THE CHINESE UNIVERSITY OF HONG KONG

Department of Computer Science and Engineering

On Structural Characteristics and Improved Scheme for Graph-Based Digital Circuit Rewiring

Fu-Shing CHIM, Tak-Kei LAM , Professor Yu-Liang WU

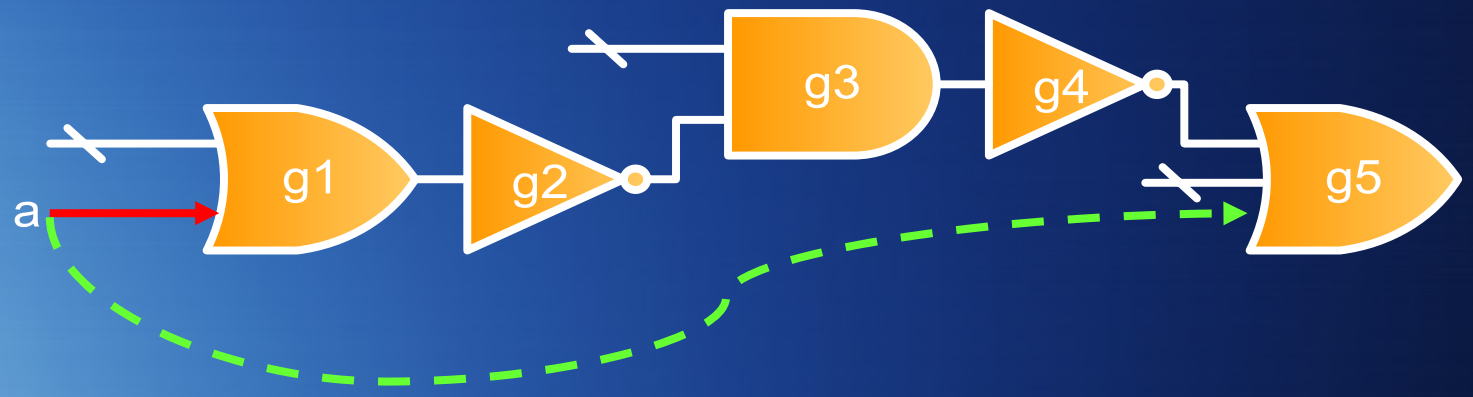
20 January, 2009

Overview

- Introduction
  - ATPG-based and graph-based rewiring
- Structural Characteristics Related to Rewiring
  - Single fanout chains for rewiring
- ATPG-based Rewiring Analysis
  - Improving redundancy identification
- Hybrid Rewiring Approach
  - Balance between runtime and power
- Q & A

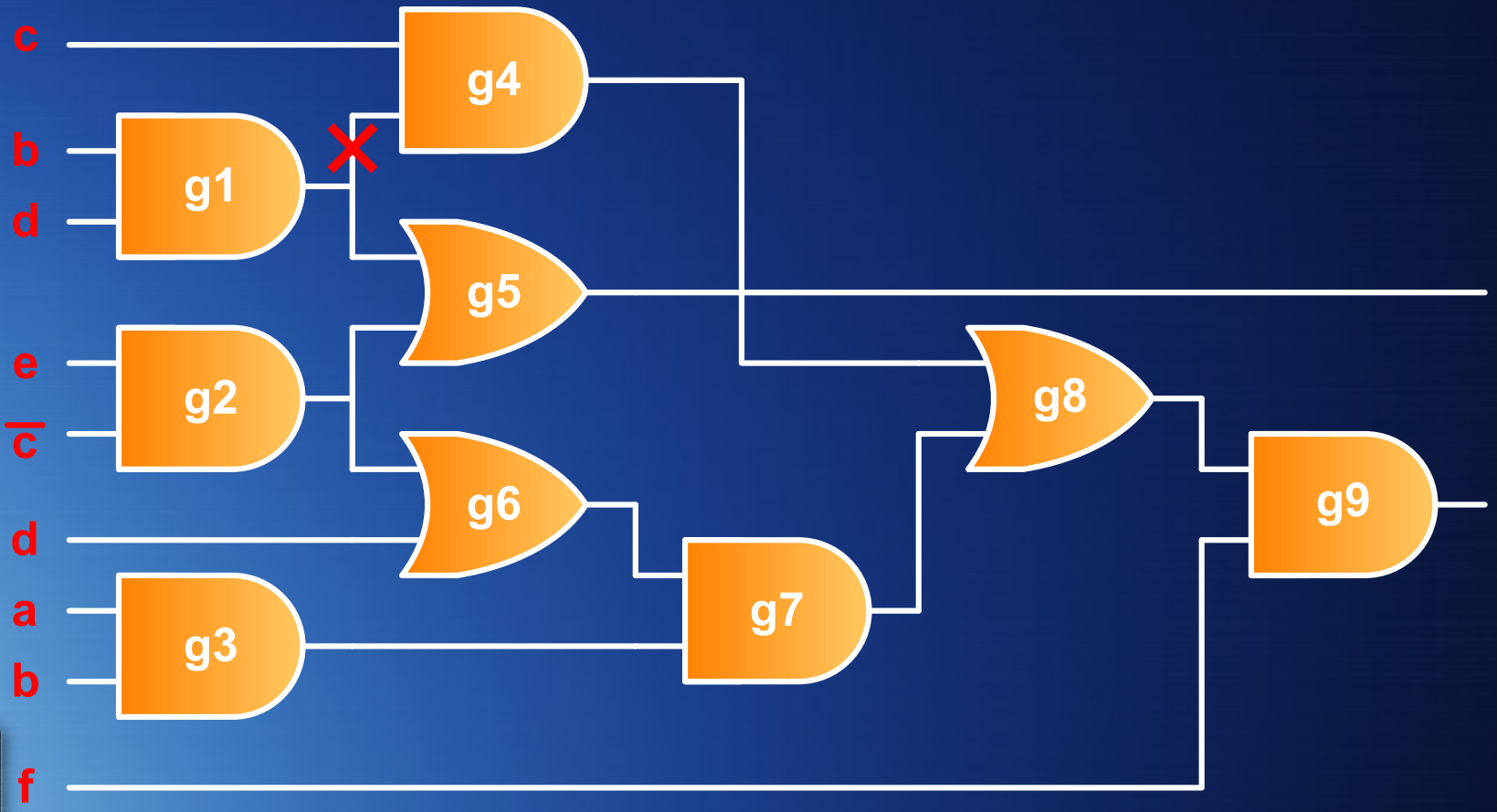
Introduction

- Rewiring – replacing *target wires (TW)* in the circuit with *alternative wires (AW)*
- Circuit functionality is maintained
- 2 categories of rewiring technique
  - ATPG-based: redundant faults identification
  - Graph-based: pattern search



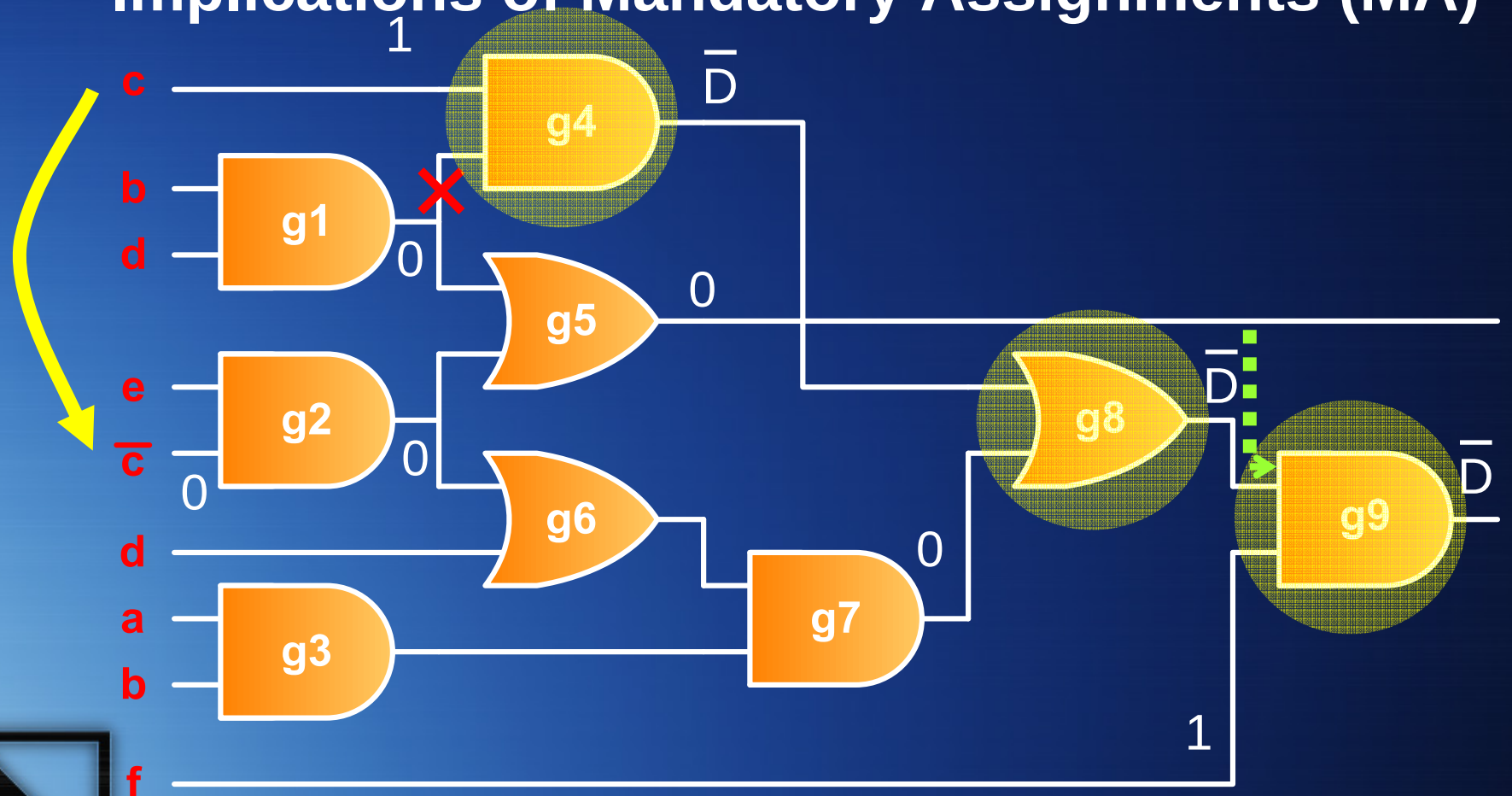
ATPG-Based: REWIRE

- REWIRE: Best ATPG-based tool
- Look for AWs by ATPG



ATPG-Based: REWIRE

- Stuck-at fault test on TW
- Implications of Mandatory Assignments (MA)

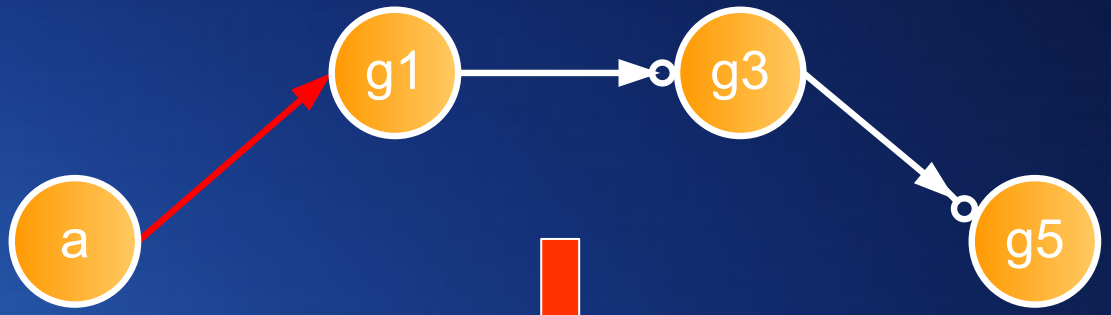


Graph-Based: GBAW

- No implication or justification
- Model input circuits as **directed acyclic graphs (DAG)**
- Patterns with Target-alternative wire pairs **predefined** within **GBAW's library**
- Look for these patterns in the DAG of input circuits
- **Pattern-matching** of graph manipulation

Graph-Based: GBAW

Look for:



(OR, n, 1

Transform :

(OR, n-1, 1)
(AND, n, 1)
g1
g3
(OR, n+1, dc)
(dc, n, dc)
(dc, n, dc)
AW
a
g5
TW

Pros and Cons

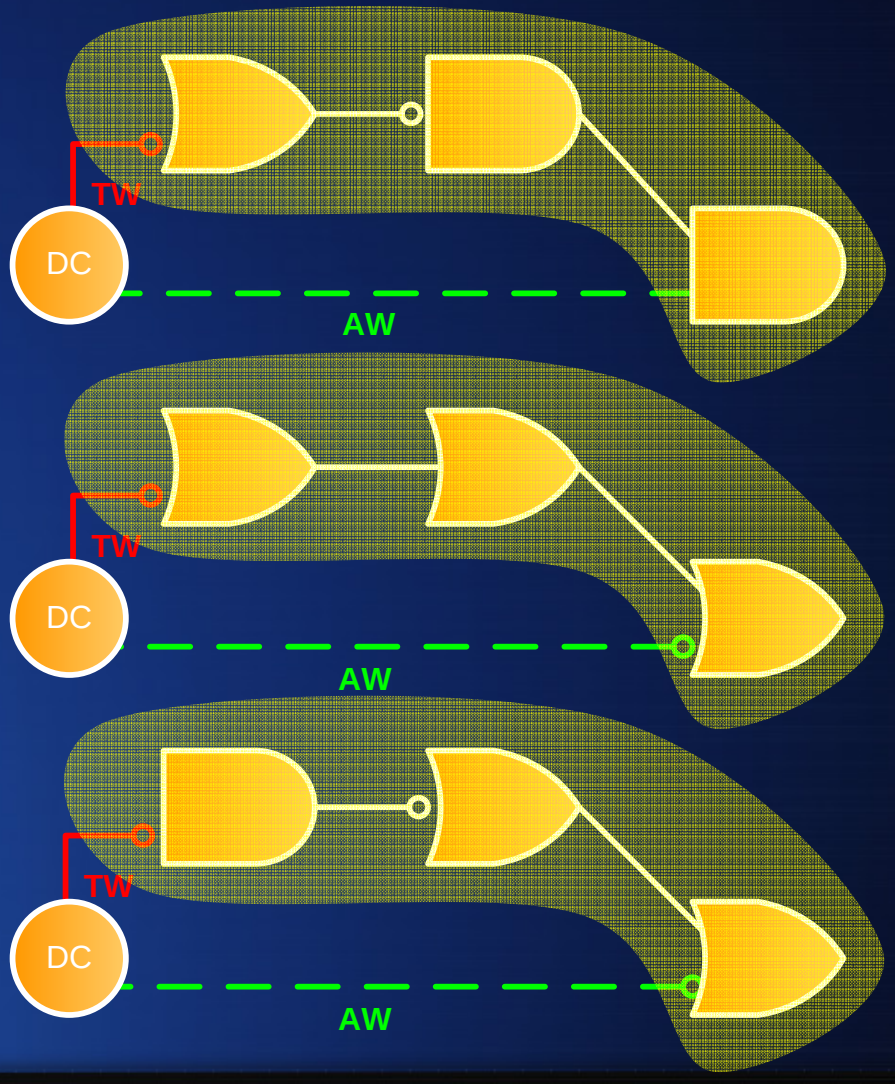
- **ATPG-based:**
  - **Higher wire coverage**
  - **Implication is slow**

- **Graph-based:**
  - **No implication, faster**
    - **Significant speedup over ATPG-based tools**
  - **Limited wire coverage**
    - **Only match patterns in its library**
    - **Limited patterns and their size**
    - **Find part of AWs located by ATPG-based tools**

Patterns Structure of GBAW

- What kind of circuit structure favor alternative wires?
  - Fingerprint structure?

Single fanout chain Reconverging AW



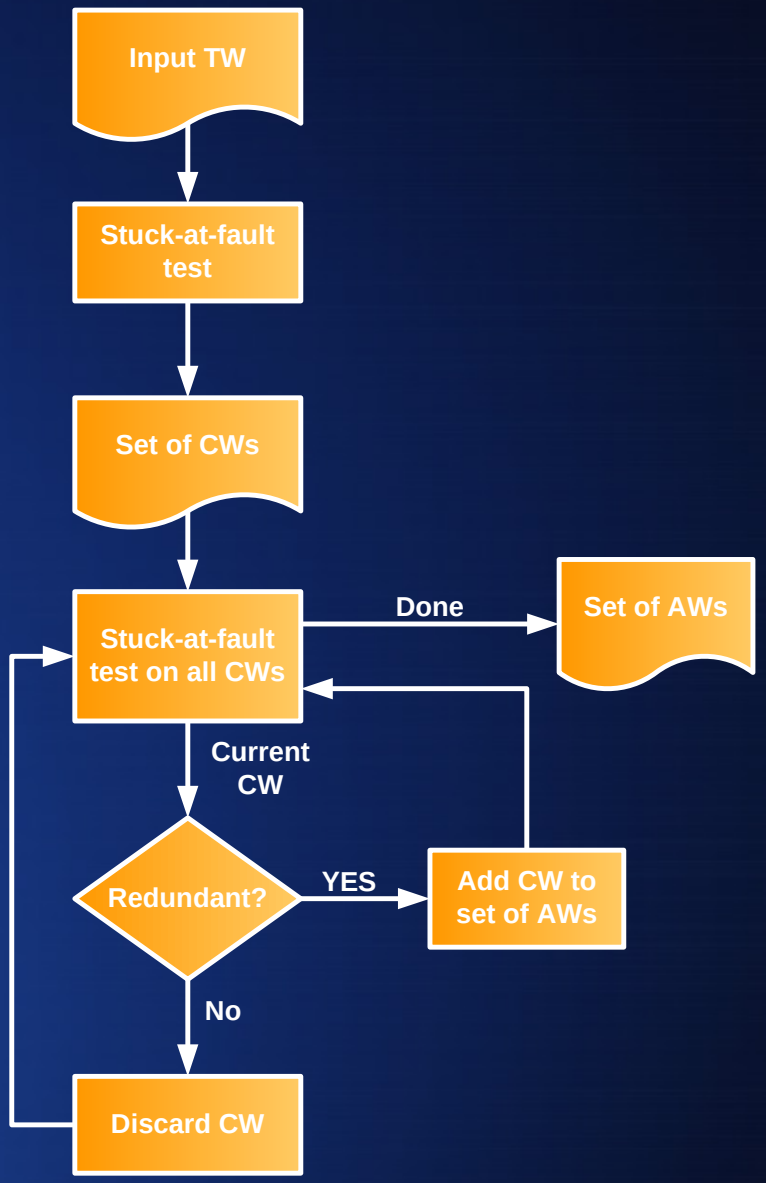
- Backbone of GBAW patterns
  - Appear in all patterns
  - Confine logic change within patterns

Common Structure from REWIRE

- REWIRE needs occurrence of single fanout chain for rewiring
  - REWIRE adds AWs to dominators of TW
  - Conflicting MA in dominators → Block fault propagation to PO → Redundant!
- Single fanout chains are dominators of TW
- Mask stuck-at faults of TW by adding AWs to the chain

ATPG-based Rewiring Analysis

- REWIRE's philosophy
  - Redundancy addition and removal (RAR)
- Candidate wires (CWs) are added
  - Guarantee to make TW redundant
  - Found through TW's stuck-at fault test
- CWs' redundancy confirmed by
  - Stuck-at faults tests on CWs



ATPG-based Rewiring Analysis

- **Multiple passes of stuck-at fault tests**
  - 1 pass for each TW
  - 1 pass for each CW for that TW
- **Redundancy Identification (RID)**
  - Determining redundancy of CWs
  - Adding CW won’t change functionality
- **RID unique to ATPG-based algorithms**
  - AW redundancy is guaranteed in graph-based
  - During pattern construction

ATPG-based Rewiring Analysis

- Complexity of RID in REWIRE
  - # of stuck-at fault test ∝ # of CWs
  - # of CWs ∝ # of TWs X (Circuit Size)
  - # of TWs ∝ (Circuit Size)
  - # of CWs ∝ (Circuit Size)^2
- # of stuck-at fault test ∝ (Circuit Size)^2
  - RID grows quadratically with circuit size
- 1 stuck-at fault test
  - A number of implications involved
- Performance Bottleneck: RID on CWs

Hybrid Rewiring Approach

- Graph-based only: limited rewiring power
- ATPG-based only: long runtime
- Hybrid Approach
  - Implications from ATPG-based

Explore custom-made CWs

Improve rewiring power

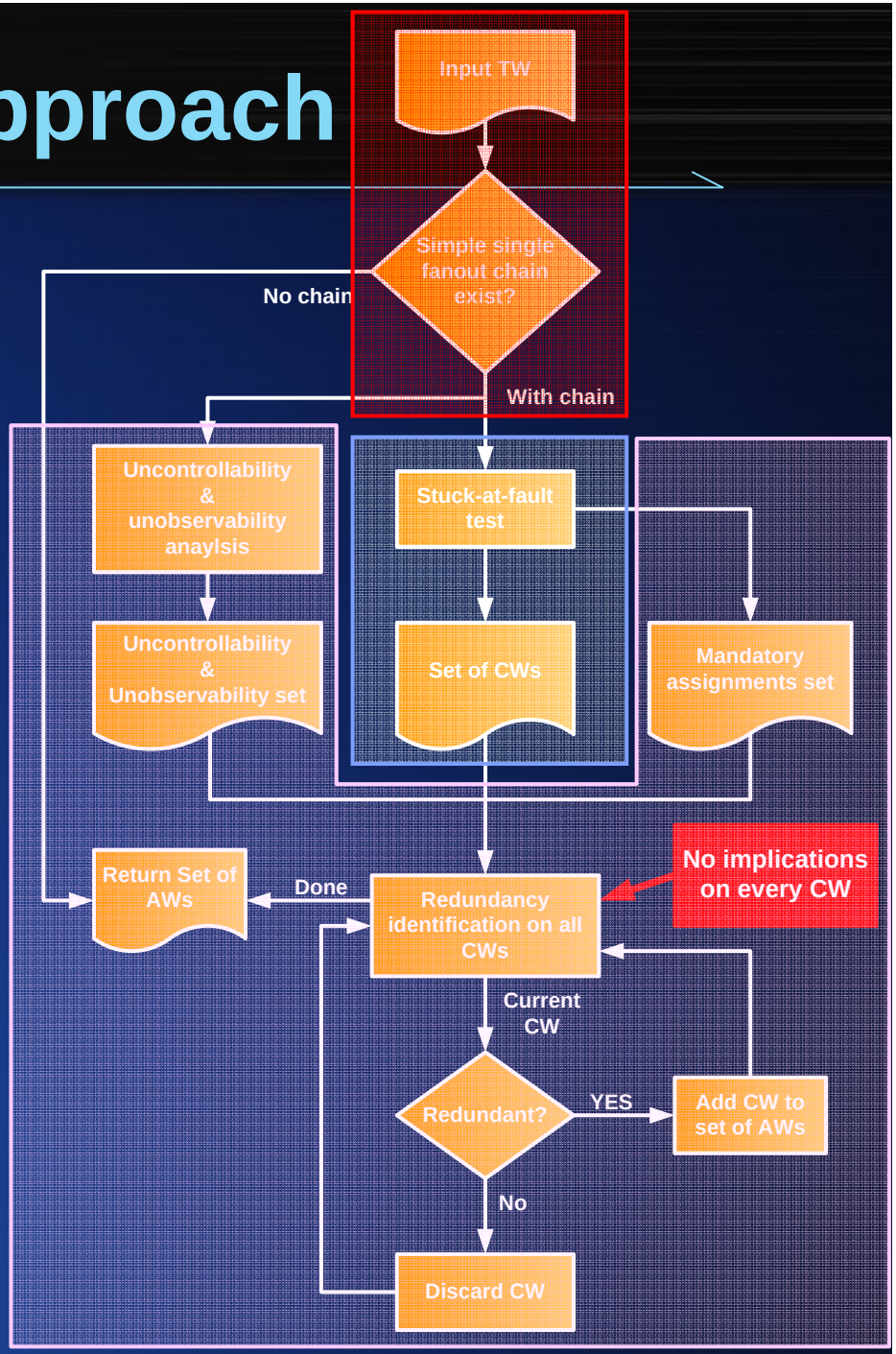
  - Structural characteristics from graph-based

Augment the RID process

Improve runtime

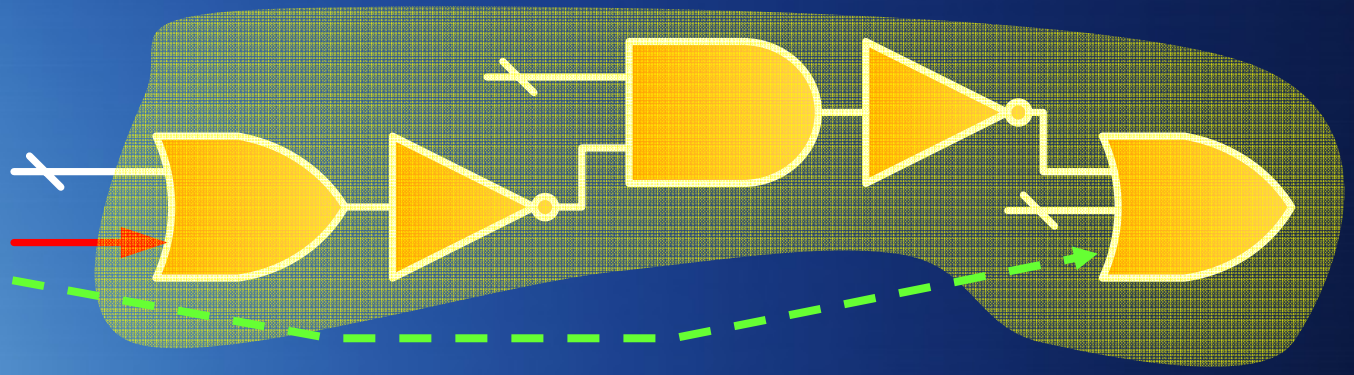
Hybrid Rewiring Approach

- Flow of hybrid rewiring
  - Chain-based Preliminary target wire filtering
  - Implication-based candidate wires generation
  - Fast redundancy identification



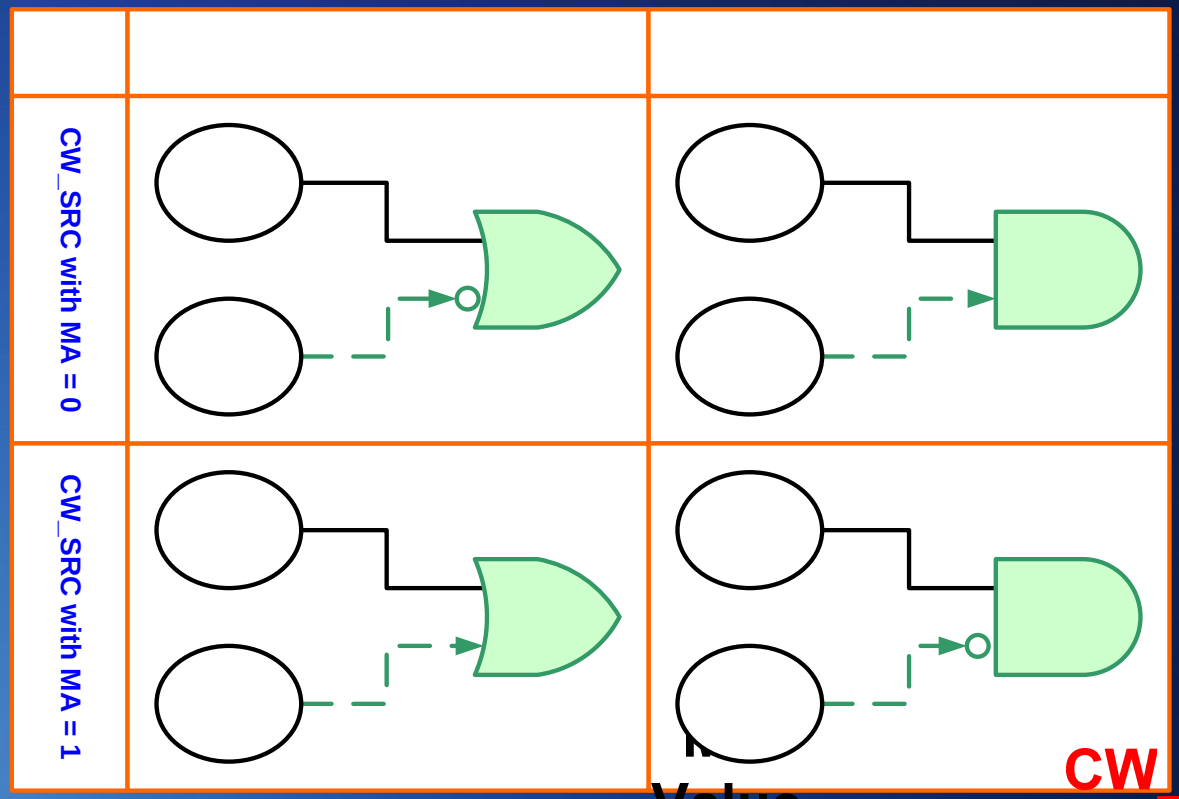
Chain-based Preliminary Target Wire Filtering

- Single fanout chains tightly related to AWs
- Effective indicator for presence of AWs
- Rewiring on demand:
  - Single fanout chains at TW_DST → Invoke rewiring engine
  - Increase probability of finding AWs
  - Reduce unnecessary AW searches



Implication-Based Candidate Wires Generation

- Adopted from REWIRE
- Most effective technique to discover CWs
- Guarantee to make TW redundant



Fast Redundancy Identification

- CW's SRC and DST MAs during TW stuck-at fault test
  - Forced MAs for stuck-at fault test of CW
- CW to be redundant
  - Blocks its fault activation/propagation

MAs from TW stuck-at fault test

CV-paths to chain from UC

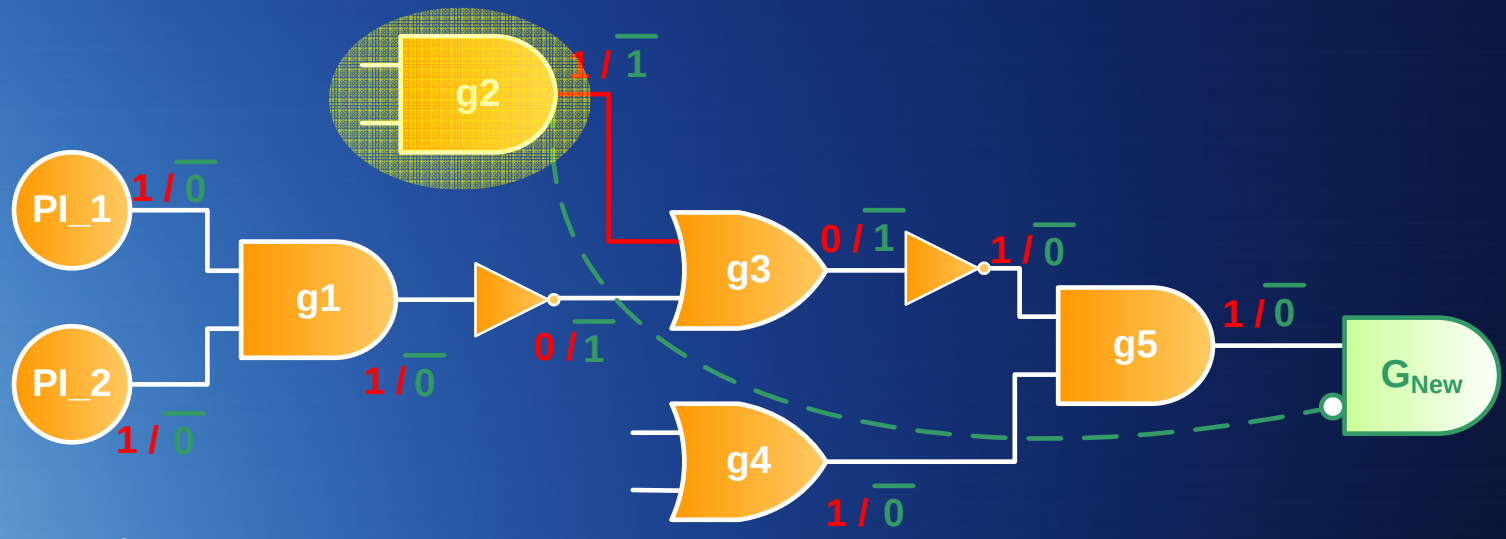
- Predict redundancy of CW
  - Without stuck-at fault tests

Fast Redundancy Identification

- **CW_SRC:**
  - Uncontrollability implied from chain end

    → **A CV-path to the chain**
  - Conflicting UC, MA from TW stuck-at fault test

    → **CV-path activated, blocking fault propagation along the chain**



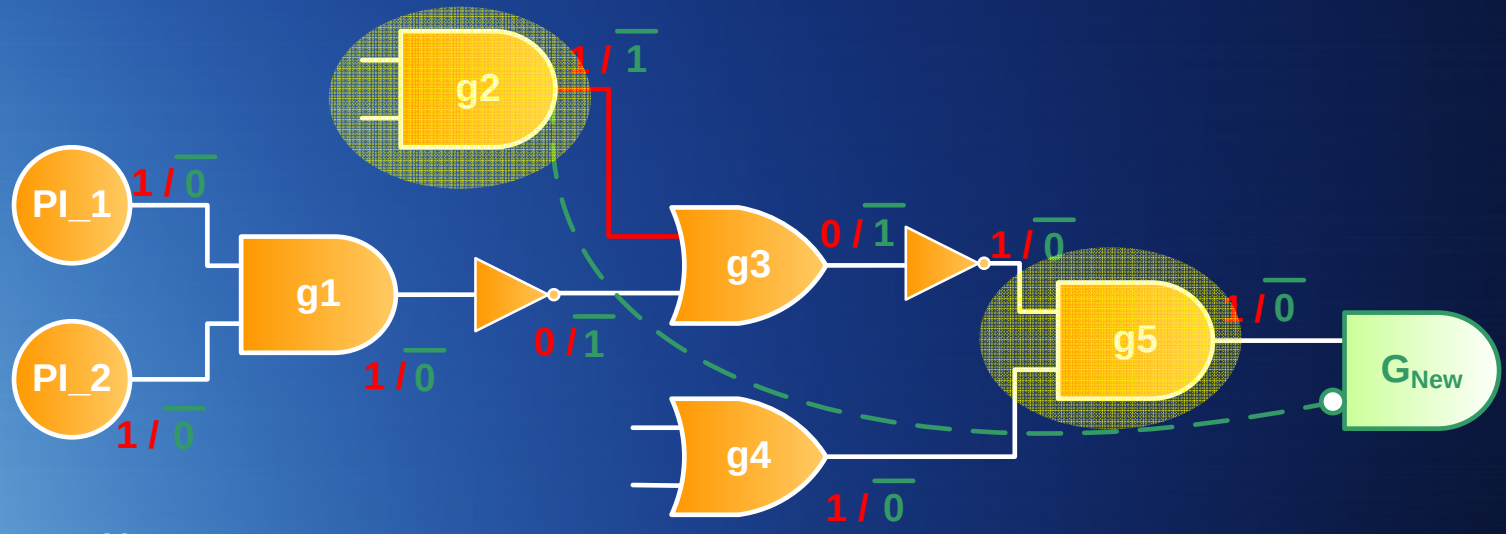
Fast Redundancy Identification

- **CW_DST:**
  - Never fanout from the fanin cone of single fanout chain before reconvergent

→ **Logic change cannot escape chain**

  - Cannot be the source node of TW

→ **MA on TW_SRC is always conflicting**



Fast Redundancy Identification

- **Linear RID**
  - Backward propagation of UC from chain end
  - 1 pass for each TW only
  - RID complexity scales linearly with circuit size
- **Quick prediction**
  - Heuristics based on circuit structures only
  - Not all redundant faults are identified

Experimental Results

Comparison of performance between HYBRID and REWIRE, gate size = 2

	REWIRE				HYBRID			
Circuit	# TW	#AW	Coverage	Time(s)	# TW	#AW	Coverage	Time(s)
5xp1	258	2052	79.07%	0.79	208	897	87.50%	0.11
alu2	668	5658	79.79%	14.97	492	2841	87.80%	0.60
alu4	1256	10044	74.20%	75.25	746	6377	89.54%	1.73
b9_n2	290	1337	82.76%	0.21	230	966	92.17%	0.03
C880	776	2710	62.37%	1.07	478	2391	87.03%	0.29
comp	174	376	62.07%	0.12	112	252	85.71%	0.06
duke2	804	8653	80.97%	19.72	600	3211	87.17%	0.59
f51m	186	680	69.89%	0.18	134	411	85.82%	0.05
misex3	798	6290	79.95%	9.82	652	2536	86.66%	1.07
pcler8	160	465	69.38%	0.20	120	313	86.67%	0.04
term1	370	2513	77.03%	0.50	288	1287	88.54%	0.09
ttt2	408	2014	78.19%	0.72	284	687	86.27%	0.07
Average	512.33	3566.00	74.64%	10.30	362.00	1847.42	87.57%	0.39

- AW Coverage: HYBRID 51.8% of REWIRE
- Speed: 26 times faster than REWIRE on average
- TW tested: REWIRE > HYBRID
- Coverage: HYBRID > REWIRE

Experimental Results

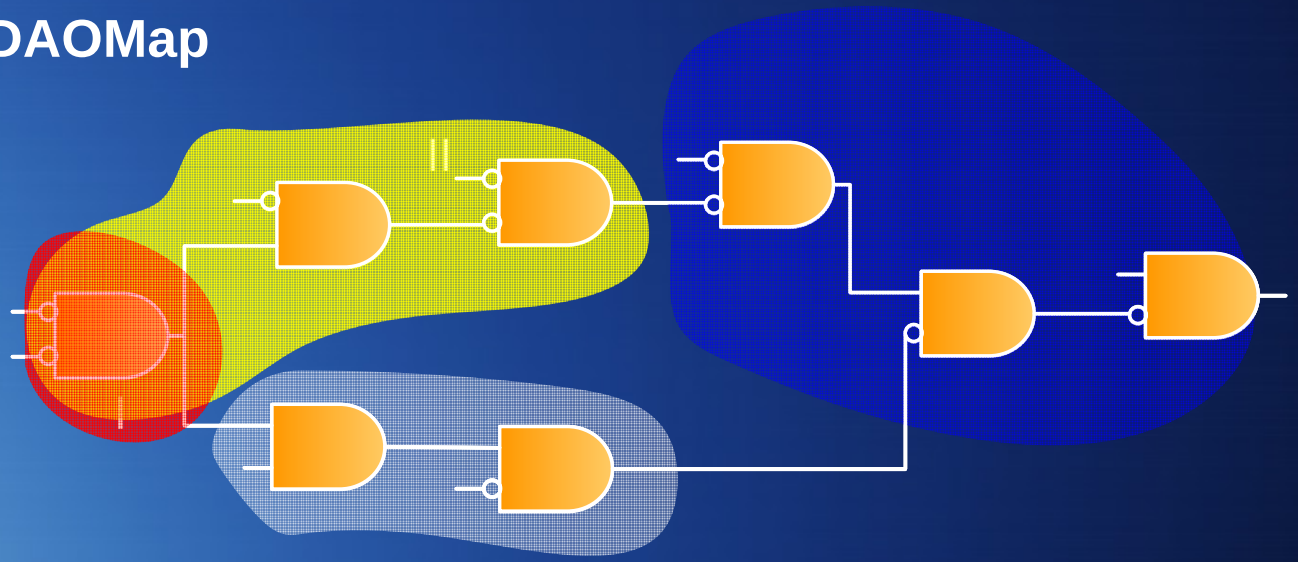
Comparison of performance between HYBRID and REWIRE, gate size = 4

	REWIRE				HYBRID		
Circuit	# TW	#AW	Coverage	Time(s)	#AW	Coverage	Time(s)
5xp1	198	546	42.64%	0.36	290	93.33%	0.02
alu2	527	1784	48.50%	6.81	1034	94.77%	0.14
alu4	1035	3943	47.85%	40.15	2622	97.17%	0.45
b9_n2	229	395	46.21%	0.13	220	89.61%	0.01
C880	638	1062	31.70%	0.57	885	97.25%	0.05
comp	174	136	35.63%	0.07	31	82.35%	0.01
duke2	629	2712	52.99%	11.48	996	94.77%	0.14
f51m	147	214	34.95%	0.11	130	100.00%	0.01
misex3	615	1481	47.49%	5.41	693	94.02%	0.26
pcler8	125	89	29.38%	0.09	0	0.00%	0.00
term1	291	585	49.73%	0.28	296	90.65%	0.03
ttt2	329	607	48.77%	0.40	88	89.19%	0.01
Average	**411.42**	**1129.50**	**42.99%**	**5.49**	**607.08**	**85.26%**	**0.09**

- Similar trends in # TW, # AW, runtime and coverage
- 54% AWs found, 50 times faster than REWIRE
- Less drop in coverage than REWIRE

Application on FPGA technology mapping

- **FPGA technology mapping**
  - **Mapping gate level circuit description to lookup tables (LUTs) on FPGA**
    - **Maximum number of inputs of a LUT = k**
    - **k-LUT**
  - **Structural based**
    - **DAOMap**



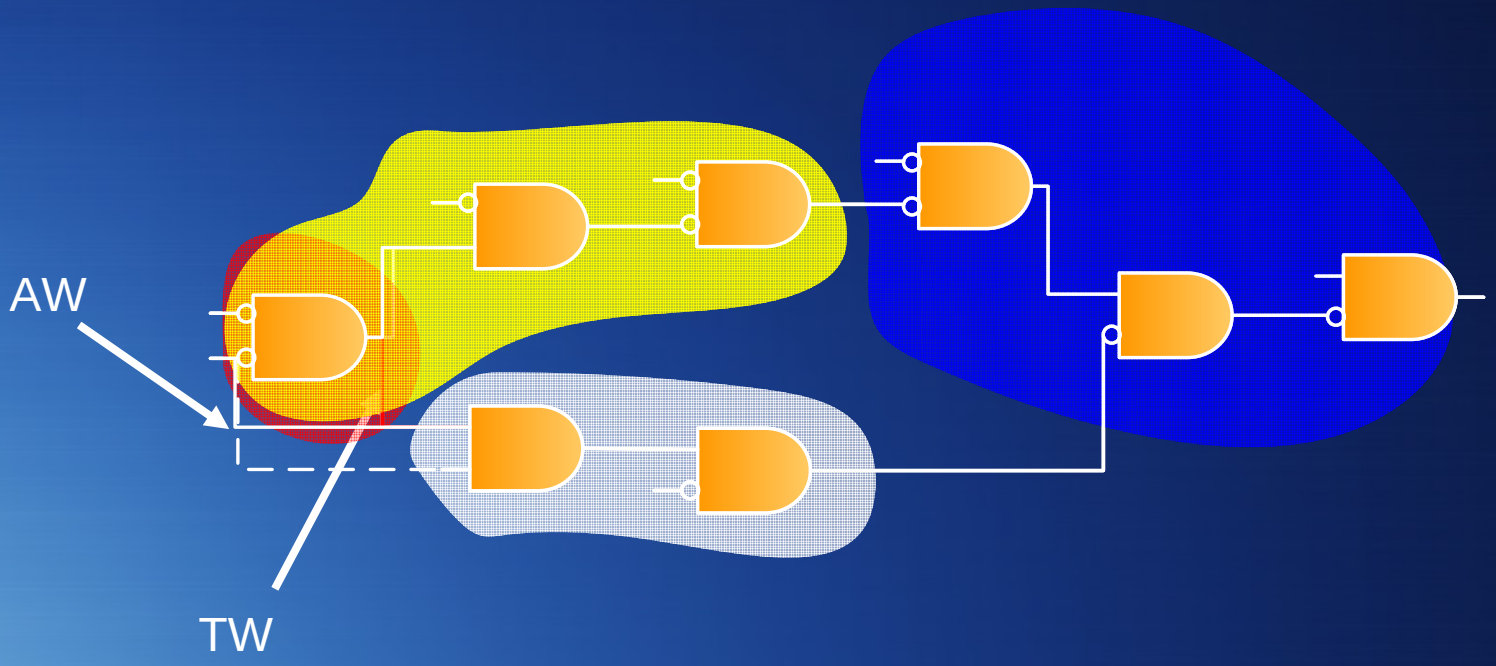
Example mapping: k=4

Application on FPGA technology mapping

- Rewiring + FPGA technology mapping
  - Applying logic transformation during mapping
- Incremental logic resynthesis (ILR) approach
  - Apply one rewiring transformation on the circuit at a time
  - Check the gain in level and area reduction in mapping
  - Adopt the change if there the gain is positive
- ILR mapping is rewiring-intensive
  - Sensitive to rewiring power and runtime

Application on FPGA technology mapping

- **ILR mapping example, k=4**



- Level reduced from 3 to 2
- LUT count reduced from 4 to 3

g3

g2

Application on FPGA technology mapping

- Previous works were based on REWIRE
- HYBRID based rewiring augmented technology mapper
  - Tradeoff between runtime and rewiring power?

Experimental Results

Comparison of performance between different technology mappers, k=4

	Initial (DAOMap)		ILR-DAOMap (With REWIRE)			ILR-DAOMap (With HYBRID)		
Circuit	Level	#LUT	Level	#LUT	Time(s)	Level	#LUT	Time(s)
5xp1	4	40.00	4	34.00	7.29	4	38.00	1.46
alu2	11	151.00	11	132.00	130.88	11	143.00	36.93
C1355	6	80.00	6	78.00	25.48	6	80.00	1.92
C1908	8	111.00	8	105.00	36.95	8	106.00	16.93
C2670	10	211.00	9	196.00	374.88	10	194.00	371.52
C432	11	85.00	11	66.00	22.98	11	72.00	18.91
C499	5	78.00	5	78.00	23.32	5	78.00	1.77
C880	9	123.00	9	114.00	35.60	9	115.00	26.87
duke2	5	146.00	5	136.00	95.05	5	140.00	17.86
f51m	4	39.00	4	36.00	8.04	4	36.00	1.55
misex3	8	214.00	8	183.00	263.72	8	198.00	40.88
pcler8	4	30.00	4	29.00	1.47	4	29.00	1.37
term1	5	66.00	5	53.00	17.12	5	63.00	4.45
ttt2	4	54.00	4	48.00	14.03	4	53.00	2.75
Average	**6.71**	**102.00**	**6.64**	**92.00**	**75.49**	**6.71**	**96.07**	**38.94**

- REWIRE and HYBRID coupled DAOMap produce close results
- Comparison of using HYBRID with respect to REWIRE
  - Only 4% less in improvement
  - More than 50% less in runtime

→ Quality of alternative wires are similar

Summary

- ATPG-based approaches study
  - Bottleneck - RID
- Hybrid approach
  - ATPG + structural characteristics
- Further improved RID
  - Uncontrollability
  - CV-paths
- Balance between rewiring power and runtime
  - Half alternative wires coverage
  - 25 ~ 50 times speedup
- Application on FPGA technology mapping

Conclusions

- Study principle of graph-based rewiring
  - Optimizations to GBAW
- Structural characteristics related to rewiring
  - Single fanout chains for rewiring
  - Chain-based rewiring
- Hybrid rewiring approach
  - ATPG-based approaches bottleneck
  - Improved RIDs
  - Balance between rewiring power and runtime
  - Performance proven high quality alternative wires

- Q & A -

- The End -