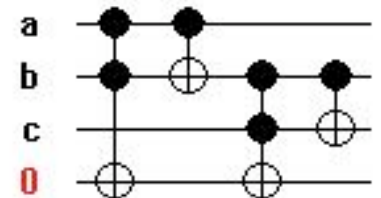


Error Detection Methods for Reversible Circuits

Nuno Alves,
Division of Engineering
Brown University
Providence, RI 02912

December 2008

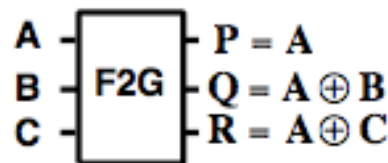
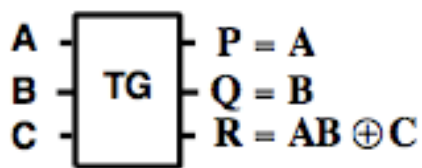


Reversible Circuits Summary

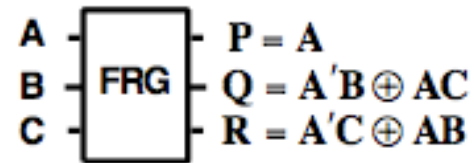
- Purpose:
 - Improve computational efficiency further than Von Neumann limit of $KT\ln 2$
- Background:
 - Landauer (1961) realized irreversible operations such as erasure of bits or merging of paths increases entropy
 - Bennet (1973) demonstrated that the reading operation does not increase entropy.
- Conclusion:
 - Building a circuit which re-uses the energy from states will save energy.

Synthesis of Reversible Circuits

- A reversible circuit is made out of reversible gates
- A reversible gate is one that has:
 - Has an equal number of inputs and outputs
 - Each input/output vector must be unique
- Synthesis of reversible circuits does not allow:
 - Feedback loops
 - Fan-out



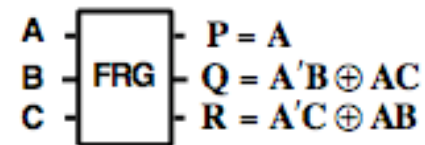
it



Example of a reversible gate

- Fredkin Gate

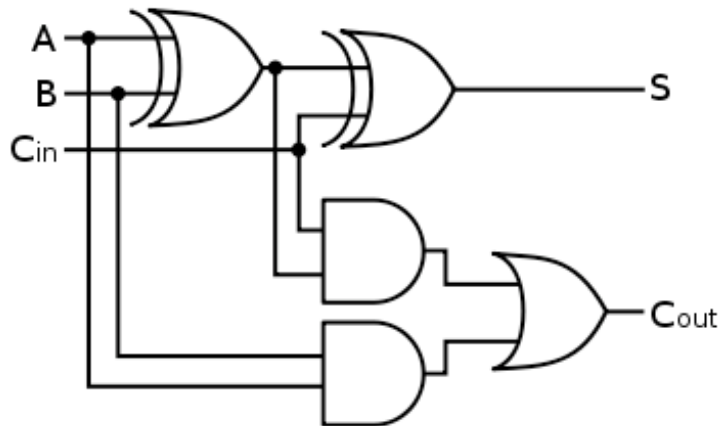
A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1



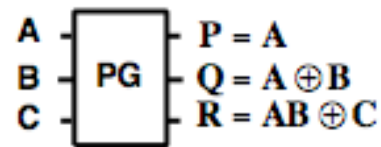
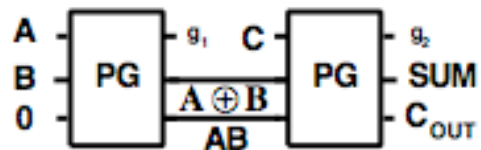
- NAND is not a reversible gate as it does NOT have a bijective mapping

Example of a reversible circuit

- Purpose: 1 Bit Adder (rd32)



- Reversible Adder Using Peres Gates
 - Two unused states: garbage outputs



Fault Tolerance In Literature

- TMR

- Trivial implementation
- A lot of extra hardware and unused states.

P. Boykin and V. Roychowdhury. Reversible fault-tolerant logic. *Dependable Systems and Networks, 2005. DSN 2005. Proceedings. International Conference on*, pages 444–453, June-1 July 2005.

- Parity

- Circuit is built out of parity preserving gates

K. N. Patel, J. P. Hayes, and I. L. Markov. Fault testing for reversible circuits. *IEEE Trans. on CAD*, 23:410–416, 2004.

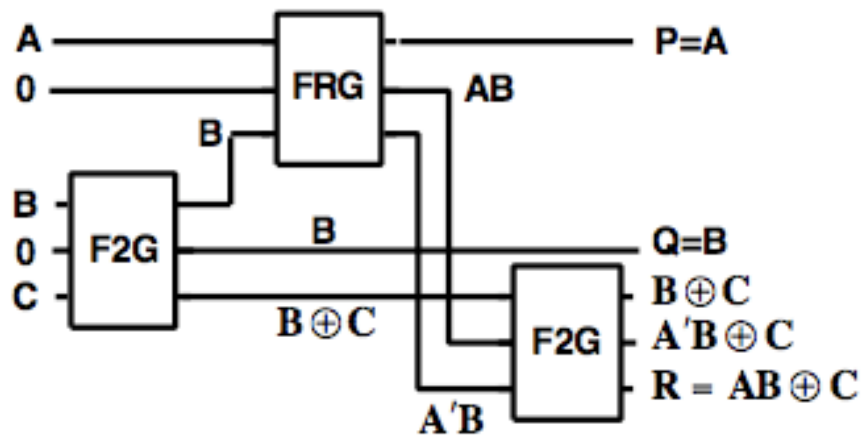
- Hamming coded gates

R. James, T. Shahana, K. Jacob, and S. Sasi. Fault tolerant error coding and detection using reversible gates. *TENCON 2007 - 2007 IEEE Region 10 Conference*, pages 1–4, 30 2007-Nov. 2 2007.

- Self Checking Reversible Pairs

D. Vasudevan, P. Lala, and J. Parkerson. Cmos realization of online testable reversible logic gates. *VLSI, 2005. Proceedings. IEEE Computer Society Annual Symposium on*, pages 309–310, May 2005.

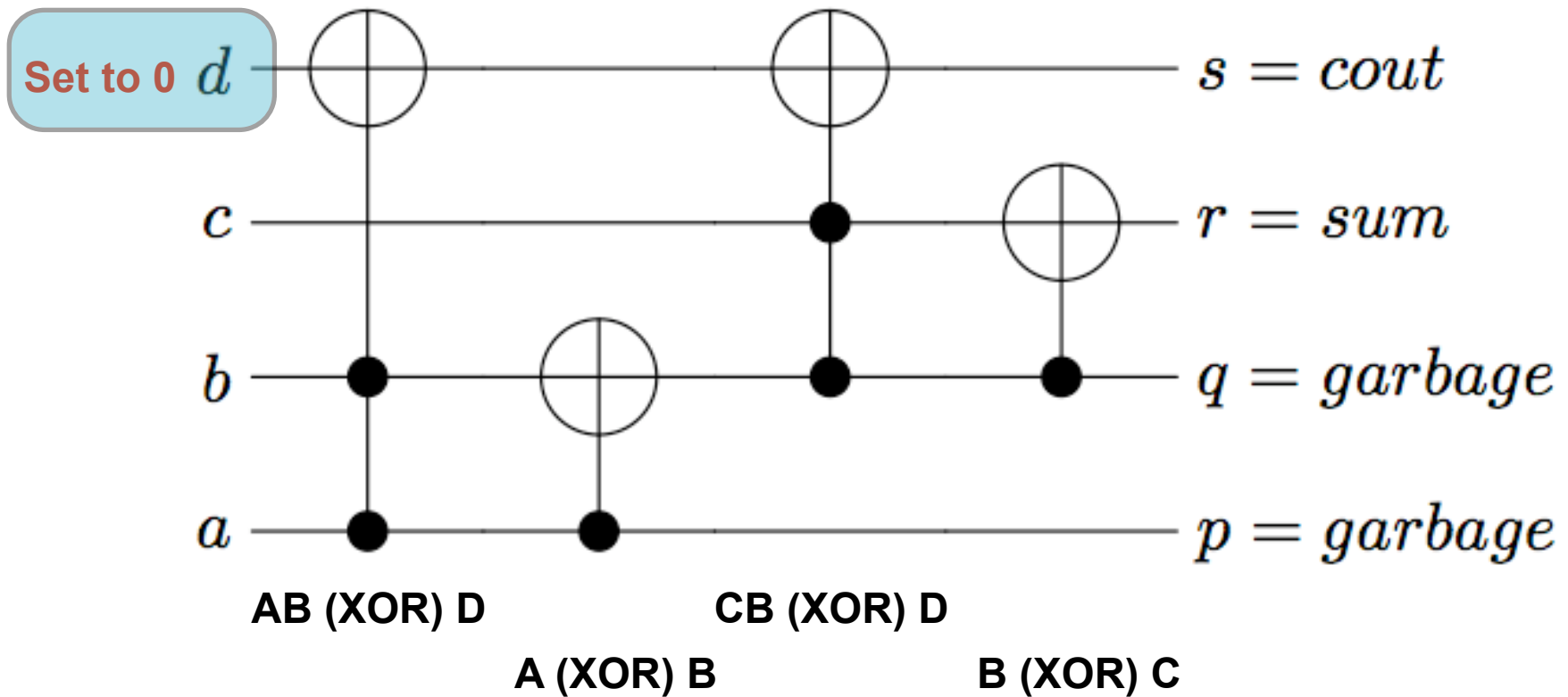
Parity Preserving 1 Bit Adder



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

Fredkin (FRG) Gate

CNOT notation for rd32

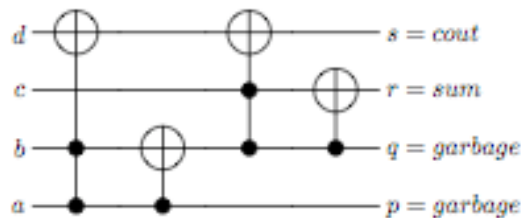
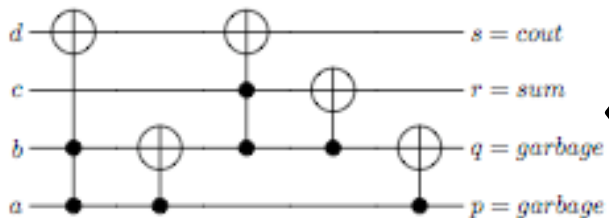


Note: When, $A_{in}=0/1$ $A_{out}=0/1$

Invariant Relationships?

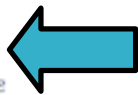
- Problem: Due to nature of reversible circuits there are VERY few of these invariant relationships.
- What to do? Force the circuit to have more of those invariant relationships!
- How? By adding extra gates...

When: $A_{in}=0/1$ $A_{out}=0/1$

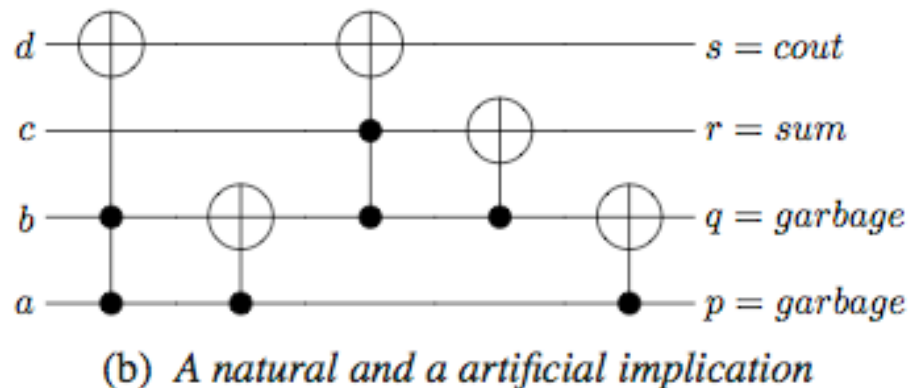
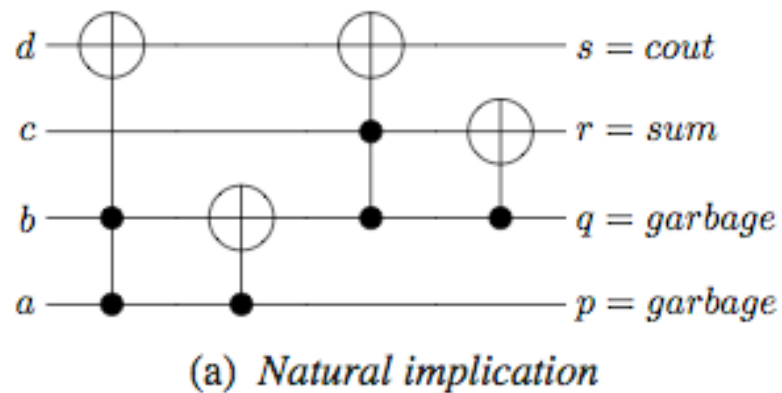


By adding a 2x2 Toffoli gate at output another invariant relationship is created:

When: $B_{in}=0/1$ $B_{out}=0/1$



Artificial Invariant Relationships



Obviously we can only add gates that will only change the logic values of garbage output!

Idea

- Get benchmark circuits
 - There are very few available, and out of those few benchmarks, most of them are impractical (e.g. 20 input gates)
- Simulate circuits
 - No simulator available, had to create mine from scratch
- Find ALL natural and artificial implications
- Do Fault Coverage on these relationships

Reversible Benchmark Circuits

Benchmark	# gates	# wires	# garbage	source
rd32	4	4	2	[15]
rd53-130	30	7	4	[26]
rd84-143	21	15	11	[26]
sym6-145	36	7	6	[26]
4gt4-v0-73	17	5	4	[26]
alu-v4-6	7	5	4	[26]
9symd2	28	12	11	[15]
ckt1-149	11553	9	0	[26]
ham7-25-49	25	7	6	[15]
hwb6-56	126	6	0	[26]

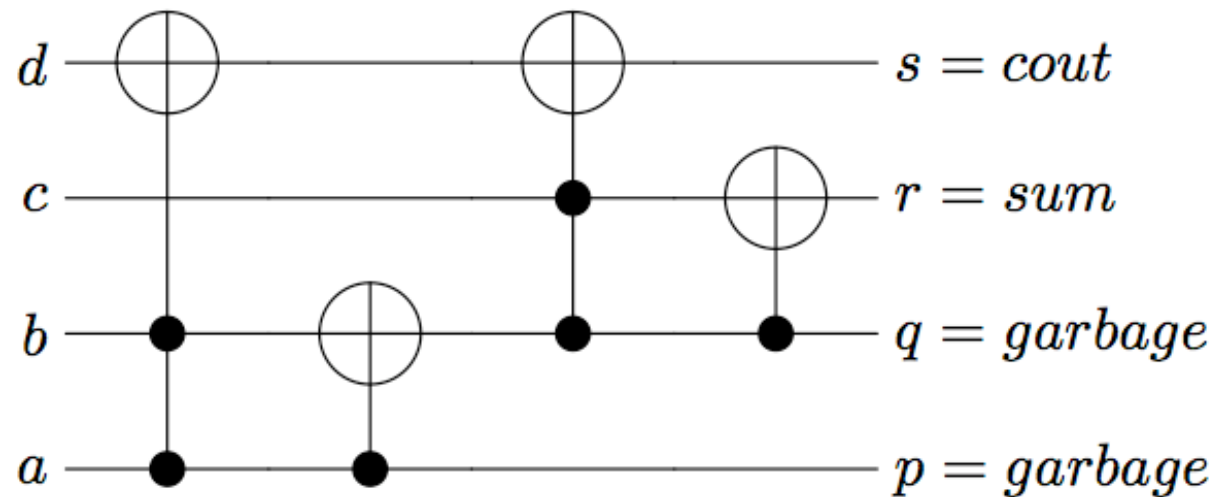
[15]

D. Maslov, G. W. Dueck, and N. Scott. Reversible Logic Synthesis Benchmarks Page, 2005. <http://webhome.cs.uvic.ca/dmaslov>.

[26]

R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler. RevLib: An online resource for reversible functions and reversible circuits. In *Int'l Symp. on Multi-Valued Logic*, pages 220–225, 2008. RevLib is available at <http://www.revlib.org>.

Fault Model in Reversible Circuits



Simple fault model: Insert a stuck-at-zero & stuck-at-one at each wire before every gate. There are 40 Faults in this circuit, 20 being stuck-at-zero!

How accurate is this model? No-one has ever built a reversible gate, and you are worried about the accuracy of this model?

K. N. Patel, J. P. Hayes, and I. L. Markov. Fault testing for reversible circuits. *IEEE Trans. on CAD*, 23:410–416, 2004.

Finding artificial implications?

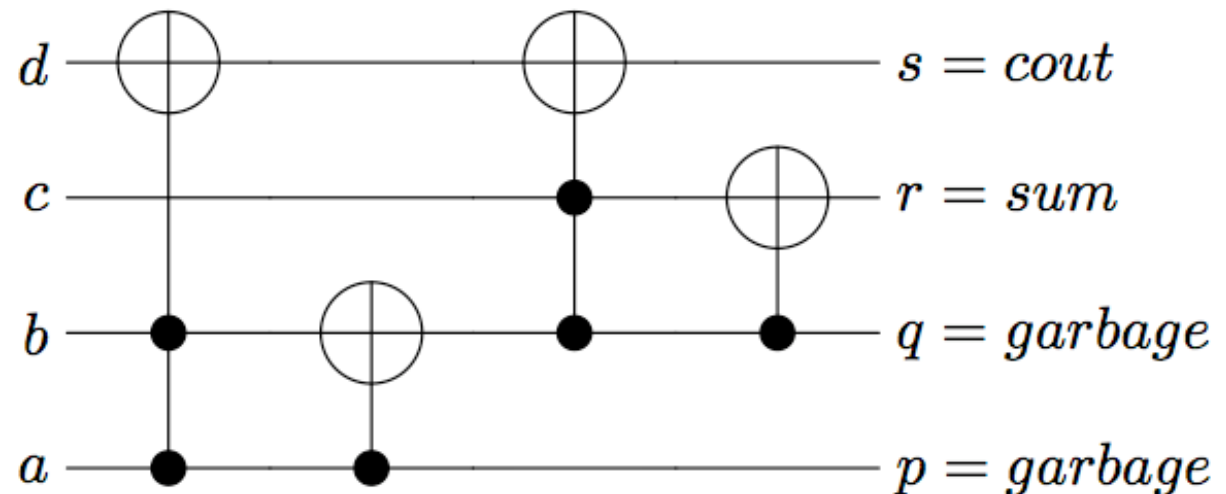
```

testgate ← any reversible gate
for each garbage wire permutation do
  - place testgate on wire permutation
  - run circuit simulation
  - check for artificial implications
end for
  
```

Figure 4: Exhaustive search algorithm to find *artificial implications*

Only 2 Wire permutations:
(q,p) or (p,q)...

... so put any reversible gate on those permutations and test for implications



Natural and Artificial Implications

benchmark	natural implications		artificial implications	
	number	avg. impact	number	avg. impact
rd32	1	12.5	1	18.75
rd53-130	3	7.14	0	0
rd84-143	1	0	0	0
★ sym6-145	5	5.12	0	0
4gt4-v0-73	0	0	0	0
alu-v4-6	1	10	0	0
★ 9symd2	2	8.2	7	22.5
ckt1-149	0	0	0	0
ham7-25-49	0	0	0	0
hwb6-56	0	0	0	0

$$implicationImpact \leftarrow \frac{errorDetected}{errorMissed + errorDetected} * 100$$

Average Impact is a metric that indicates how many faults can an implication cover.

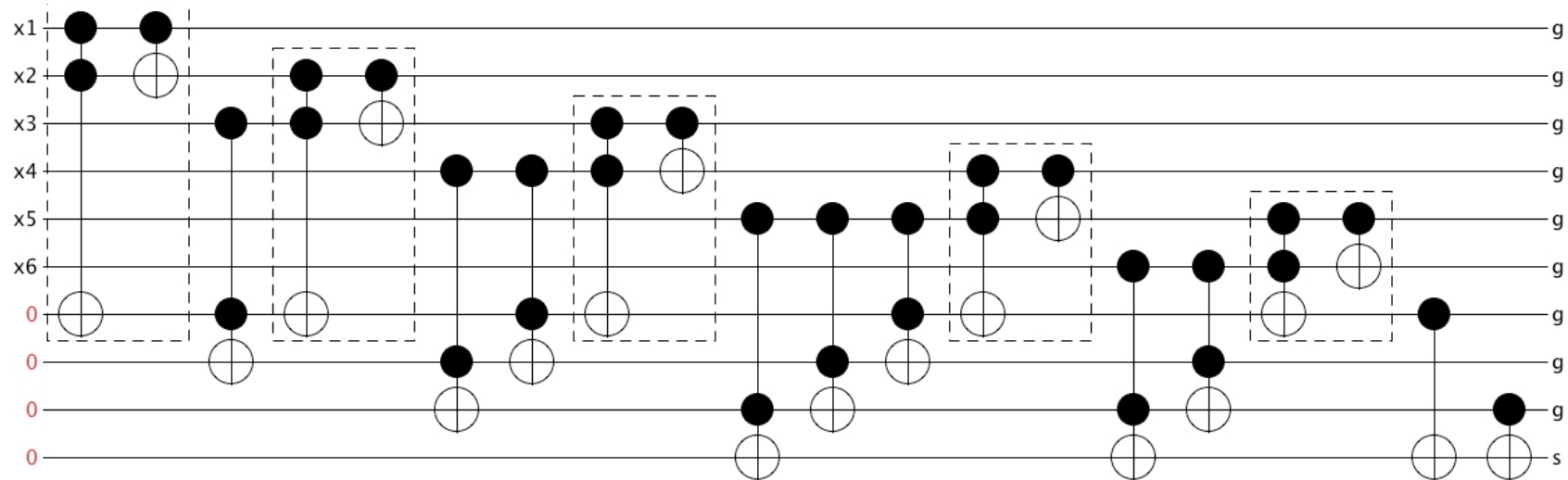
Conclusion 1: Very few natural implications...and even less artificial

Conclusion 2: Fault coverage is terrible for all except 9symd2 and sym6-145!!!

Implication Impact Algorithm

```
testImplication ← any valid implication  
errorDetected ← 0  
errorMissed ← 0  
for each input vector do  
  for each fault do  
    if TestImplication was violated  $\wedge$  error was propagated to output then  
      errorDetected ← errorDetected + +  
    end if  
    if TestImplication was NOT violated  $\wedge$  error was propagated to output then  
      errorMissed ← errorMissed + +  
    end if  
  end for  
end for  
implicationImpact ←  $\frac{\textit{errorDetected}}{\textit{errorMissed} + \textit{errorDetected}} * 100$ 
```

9sym and 6sym are bad circuits



**Here is 6sym. Of Course its a star.. the circuit is extremely simple.
As it can be seen all natural and artificial implications are trivial!**

Conclusion 3: There are no artificial implications anywhere except these trivial cases

Conclusion 4: Available benchmarks are all terribly simplistic!

Final Conclusion & Future Work

Invariance relationships DONT work on reversible circuits

Something good came out of this project:

An open source reversible simulator capable of doing fault analysis

N. Alves. *BRevSim: Brown University Reversible Simulator*, 2008. Software available at <http://www.lcms.brown.edu/nca/brevsim.zip>.

YOU can use it to (for example):

- 1) Find different reversible synthesis algorithms that maximize the number of implications in the circuit**
- 2) Design a reversible checker logic that determines if an implication was actually violated**
- 3) Find different reversible gates more suited to implications.**