

Petr Fišer, Jan Schmidt, CTU in Prague

{fiserp, schmidt}@fel.cvut.cz

Small but Nasty Logic Synthesis Examples

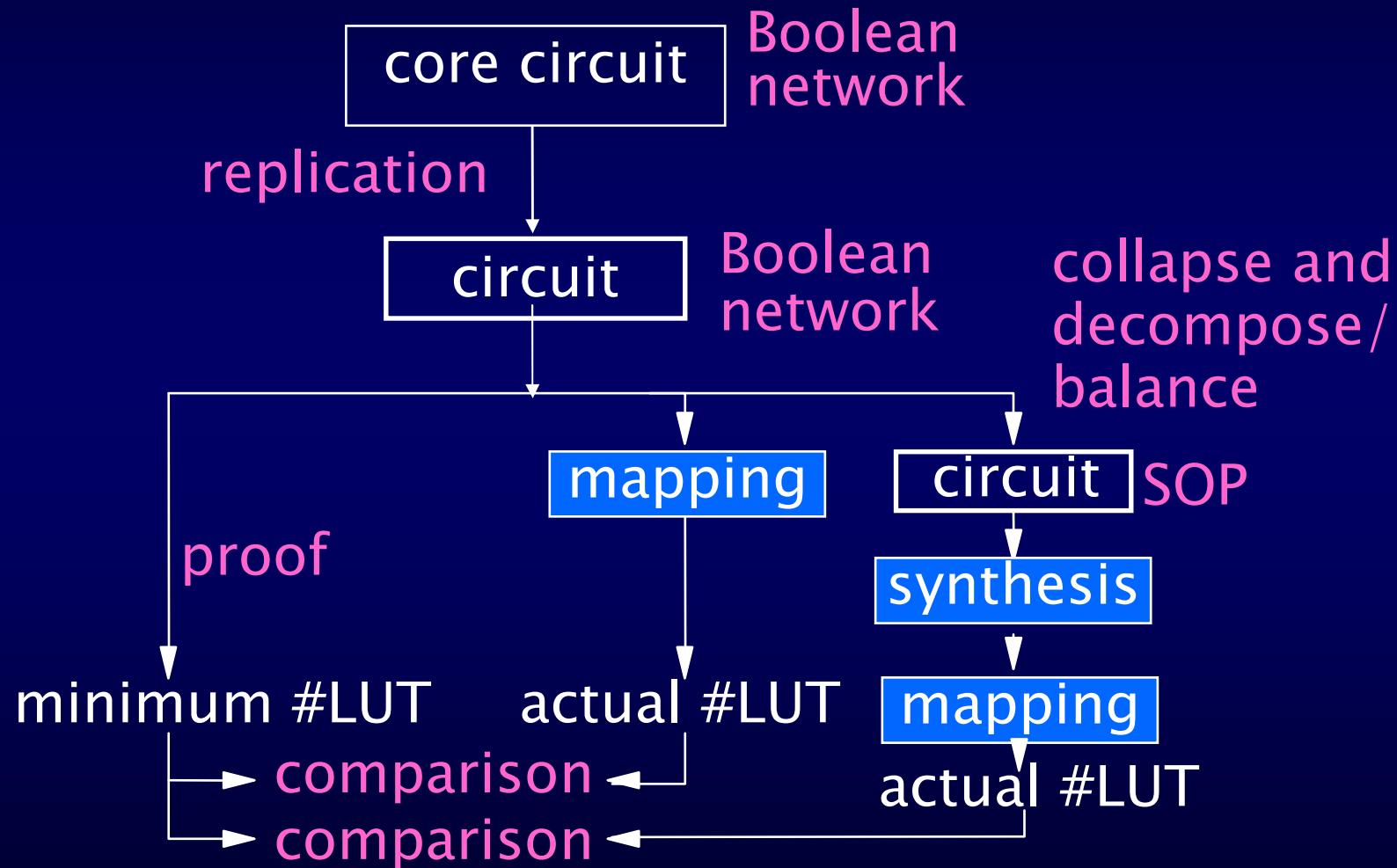
- LUT mapping examples by Cong and Minkovich
- Proposed method to construct nasty synthesis examples from existing circuits
- Possible sources of nastiness

LUT mapping examples

Jason Cong and Kirill Minkovich
VLSI CAD Lab, UCLA

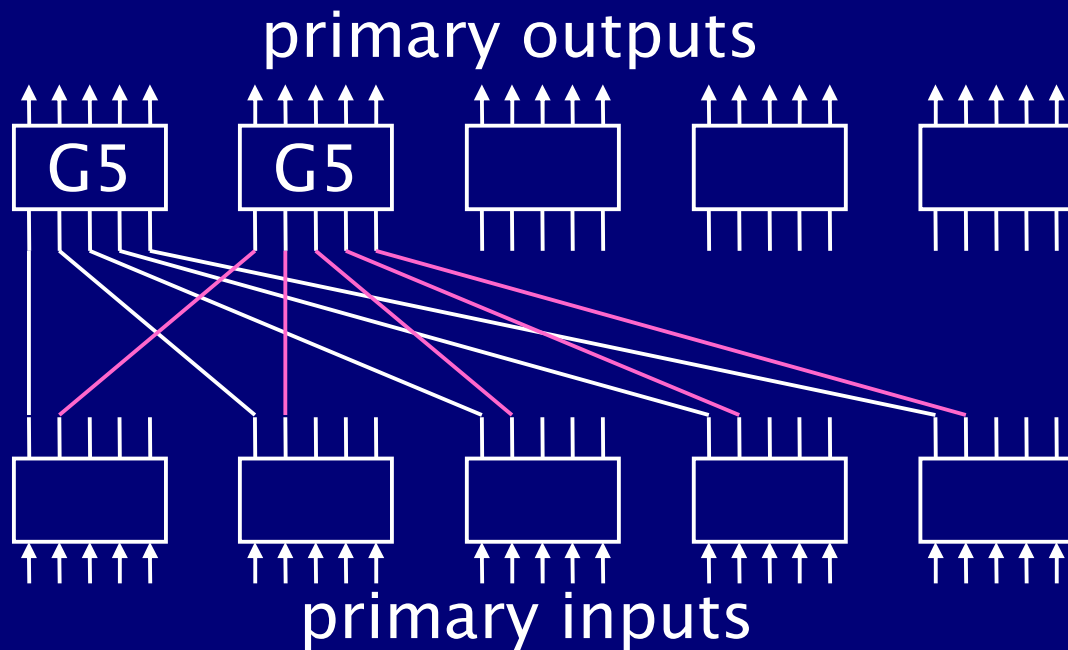
J Cong and K Minkovich,
“Optimality study of logic synthesis for LUT-based
FPGAs” (2007). IEEE Transactions on Computer-Aided
Design of Integrated Circuits and Systems. 26 (2), pp.
230–239. Postprint available free at:
<http://repositories.cdlib.org/postprints/2376>

Example construction



Core circuit and replication

G25



- G5
 - 5 inputs,
 - 5 outputs
 - Boolean network:
 - 24 nodes of 2 inputs
 - proven optimum mapping is 7 4-LUT
- G6 similar

- Several rows of core circuits
- Connection makes a path from each primary input to each primary output

Tested synthesis processes

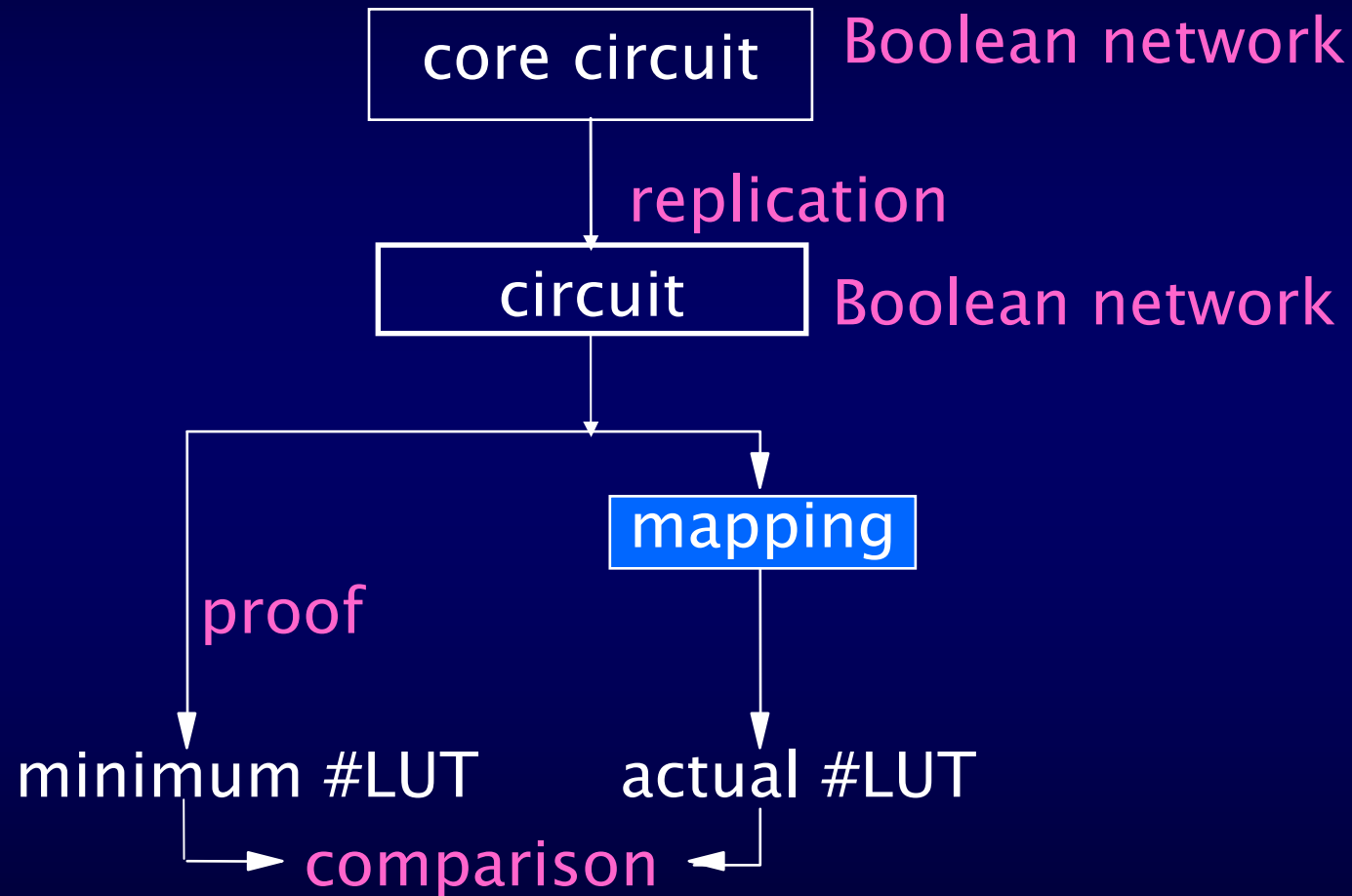
- Optimizing for area
- 4-LUT architecture

Altera	Quartus 5.0
Xilinx	ISE 7.1i
UCLA	DAOmap
Berkeley	ABC

LEKO

- Logic synthesis Examples with Known Optimals
- **Proof:** optimum mapping of the composite circuit is the composition of core circuit mappings
- **Input:** Boolean network of composite circuit
- **Tested:** the LUT mapping process
- **Metrics:** LUT#
- **Average result:** 1.15 times the optimum
- **Worst result:** 1.25 times the optimum

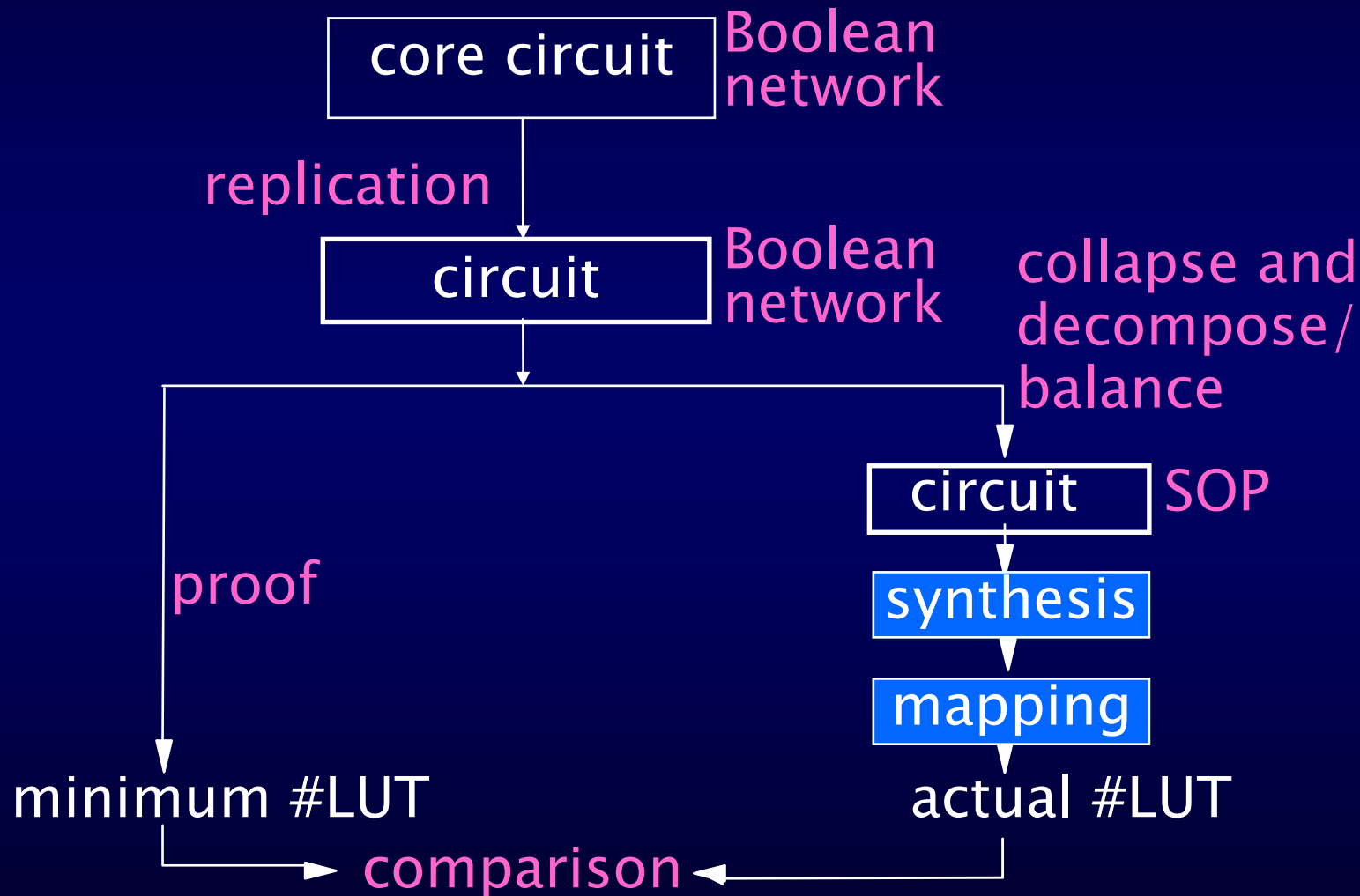
LEKO benchmarks



LEKU

- Logic Examples with Known Upper bounds
- Describe by SOP (collapse) and:
 - decompose to AND-OR gates (CD benchmarks)
 - decompose to AND-OR gates as a separate circuit for each output (CD' benchmarks)
 - balance the AND-OR trees (CB benchmarks)
- **Input:** a circuit implementing the SOP
- **Tested:** the entire logic synthesis process

LEKU - Logic Examples with Known Upper bounds



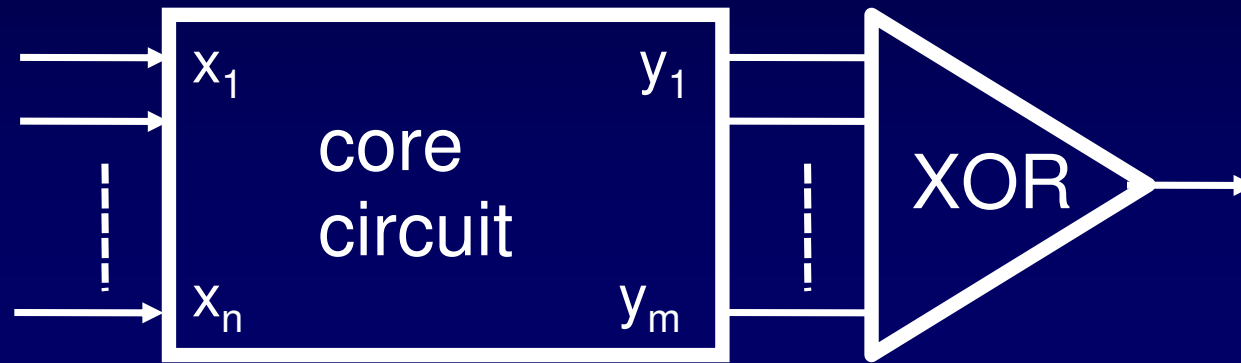
Results

Benchmark	Best ratio	Worst ratio
G25 decomposed to AND-OR circuit	148	436
G25 decomposed to separate AND-OR circuit for each output	72	504
G25 decomposed to AND-OR circuit, balanced	2.7	4.6

Reactions

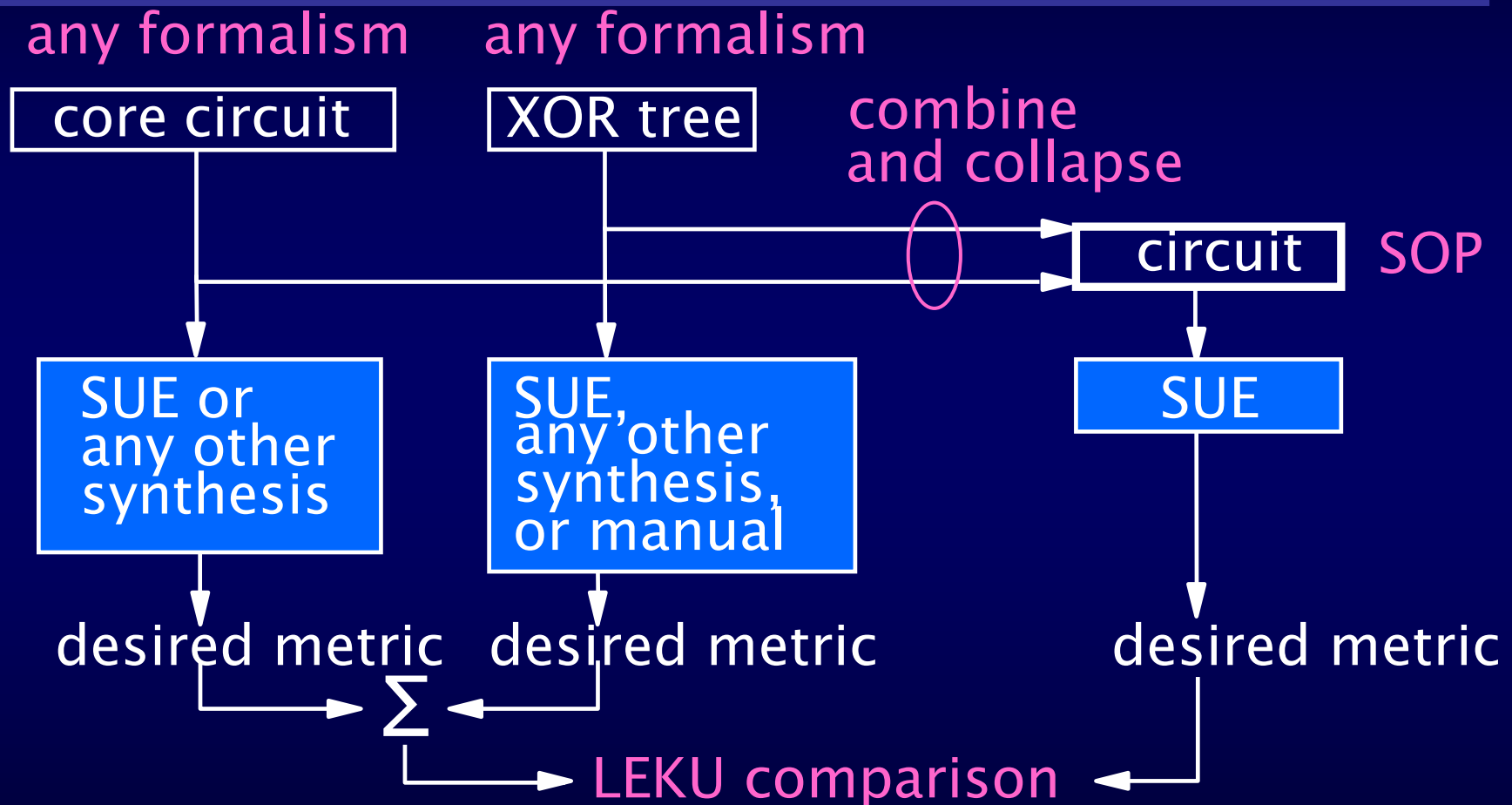
- The circuits are too large to be realistic
- Global optimization is missing, the tools cannot rediscover the original structure
- Improvements have been found:
 - use external don't cares
 - map to 5-LUT first, then to 4-LUT;
sometimes gives good results

Proposed circuits



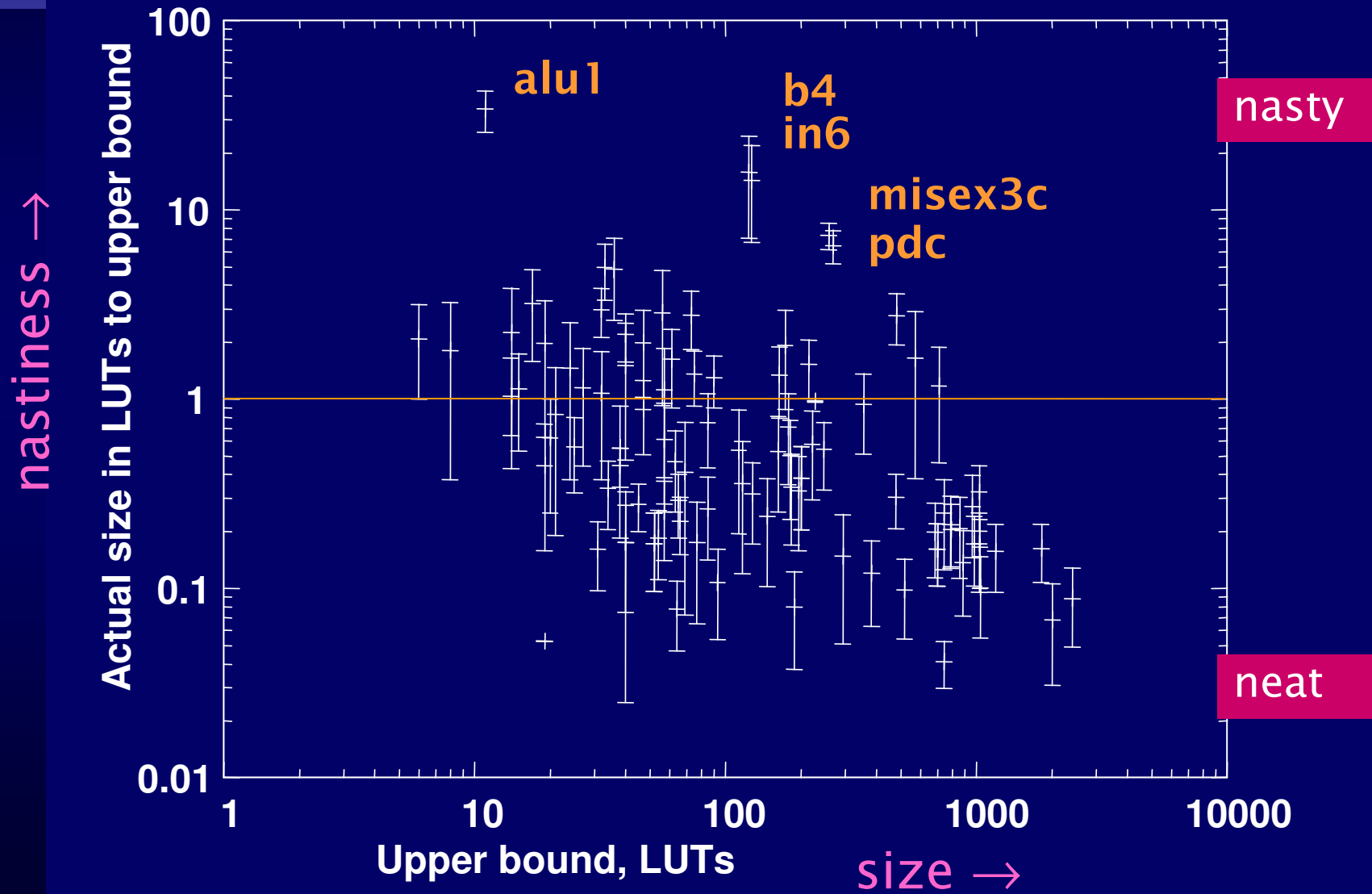
- Any core circuit, any formalism
- Any size-related metrics
- Upper bound:
the size of the core circuit plus the size of the XOR tree as implemented by the same synthesis process (or any other synthesis)

Construction

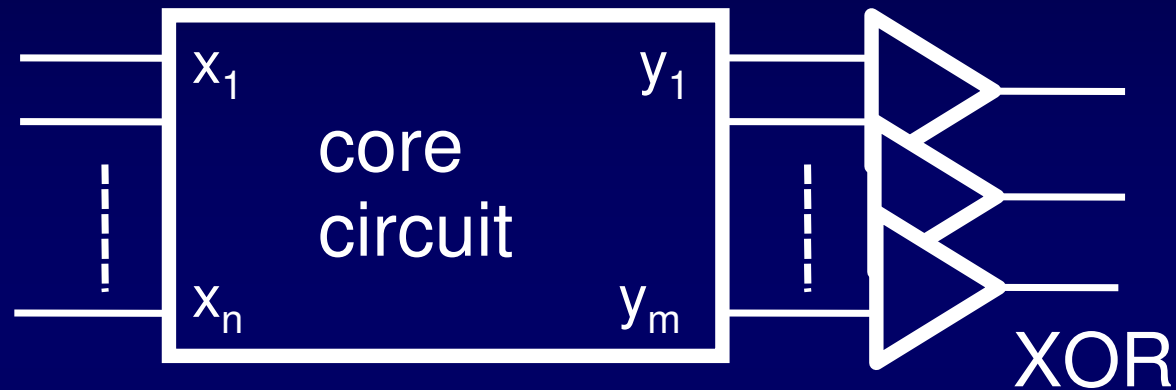


SUE: Synthesis Under Evaluation

Results: MCNC



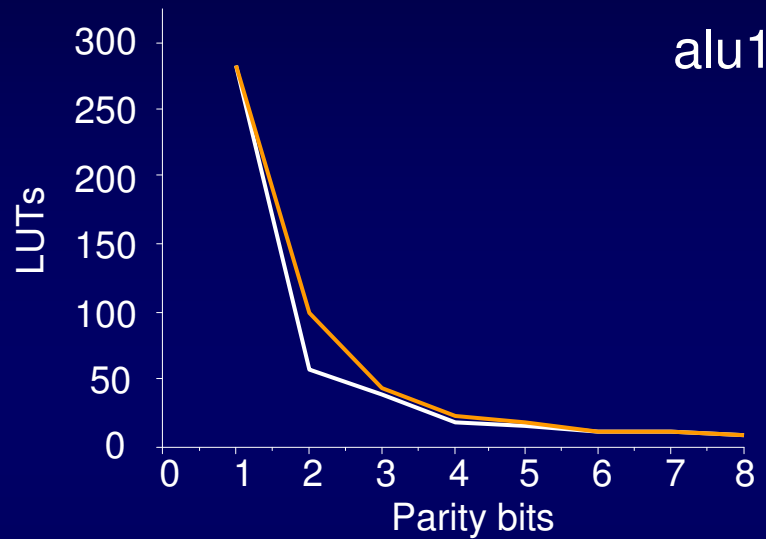
Between the core circuit and the complete circuit



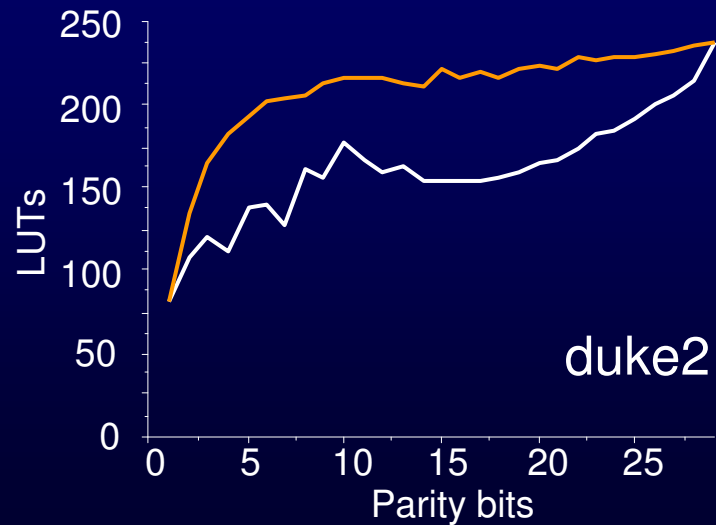
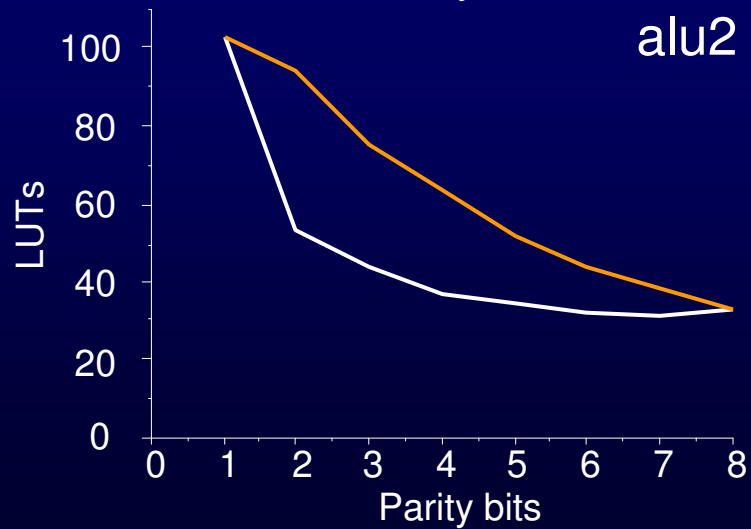
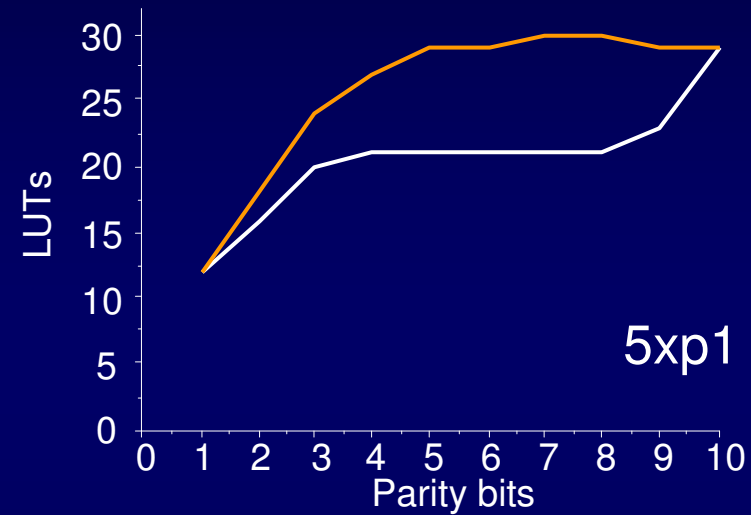
- outputs divided into disjoint groups
- randomly or by a 'clever' process
- one group: the complete circuit
- m groups: the core circuit

Nastiness is robust

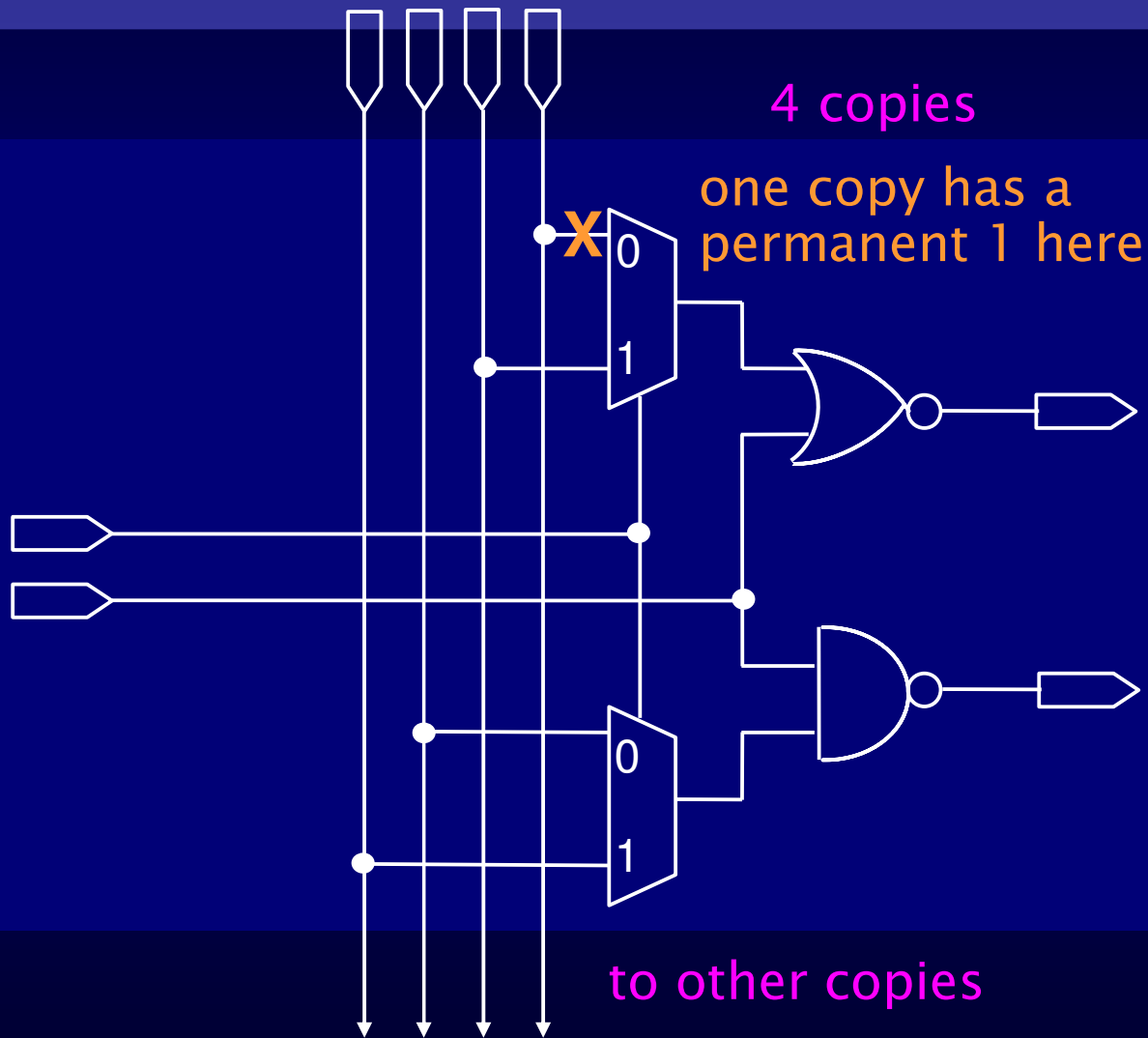
nasty



neat



The beauty of alu1

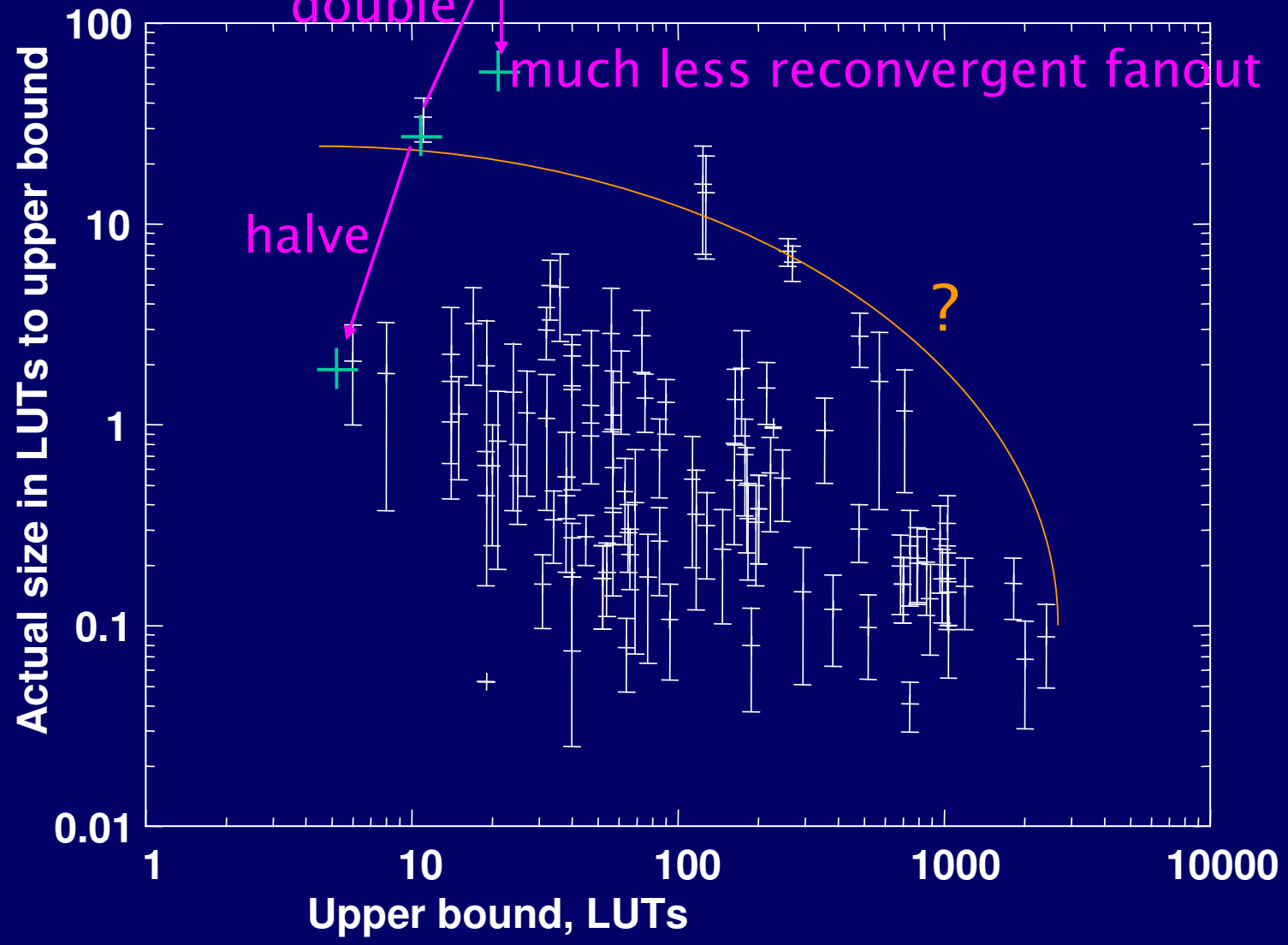


ah, the Human Designer Creativity ...

Arithmancy Magic ...

no tool can reinvent The Adder ...

Fiddling with alu1



Symmetric functions?

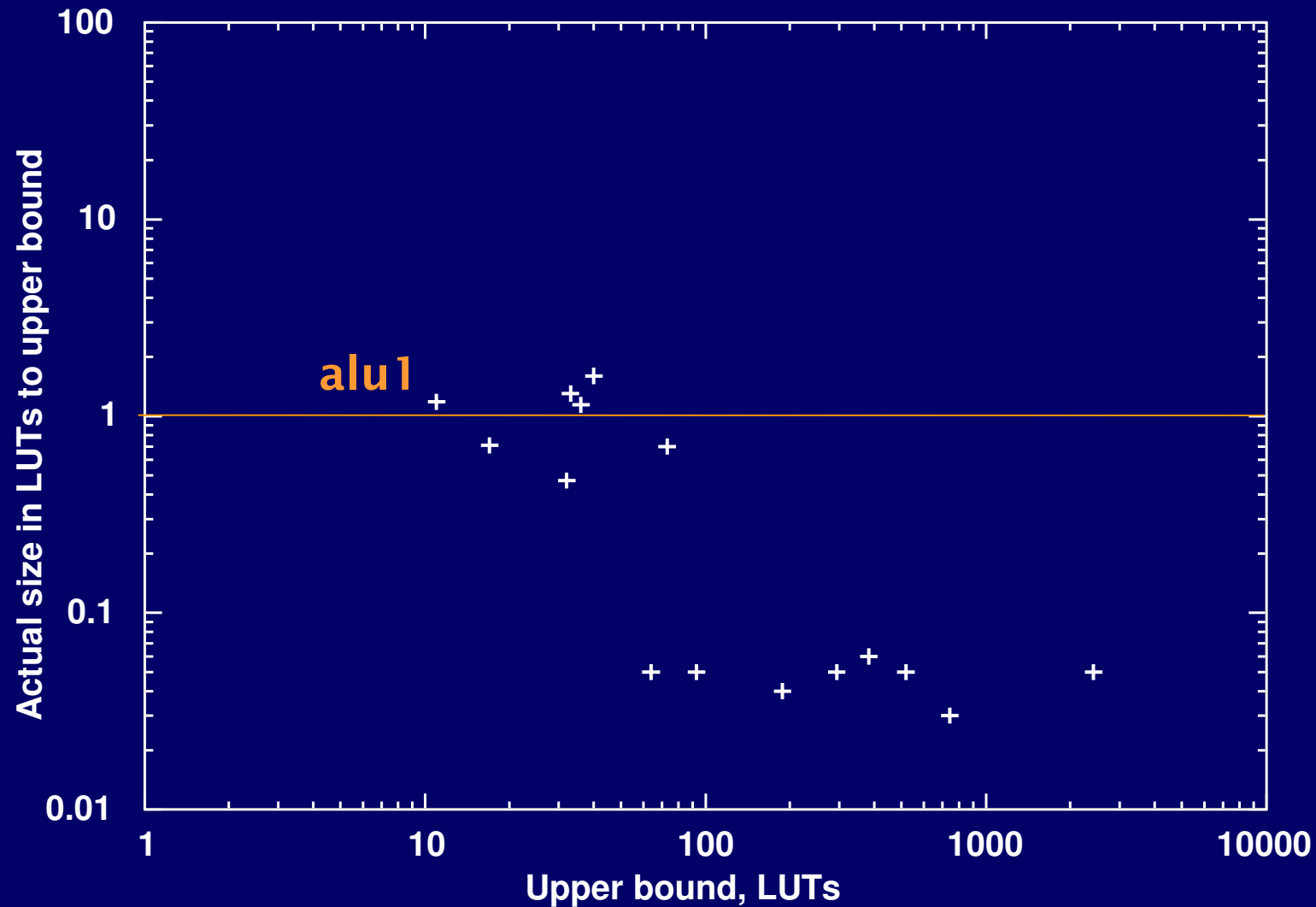
- Symmetric functions have logarithmic depth
- Algebraic decomposition is not very good at discovering that
- F Wang and D L Dietmeyer, “Exploiting Near Symmetry in Multilevel Logic Synthesis”.
IEEE Trans. on CAD 17(9), pp. 772–781
- The notion of near symmetry
- An example very similar to our circuits:
 $\text{MCNC } f51 + S^8_{\{4\}}$
- In the benchmarks, nastiness seems to be **independent** of symmetry

Biased representations?

- BDDs are IF–THEN–ELSE based
- ESOP synthesis is focused on XOR–AND
- Everything else is AND–NOT oriented
- The structure of representation often guides decomposition
- BDS: decomposition wrt. AND, XOR, MUX

- C Yang, V Singhal, M Ciesielski: “BDD Decomposition for Efficient Logic Synthesis.” 1999 IEEE International Conference on Computer Design (ICCD'99) p. 626
- C Yang, M Ciesielski, V Singhal: “BDS: a BDD–based logic optimization system.” Proceedings of the 37th conference on Design automation, p.92–97, June 05–09, 2000, Los Angeles, California

Biased representations.



The beauty of BDS

- 4 classes under NPN-equivalence
- 2 of them essentially dependent on both inputs: $x_1 + x_2$, $x_1 \oplus x_2$
- BDS can provide the negations for the NPN derivation
- BDS can decompose using any of the two NPN classes
- BDS is a
“NPN-complete” decomposition

To Do

- how come the experiments are incomplete...
- how does BDS perform on the LEKU benchmarks by Cong and Minkovich?
- what can we learn from newer benchmarks?
- can we take the BDS algorithms further...?

Conclusions

- Contemporary logic synthesis, both academic and professional, can fail on circuits that are small and practical
- A part of this failure can be attributed to unequal handling of AND/OR and XOR operators in decomposition

Welcome to the years of
Logic Synthesis II