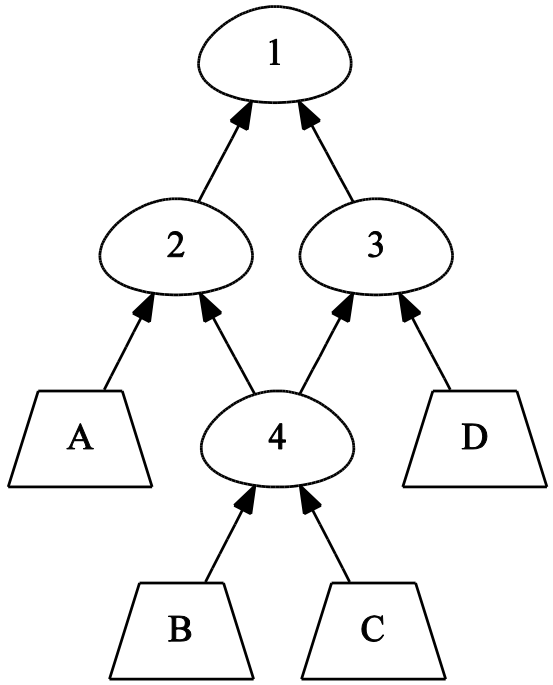
- Order PI List ()
  - COM_COM(y) = COM_PI(A) ∧ COM_PI(B)
  - OTHER = COM_PI(AB) – COM_COM(AB)
  - ORD_PI = COM_COM(AB) + OTHER

Basic Principle

- Based on Total Probability Law

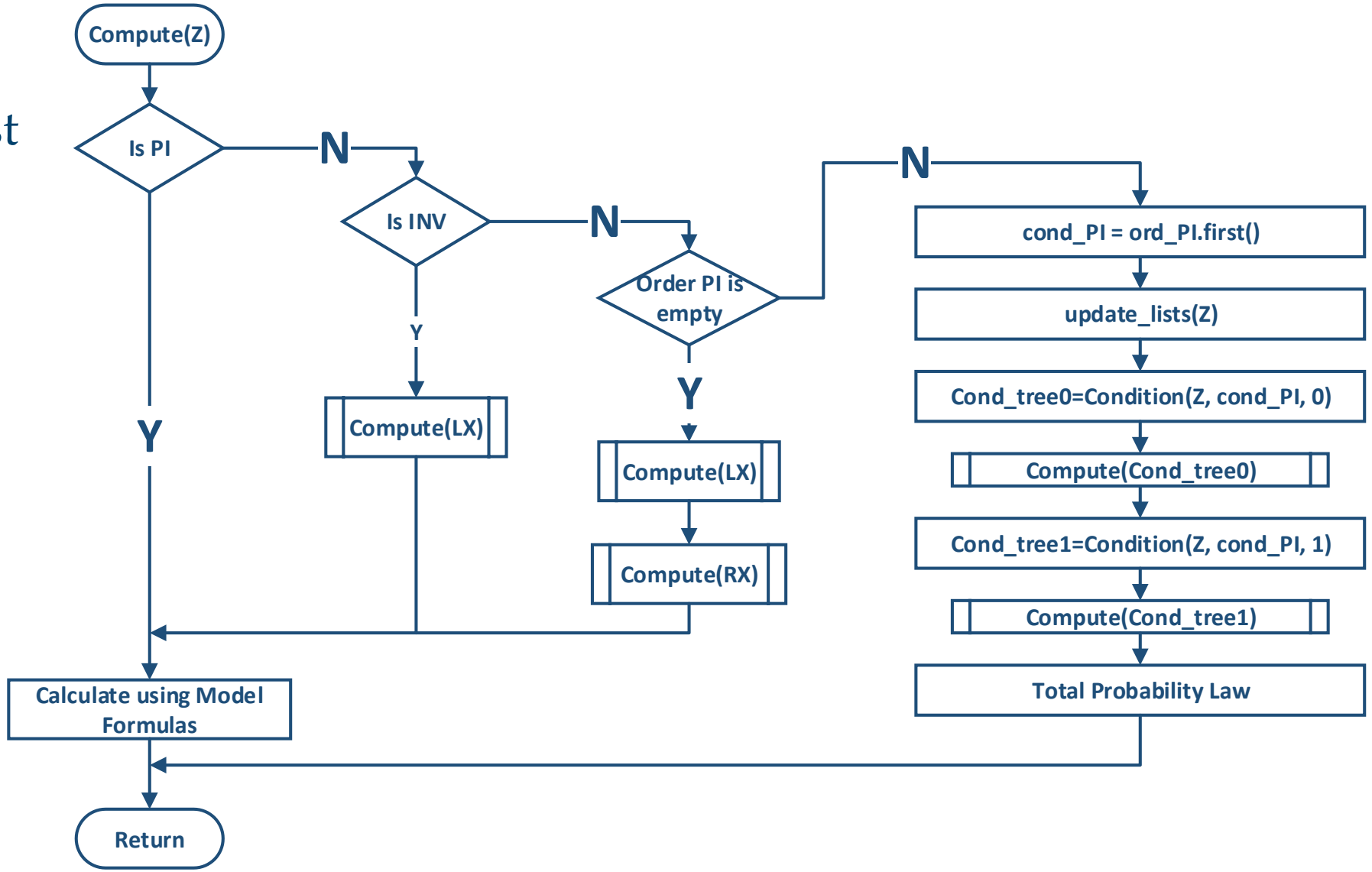
$$\boldsymbol{P(Y) = P(Y|X = 0) \times P(X = 0) + P(Y|X = 1) \times P(X = 1)}$$

Where Y is the event to be worked out, X is PI being Conditioned



Core Algorithm

- Recursive Depth-first Graph Traversal

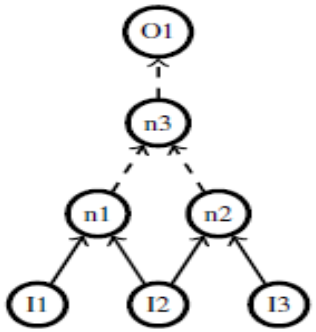


History of Logic Synthesis

- Algorithmic Based
  - Refactoring
  - Rewiring
  - Probably many other techniques as well

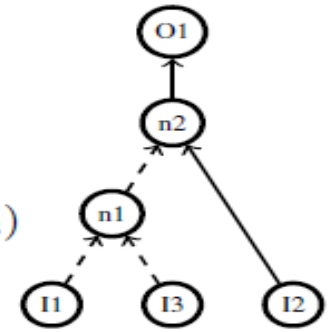
- Local Transformation Based
  - Rule Based Rewriting
  - Cut Based Rewriting

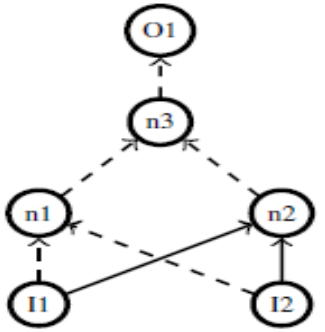
Logic Optimization – Rule Based Rewriting



Rule1

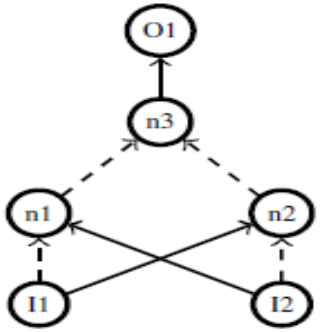
$$
\begin{aligned}
O_1 &= \overline{\overline{I_1 I_2}\ \overline{I_2 I_3}} \\
&= (I_1 I_2) + (I_2 I_3) \\
&= I_2\ (I_1 + I_3) \\
&= I_2\ \overline{\overline{I_1 I_3}}
\end{aligned}
$$

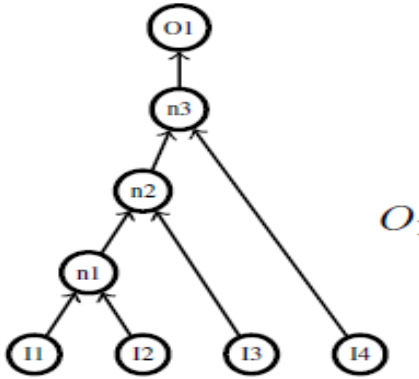




Rule2

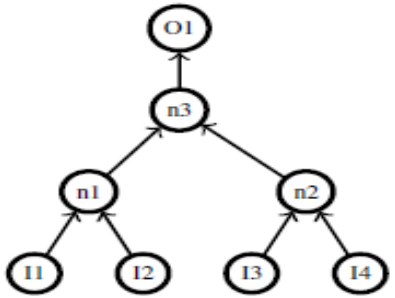
$$
\begin{aligned}
O_1 &= \overline{\overline{\overline{\overline{I_1}\ \overline{I_2}}\,\overline{(I_1 I_2)}}} \\
&= \overline{((\overline{I_1}\ \overline{I_2})(\overline{I_1 I_2}))} \\
&= \overline{((I_1 + I_2)(\overline{I_1} + \overline{I_2}))} \\
&= \overline{(\overline{I_1} I_2) + (I_1 \overline{I_2})} \\
&= \overline{(\overline{I_1} I_2)}\ \overline{(I_1 \overline{I_2})}
\end{aligned}
$$



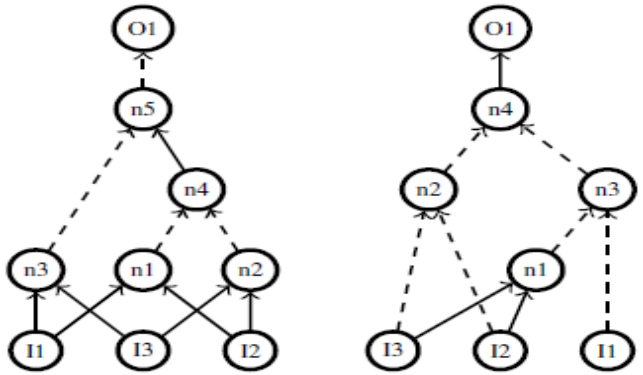


$$
\begin{aligned}
O_1 &= (((I_1 I_2) I_3) I_4) \\
&= (I_1 I_3)(I_2 I_4)
\end{aligned}
$$

Rule3

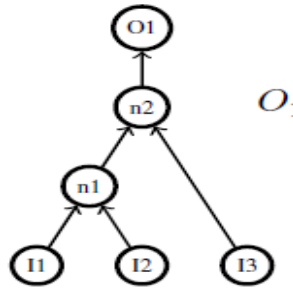


Reliability Assessment

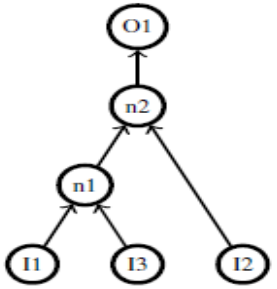


$$
\begin{aligned}
O_1 &= \overline{\overline{(I_1I_2)}\ \overline{(I_2I_3)}\ \overline{(I_3I_1)}} \\
&= I_1I_2 + I_2I_3 + I_3I_1 \\
&= I_3I_1 + I_1I_2 + I_2I_3 + I_2I_3 \\
&= I_3I_1 + I_2I_3I_3 + I_1I_2 + I_2I_2I_3 \\
&= I_3(I_1 + I_2I_3) + I_2(I_1 + I_2I_3) \\
&= (I_3 + I_2)(I_1 + I_2I_3) \\
&= (\overline{\overline{I_3}\ \overline{I_2}})(\overline{\overline{I_1}(\overline{I_2I_3})})
\end{aligned}
$$

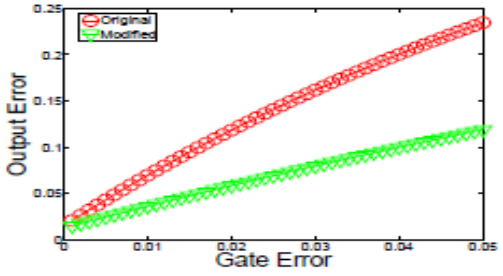
Rule4



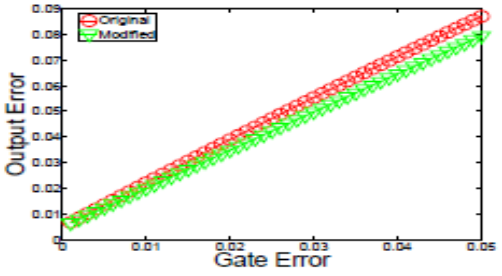
$$
\begin{aligned}
O_1 &= ((I_1I_2)I_3) \\
&= ((I_1I_3)I_2)
\end{aligned}
$$



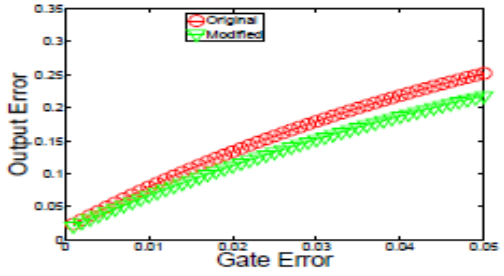
Rule5



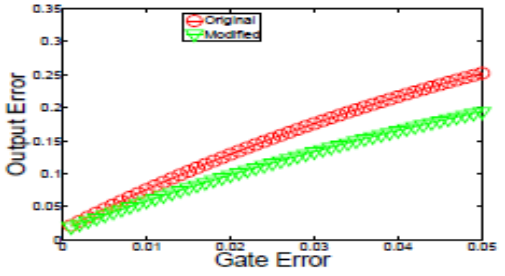
(a) Rule1



(b) Rule2



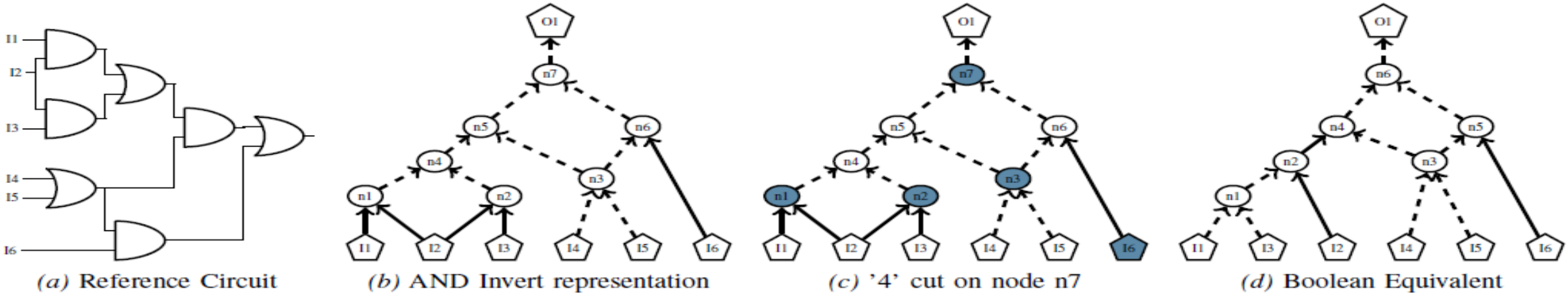
(c) Rule3



(d) Rule4

Logic Optimization – Cut Based Rewriting

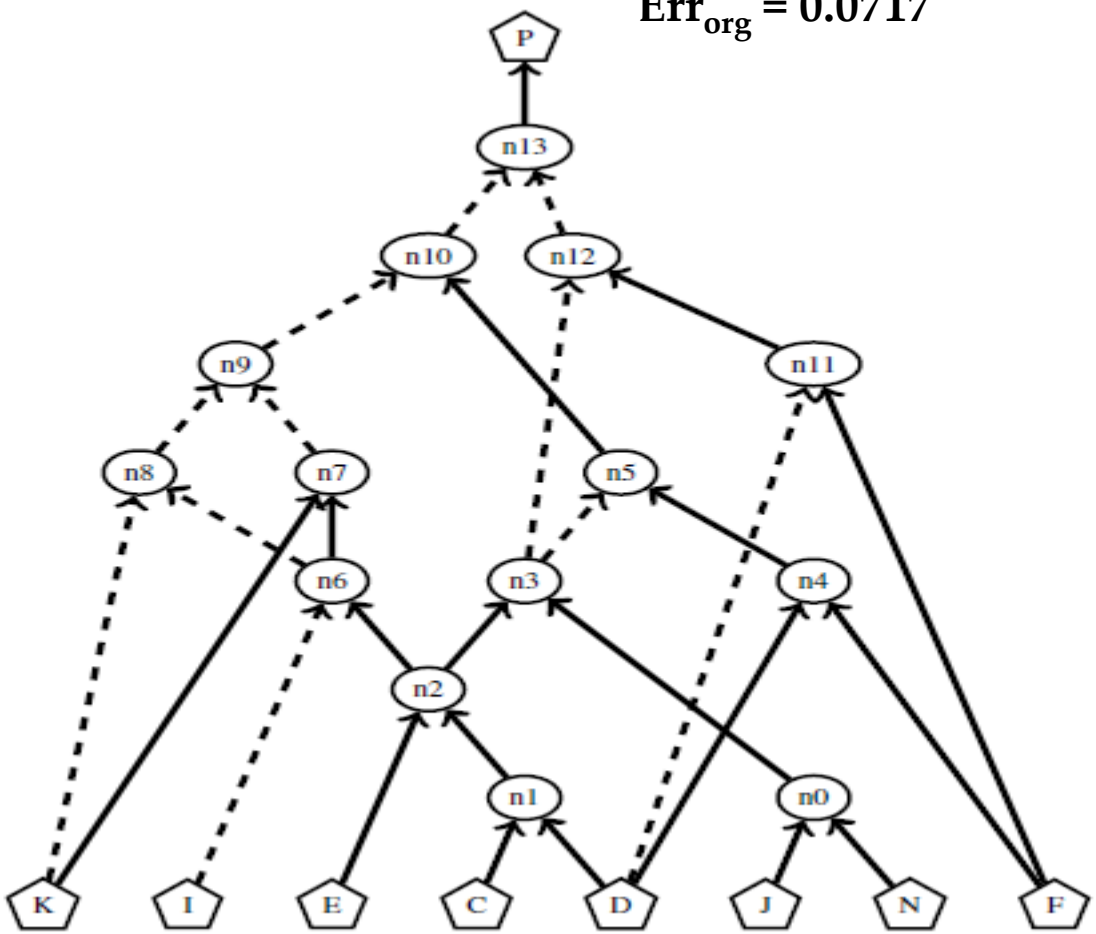
- Rewriting is a common approach to logic optimization based on local transformations.
- The best circuits are pre-computed for a subset of NPN classes of ‘n’-variable functions.
- Cut enumeration and Boolean matching are used to identify replacement candidates.



(a) Reference Circuit (b) AND Invert representation (c) ’4’ cut on node n7 (d) Boolean Equivalent

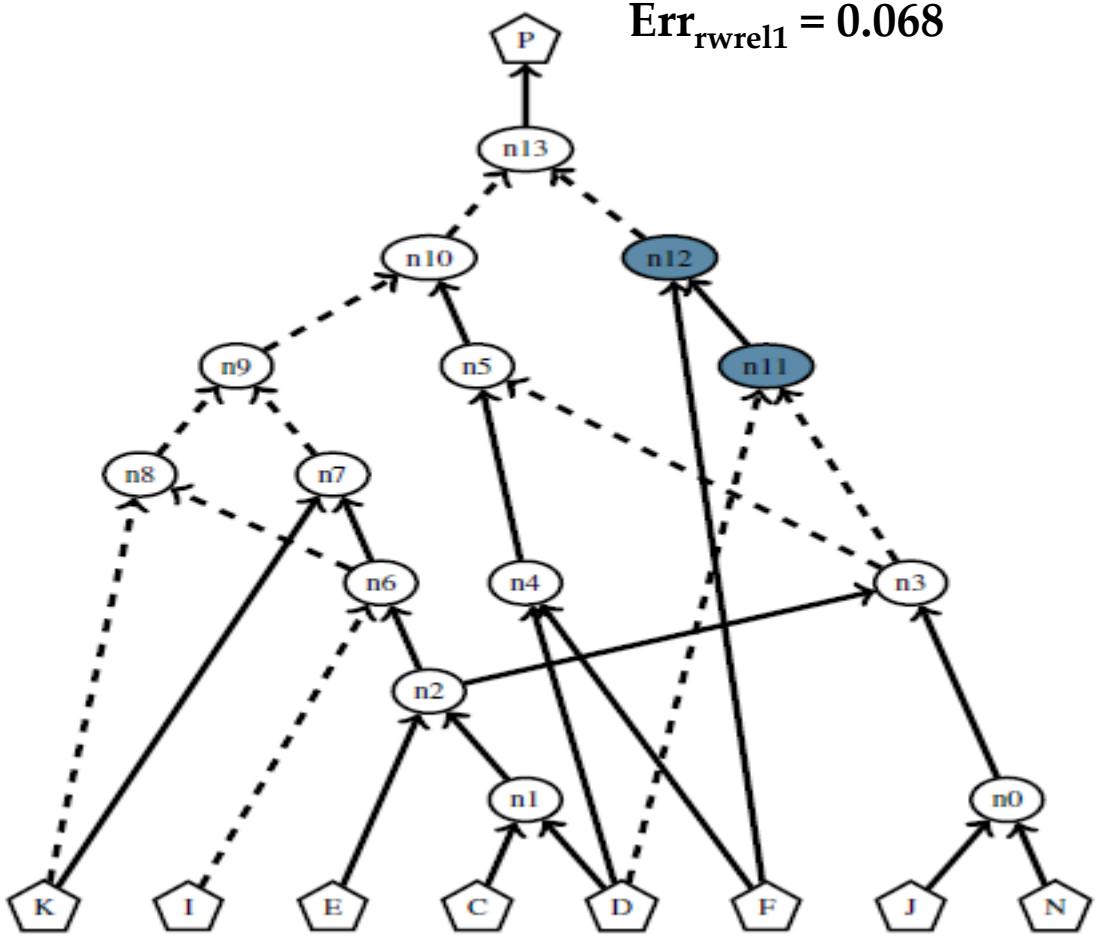
Rule1 : Move PI's closer to PO's

$Err_{org} = 0.0717$



(a) Default Configuration

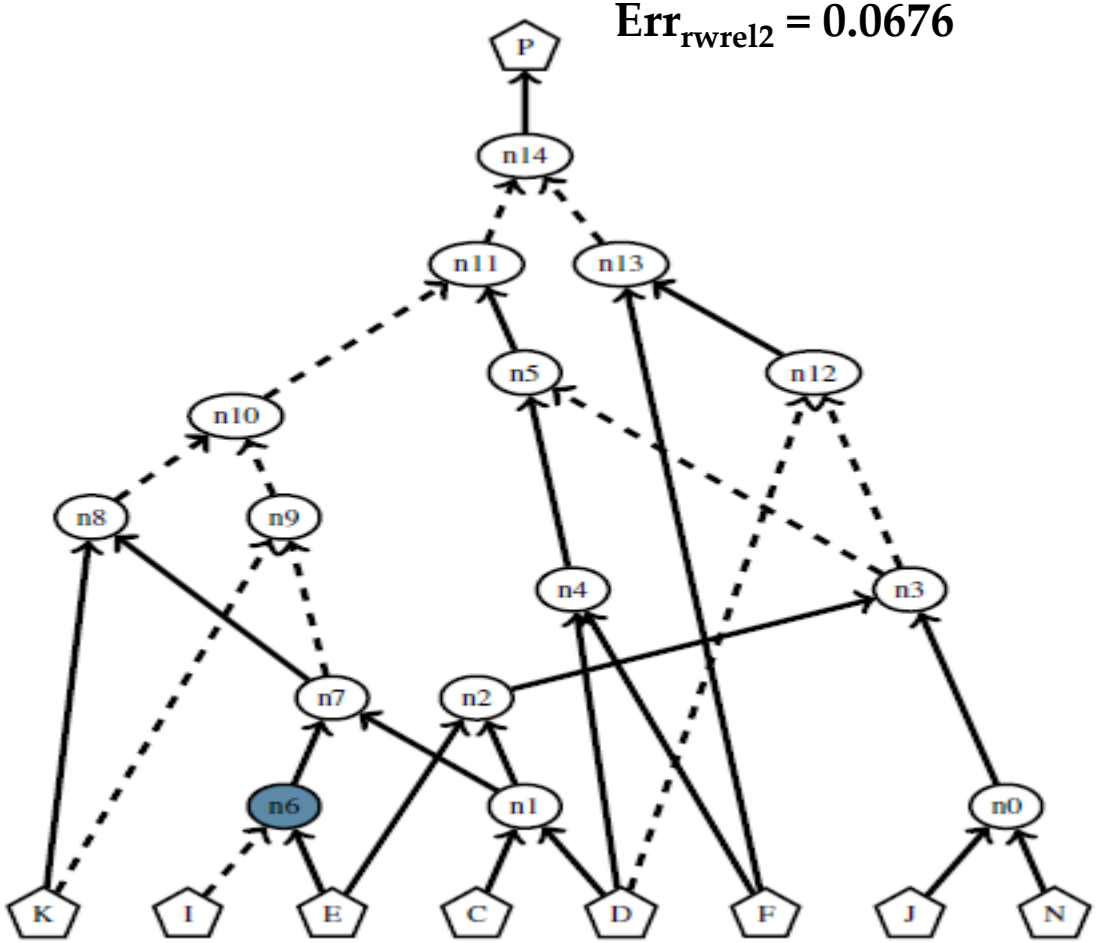
$Err_{rwrel1} = 0.068$



(b) RWREL Config1(Move PI closer to PO)

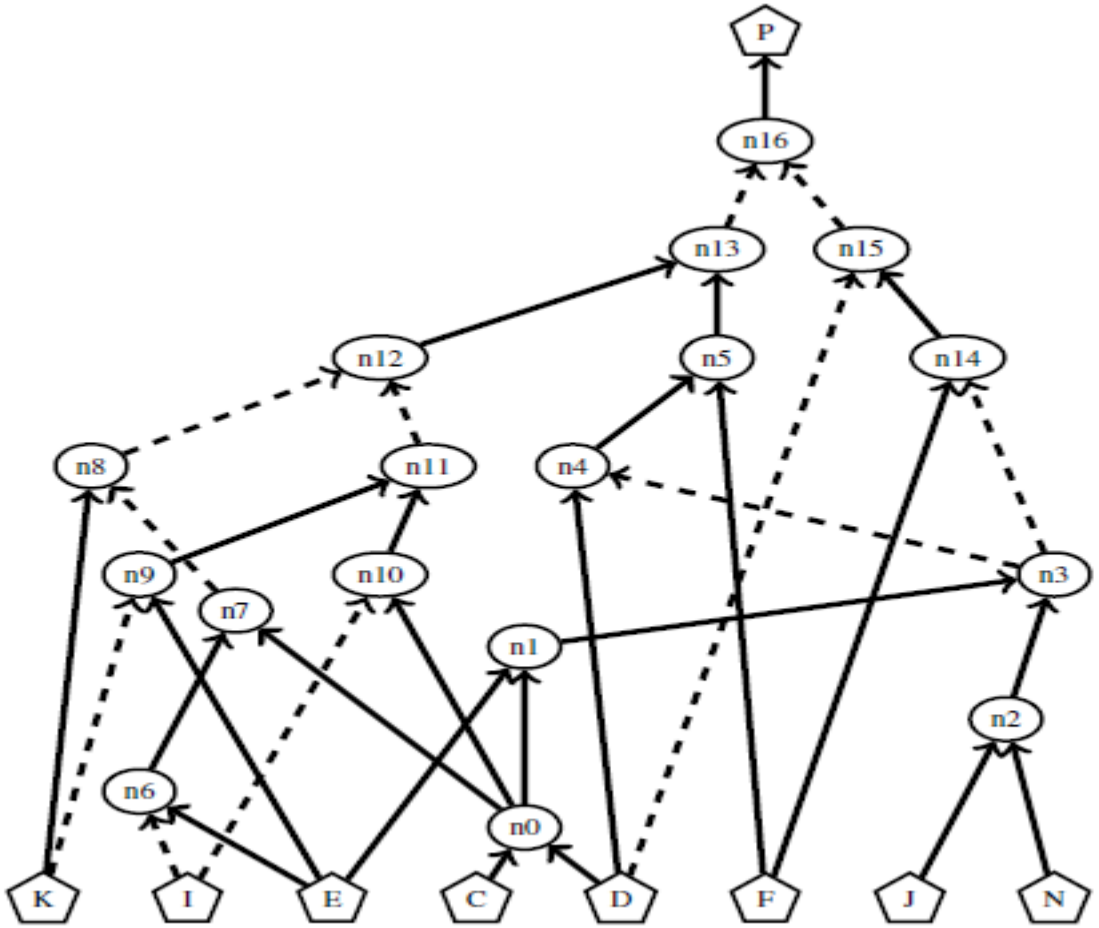
Rule2 : Adding Redundant Nodes

$Err_{rwrel2} = 0.0676$



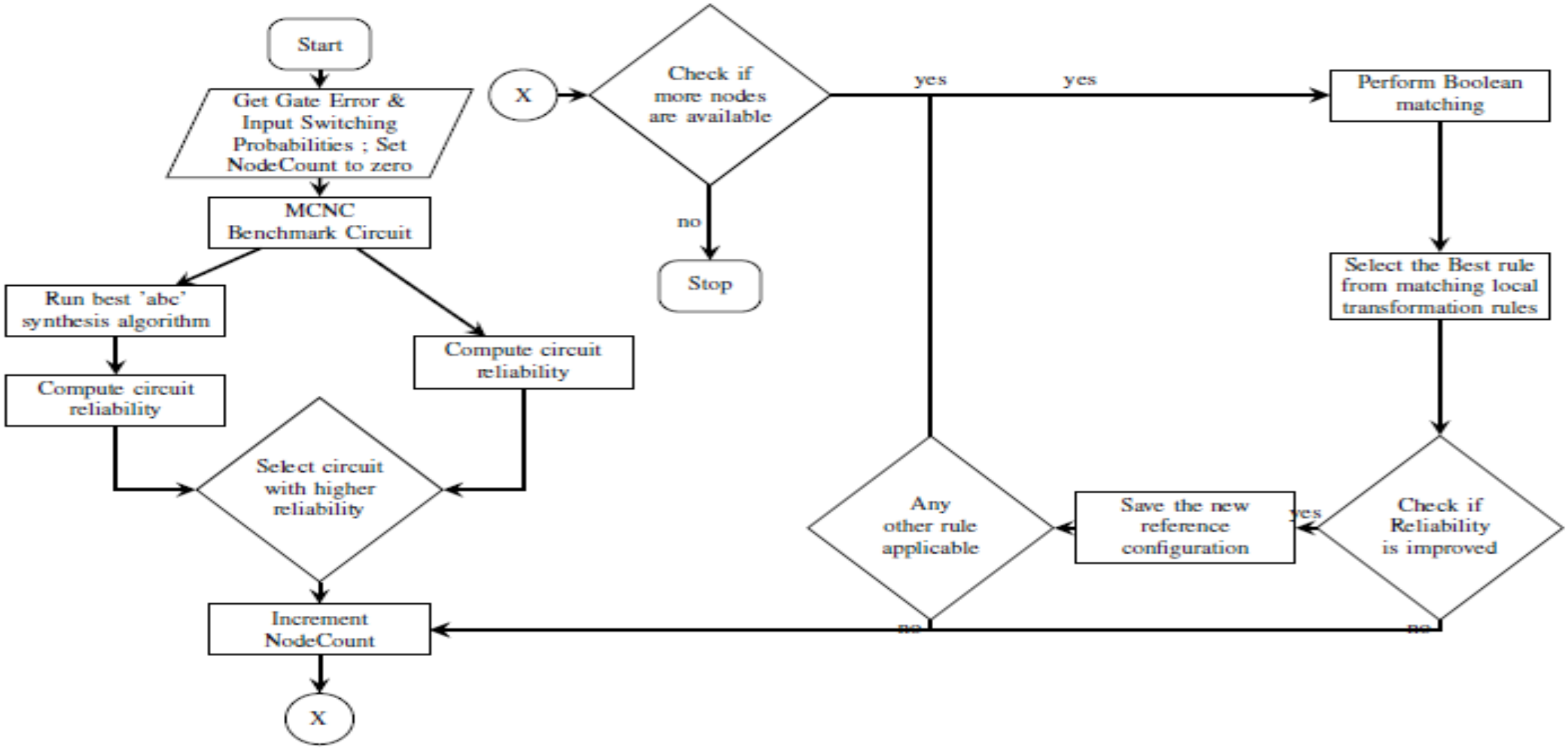
(c) RWREL Config2(Insert Redundant Node)

$Err_{opt} = 0.0633$



(d) RWREL Config3 (Optimized)

The Complete CAD Flow

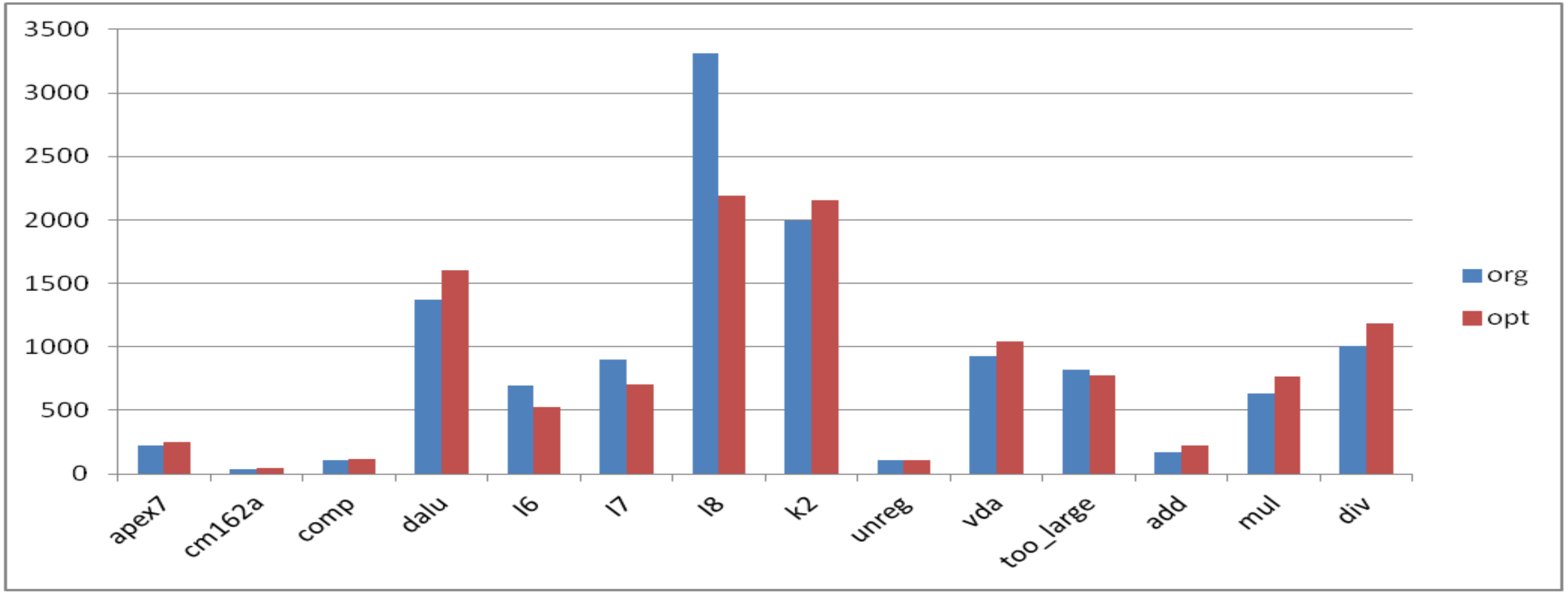


MCNC Benchmark Evaluation

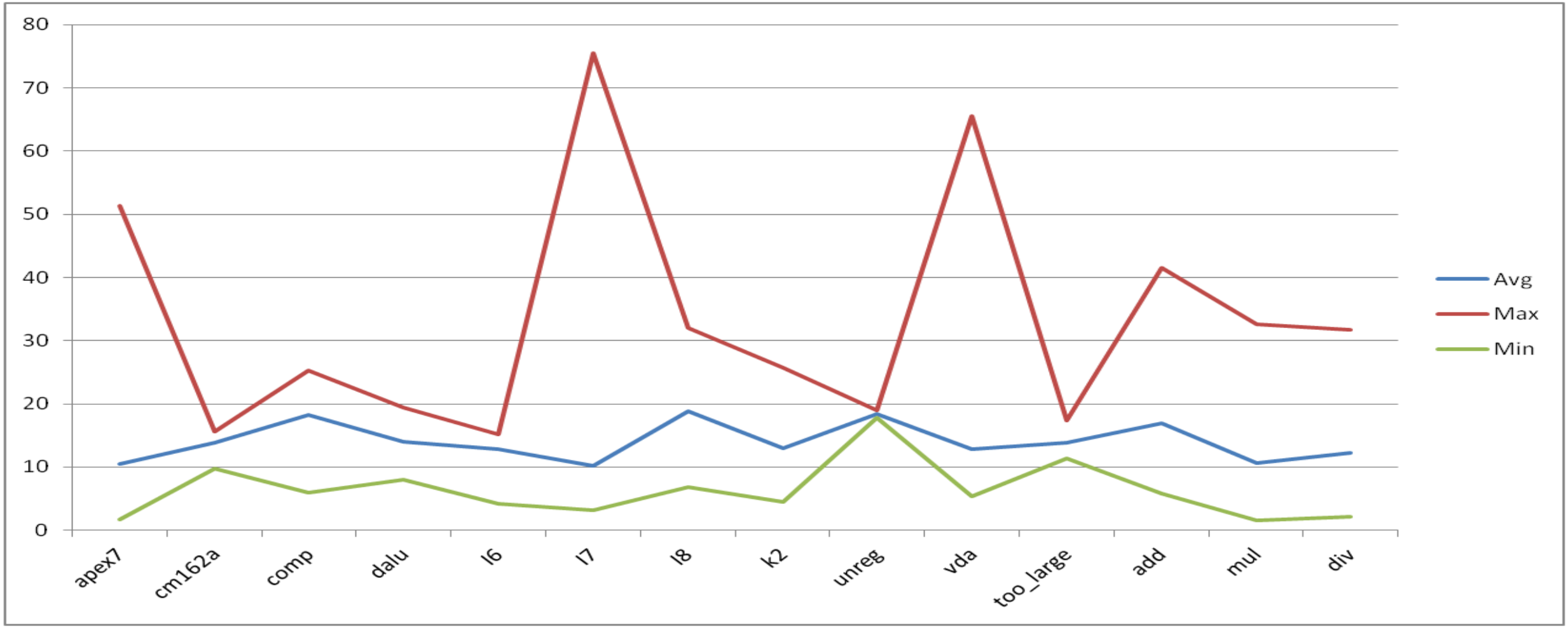
Benchmark	GateCount	Output Error Probability		Reliability Improvement $R_{Metric}\%$	Output Node Details	
		Original	Optimized		Total	$R_{Metric} \geq 0.5\%$
b9	99	0.16023	0.15036	6.15808	20	7
cm162a	33	0.22993	0.21427	6.81026	5	4
cm85a	35	0.20816	0.19640	5.65277	3	2
cu	45	0.13332	0.12700	4.73912	5	5
dalu	1371	0.32429	0.31516	2.81394	16	15
frg1	125	0.17372	0.17089	1.62925	3	1
pair	1500	0.20542	0.20429	0.54835	131	28
unreg	112	0.09779	0.09365	4.23406	16	16
vda	924	0.15885	0.15724	1.01155	39	18
x2	60	0.16726	0.15468	7.51923	7	6

Benchmark	No. of Outputs	Node Count			Circuit Depth		Averaged Absolute Error		Output Error Improvement (%)		
		Original	Optimized	Increase (%)	Original	Optimized	Original	Optimized	Average	Maximum	Minimum
apex7	36	221	252	14.03	14	14	0.05895	0.05240	10.44	51.33	1.74
cm162a	5	36	44	22.22	9	8	0.05874	0.05128	13.88	15.66	9.74
comp	3	107	119	11.21	18	19	0.09510	0.07232	18.31	25.31	5.98
dalu	16	1371	1602	16.85	35	35	0.19144	0.16449	13.98	19.45	7.98
I6	67	692	523	-24.42	5	4	0.08381	0.07291	12.83	15.19	4.23
I7	67	903	702	-22.26	6	5	0.09235	0.08303	10.21	75.5	3.16
I8	81	3310	2187	-33.93	21	20	0.18626	0.15219	18.83	32.04	6.8
k2	45	1998	2152	7.71	23	19	0.31802	0.28530	12.94	25.65	4.52
unreg	16	112	111	-0.89	5	5	0.06506	0.05382	18.43	19.03	17.78
vda	39	924	1042	12.77	16	18	0.30577	0.27491	12.81	65.6	5.31
too_large	3	824	773	-6.19	30	27	0.53492	0.46468	13.87	17.43	11.33
8051_add	19	175	222	26.86	28	29	0.14149	0.13330	16.99	41.56	5.86
8051_mul	17	630	765	21.43	48	47	0.30946	0.29475	10.58	32.65	1.58
8051_div	17	1010	1181	16.93	198	181	0.23831	0.22888	12.31	31.75	2.18

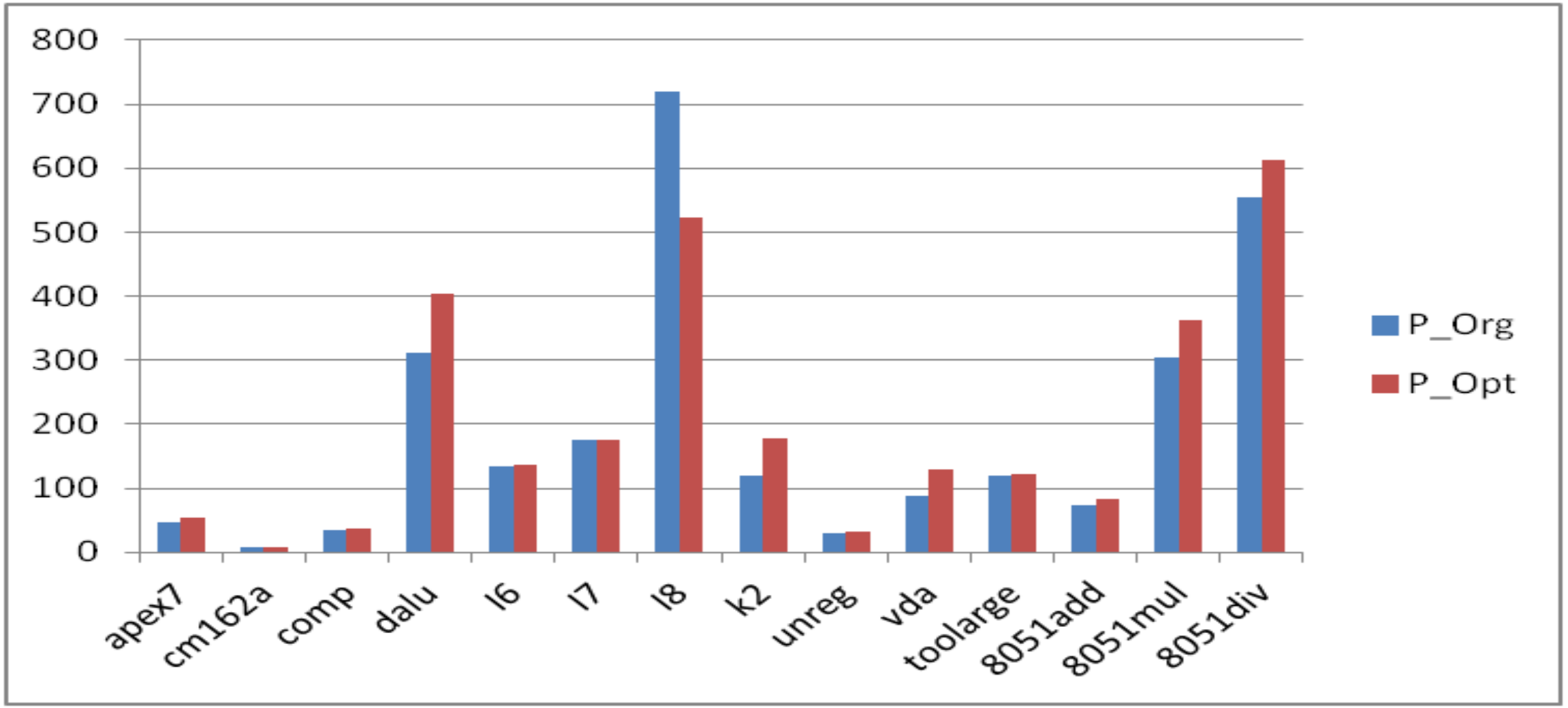
Results : Node Count



Results : Improvement in Reliability



Results : Power Consumption



Conclusion

- Gate Error model and reliability estimation for logical errors.
- Proposed a set of AIG based rewriting techniques {rule based & cut based} that improves circuit reliability.
- A EDA framework which employs both these into single framework.
- Its application on the MCNC benchmark circuits improved overall circuit reliability by
  - up to 7.5% {rule based}
  - up to 14% {cut based}

Sponsors

Collaborators