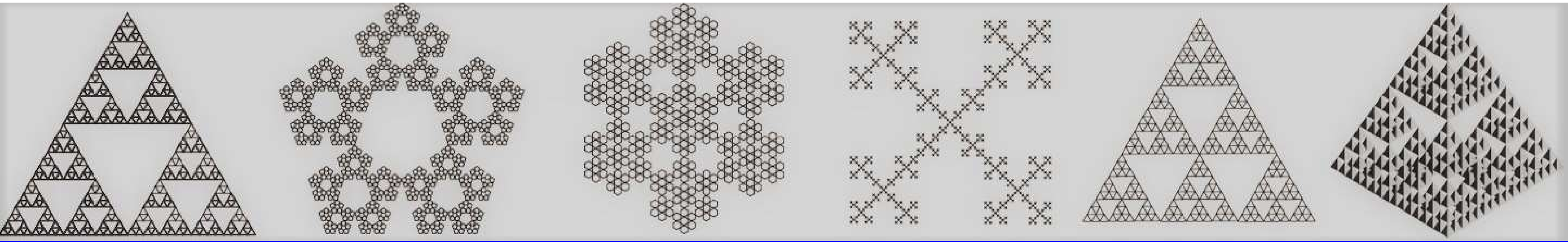


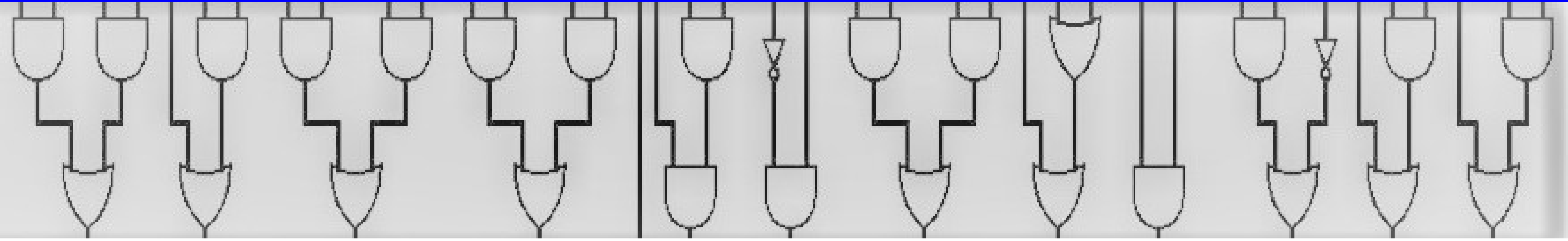
Recursive Implementation of Voting Networks

Behrooz Parhami

Department of Electrical and Computer Engineering

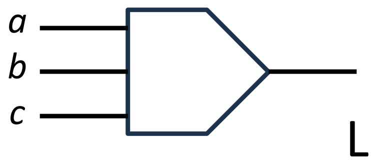
University of California, Santa Barbara, USA

IEEE 14th Annual Computing and Communication Workshop and Conference
Las Vegas, Nevada, USA, January 2024



Voters or Voting Networks

Majority voters (for TMR & NMR redundancy)



2-out-of-3	$ab \vee bc \vee ca$
Larger voters	
3-out-of-5	10 terms
4-out-of-7	35 terms
5-out-of-9	126 terms
In general	
k-out-of-$(2k-1)$	$\binom{2k-1}{k}$

Bit-voters

vs.

Word-voters

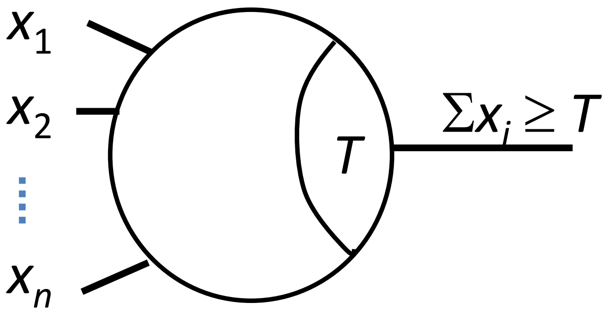
Number of inputs need not be odd

4-out-of-6

5-out-of-8

6-out-of-8

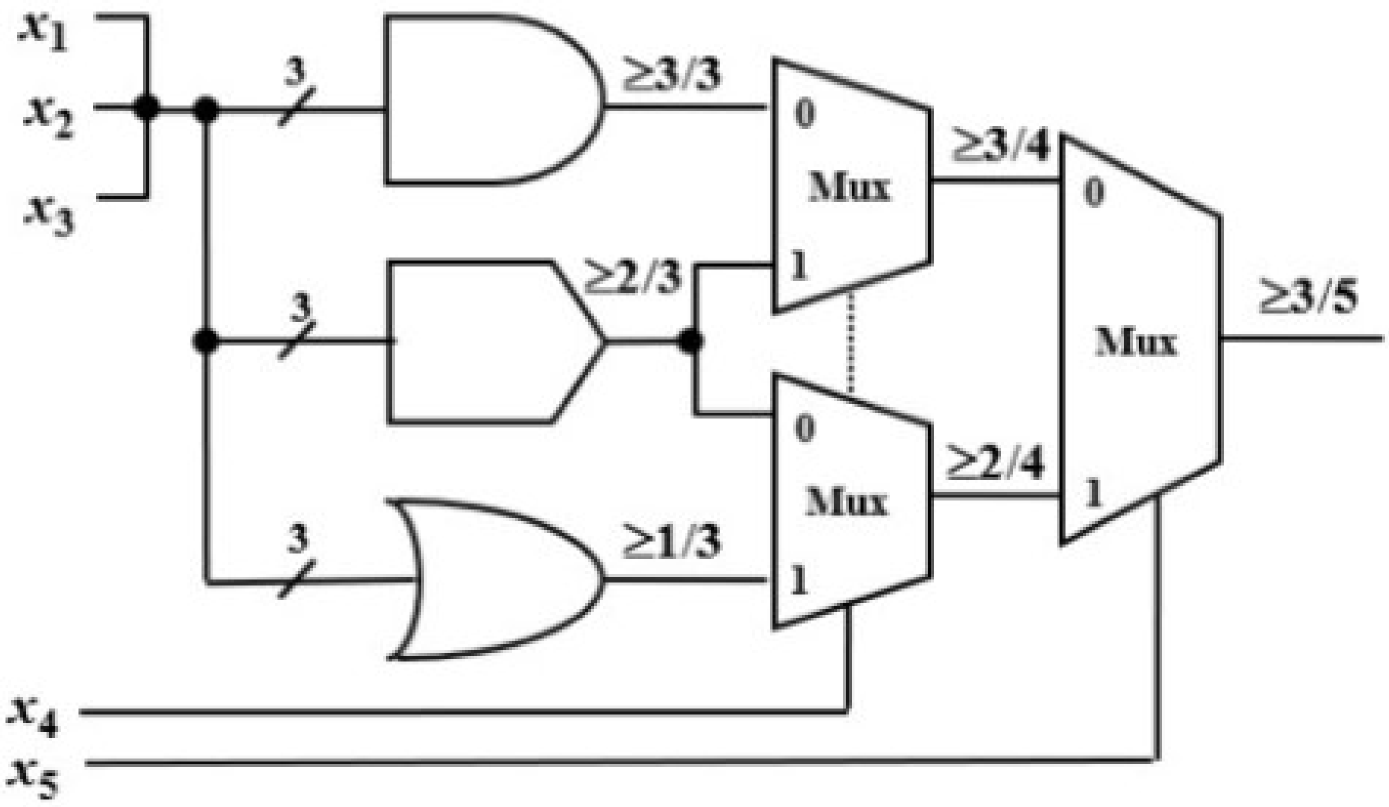
3-out-of-8



Recursive Implementation of a ≥3/5 Voter

Shannon expansion or decomposition

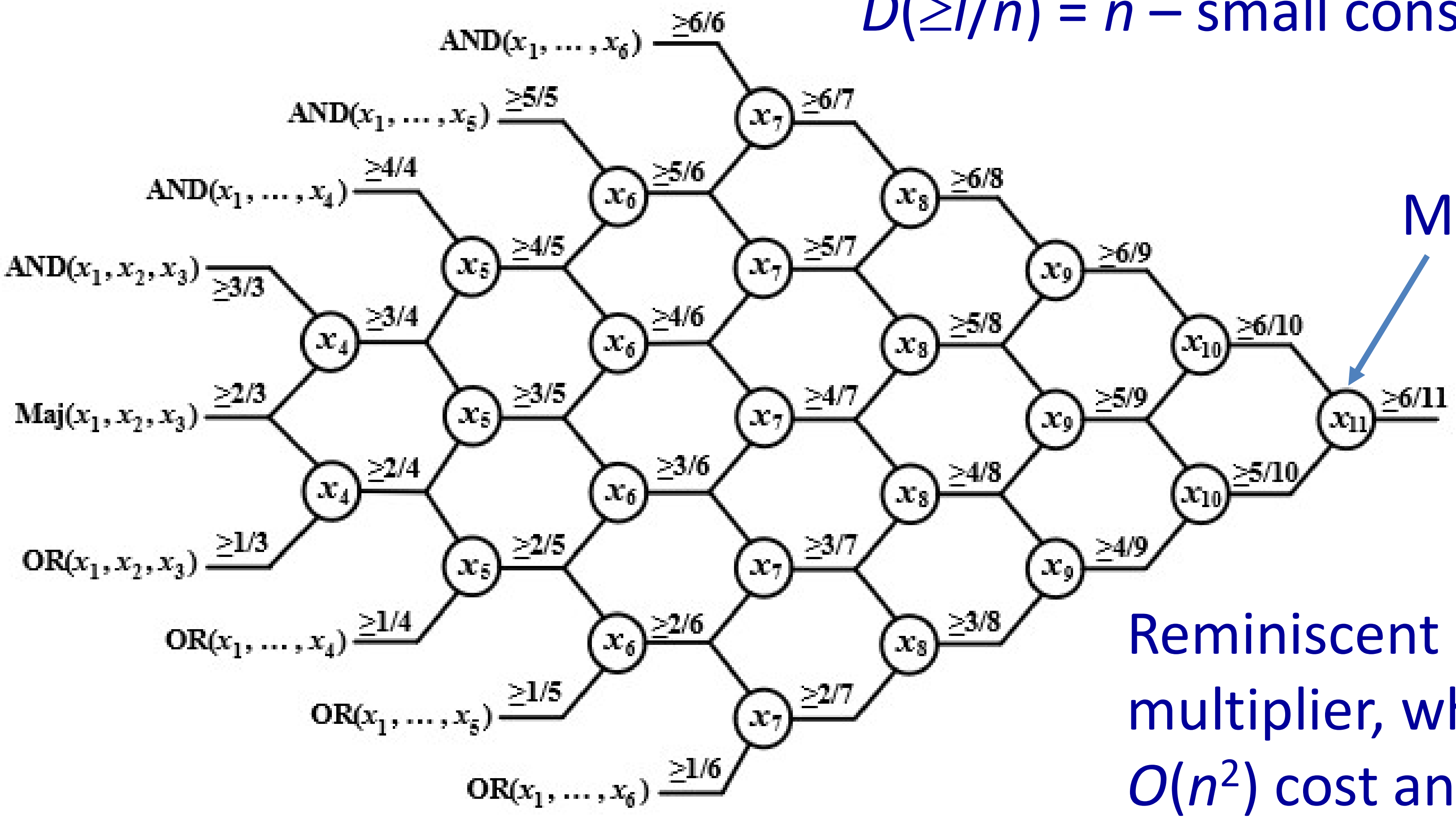
$f(x_1, x_2, \ldots, x_{n-1}, x_n) = x_n' f(x_1, x_2, \ldots, x_{n-1}, 0) \vee x_n f(x_1, x_2, \ldots, x_{n-1}, 1)$



Cost and Delay Formulas for ≥*l*/*n* Voter

$C(\geq l/n) = (n - l)(l - 1)$ + linear terms

$D(\geq l/n) = n$ – small constant

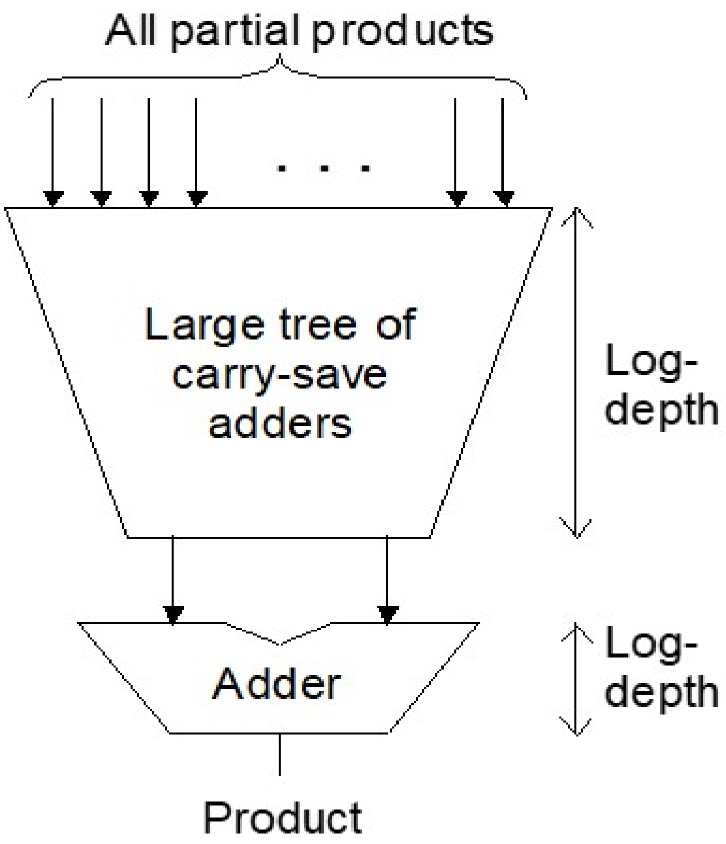


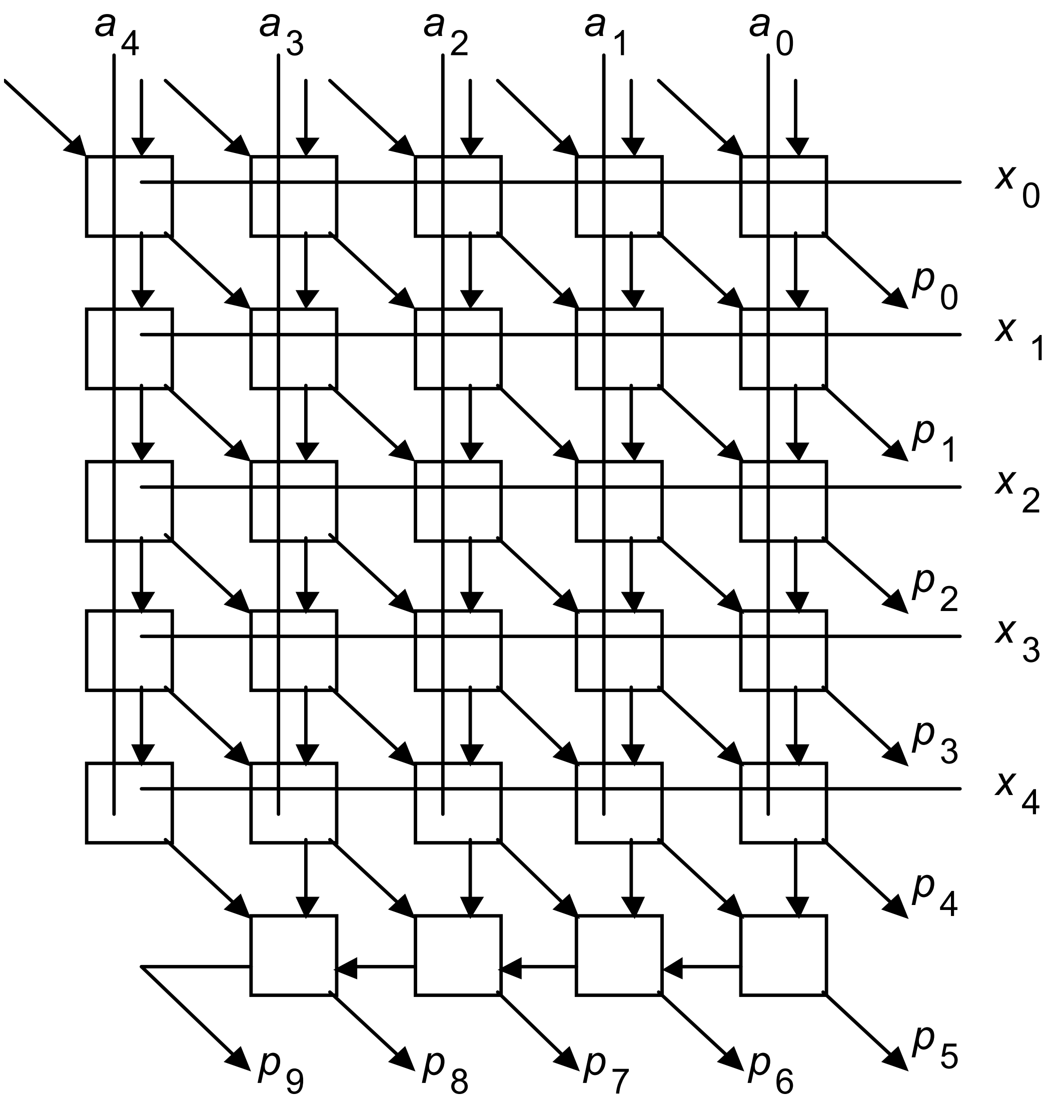
Mux

Reminiscent of array multiplier, which also has $O(n^2)$ cost and $O(n)$ delay

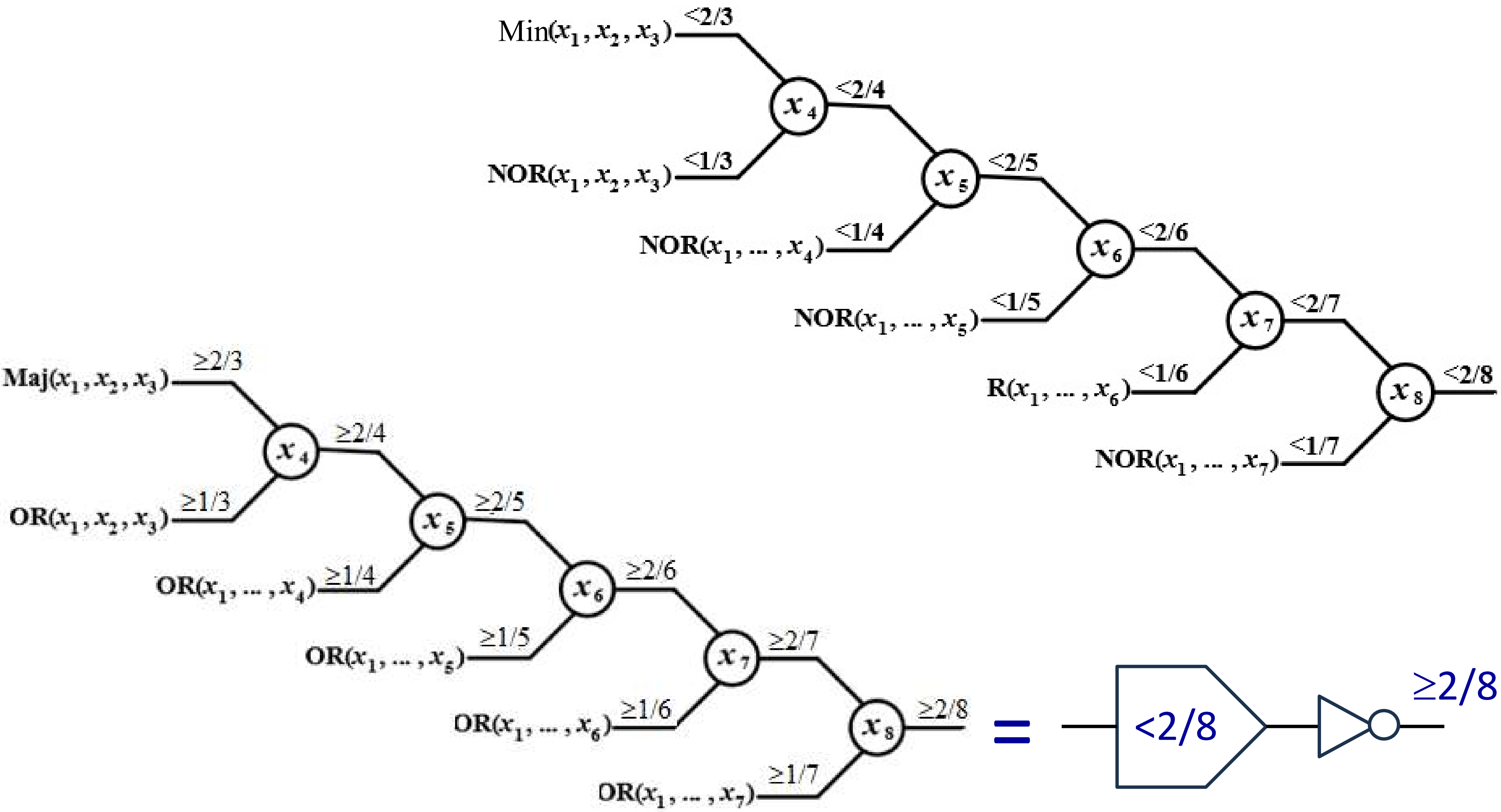
Theoretical Speed vs. VLSI-Friendliness

- Tree: Fast, but irregular
- Array: Slow, but regular

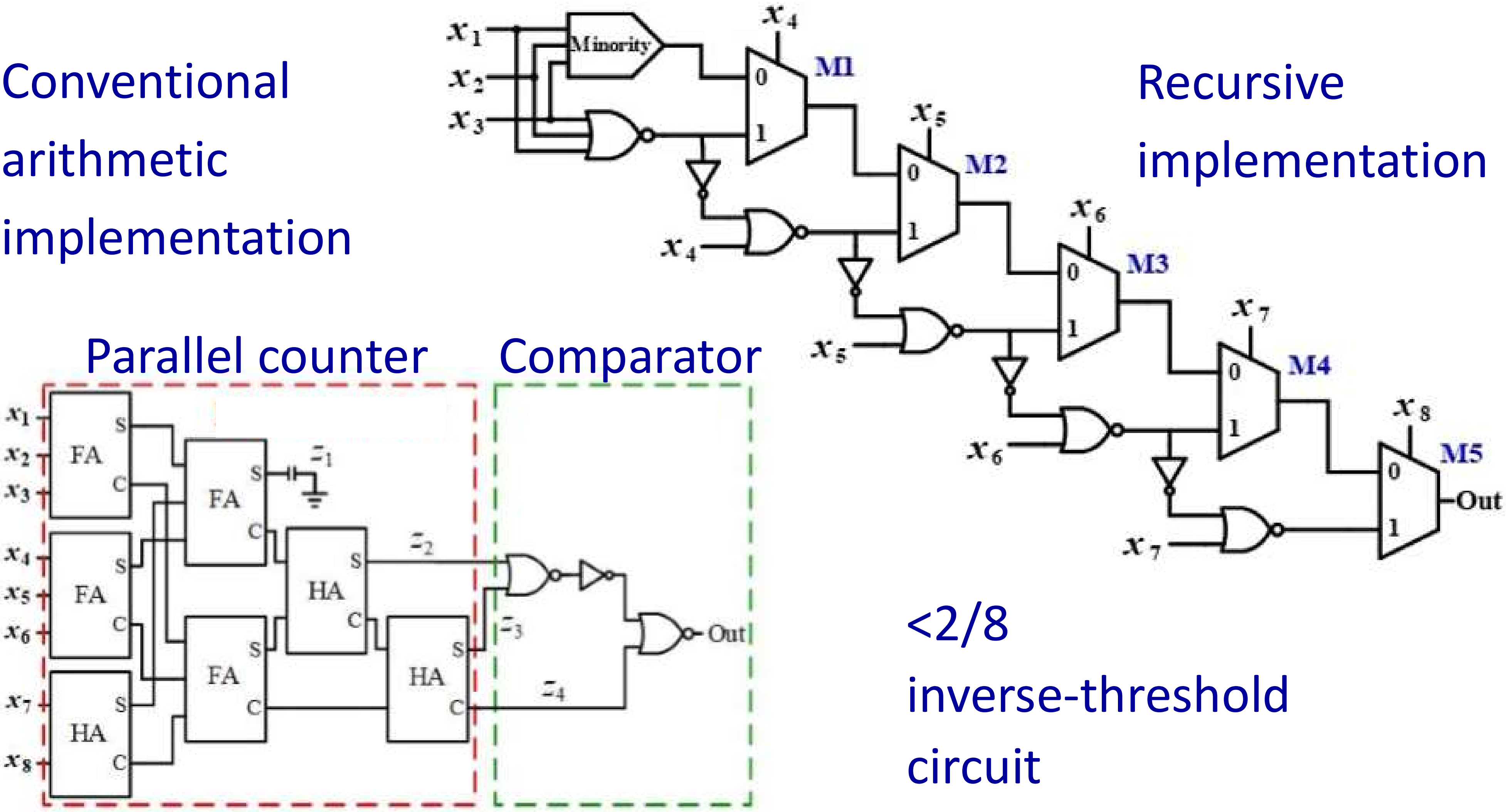




Recursive Implementation of a <2/8 Circuit



Comparisons with Prior Designs



Conventional arithmetic implementation

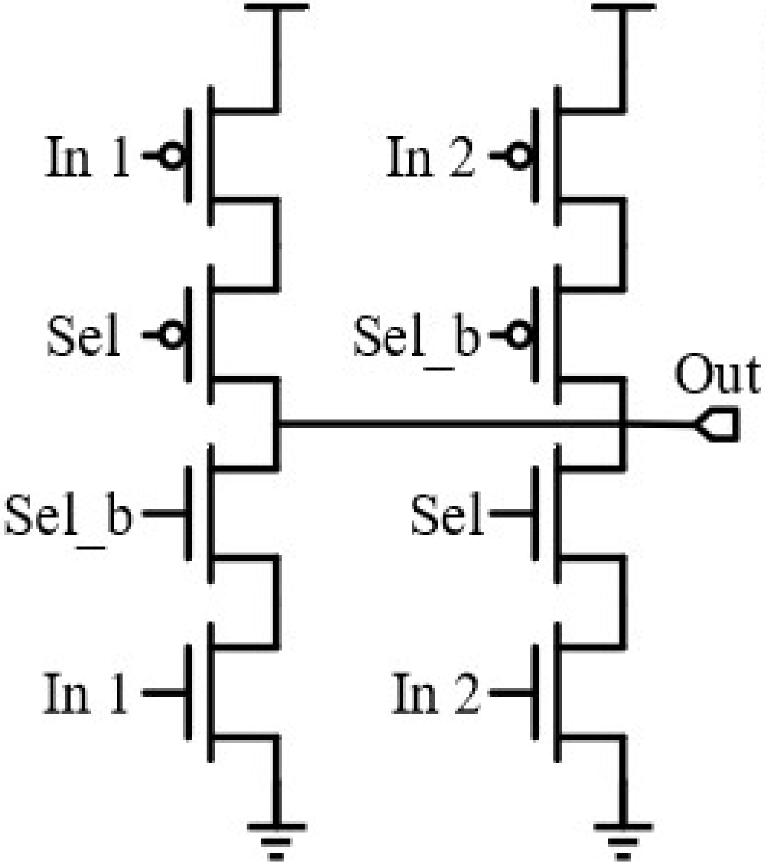
Recursive implementation

Parallel counter

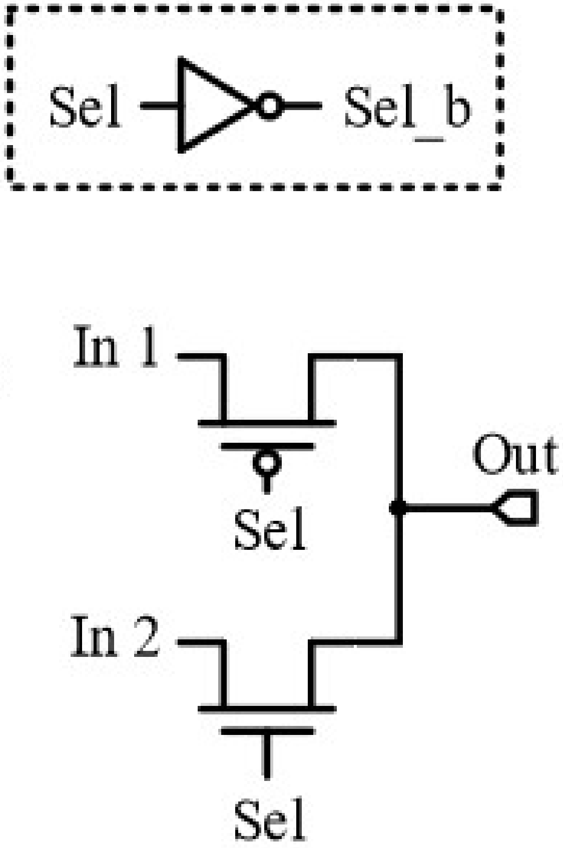
Comparator

<2/8 inverse-threshold circuit

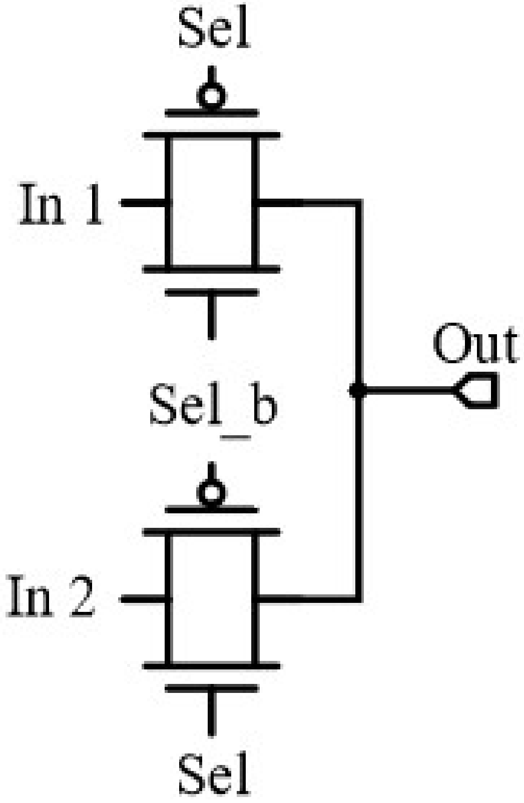
Multiplexer Options



Ordinary CMOS

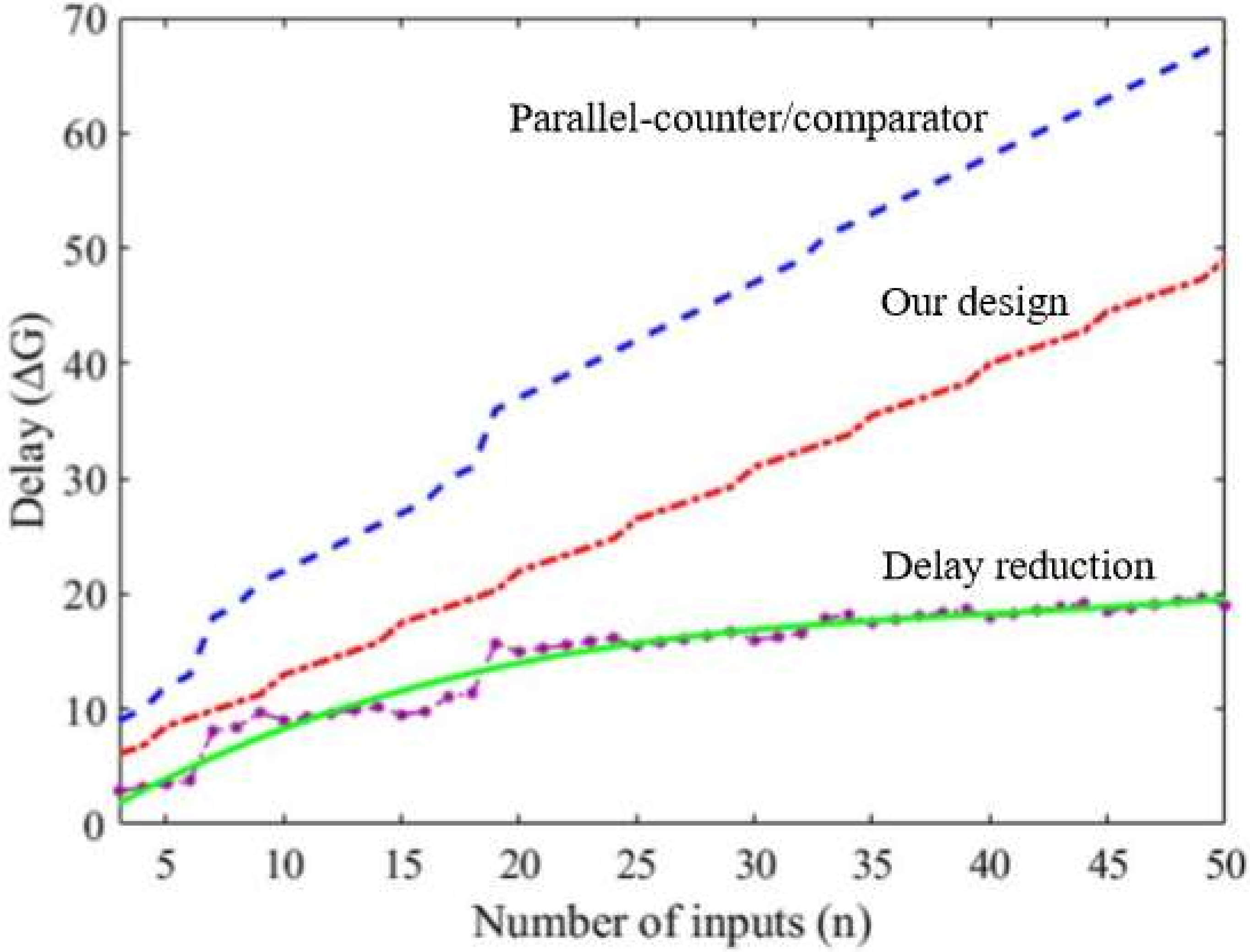


Bypass transistor



Transmission gate

Speed, Area, Power, Energy Gains



Absolute delay reduction increases with *n*, but relative reduction decreases

Reductions achieved over 5 different implementations

Delay: 18%

Transistors: 51%

Power: 54%

Energy: > 60%

Advantages and Drawbacks

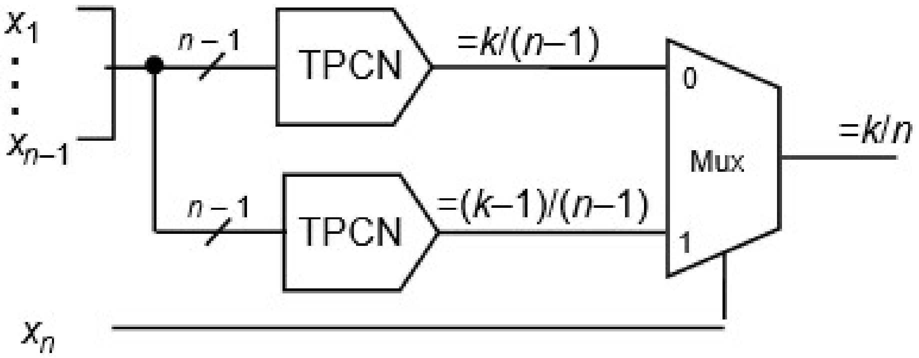
- Recursion not applicable to all of our needs
- May not lead to theoretically-optimal design
- But … Optimal designs tend to be complex

  → Long design times and many design errors
- Recursive designs: Analyzable and verifiable
- Stop recursion upon hitting a known design
- Commonly-used parts can be fully optimized
- Good for prototyping, if not for final circuit

Recursive Design of Weight-Checkers

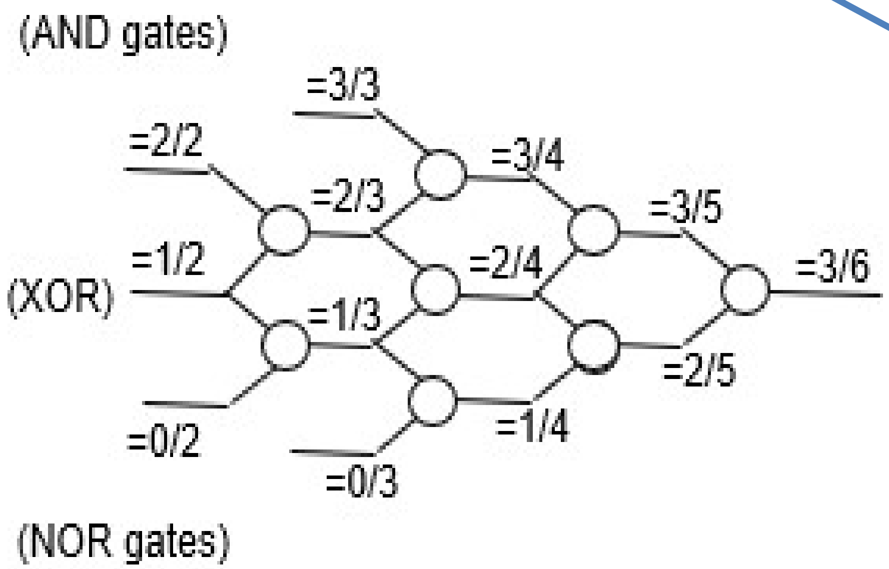
$C(=k/n) = k(n - k)$ + linear terms

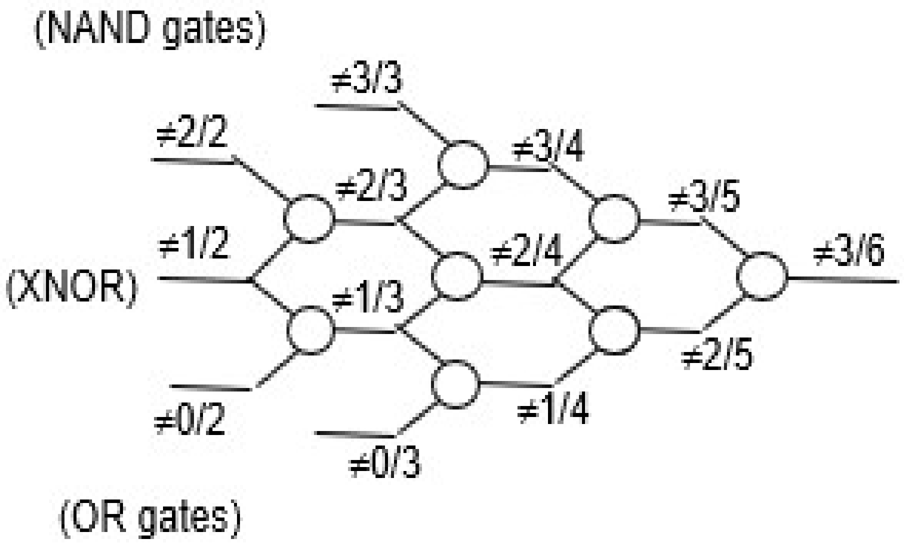
$D(=k/n) = n$ – constant



k-out-of-*n*

¬*k*-out-of-*n*





Between-Limits Threshold Counters

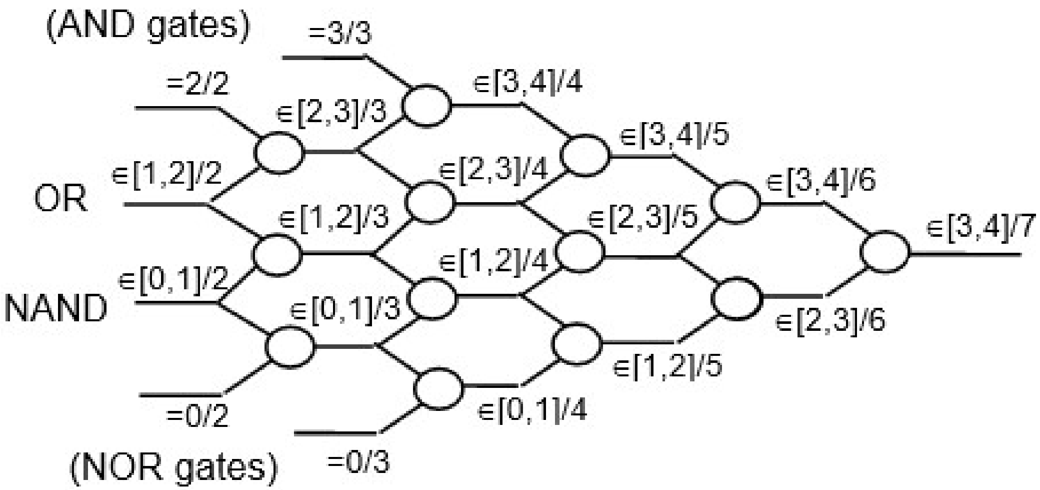
$C(\in[l, m)/n)$ = Open problem

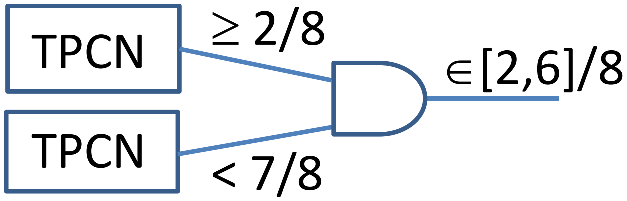
$D(\in[l, m)/n)$ = $n - 2$ + a small constant

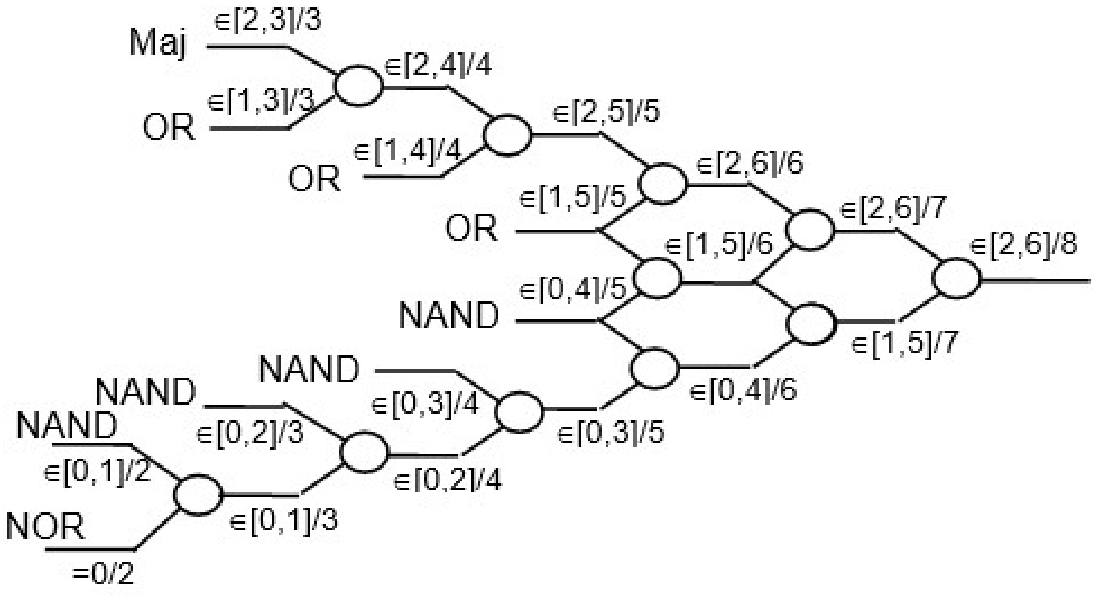
Example application:

Codewords of length 9 bits and weights 4 or 5

$C(4, 9) + C(5, 9) = 126 + 126 = 252$



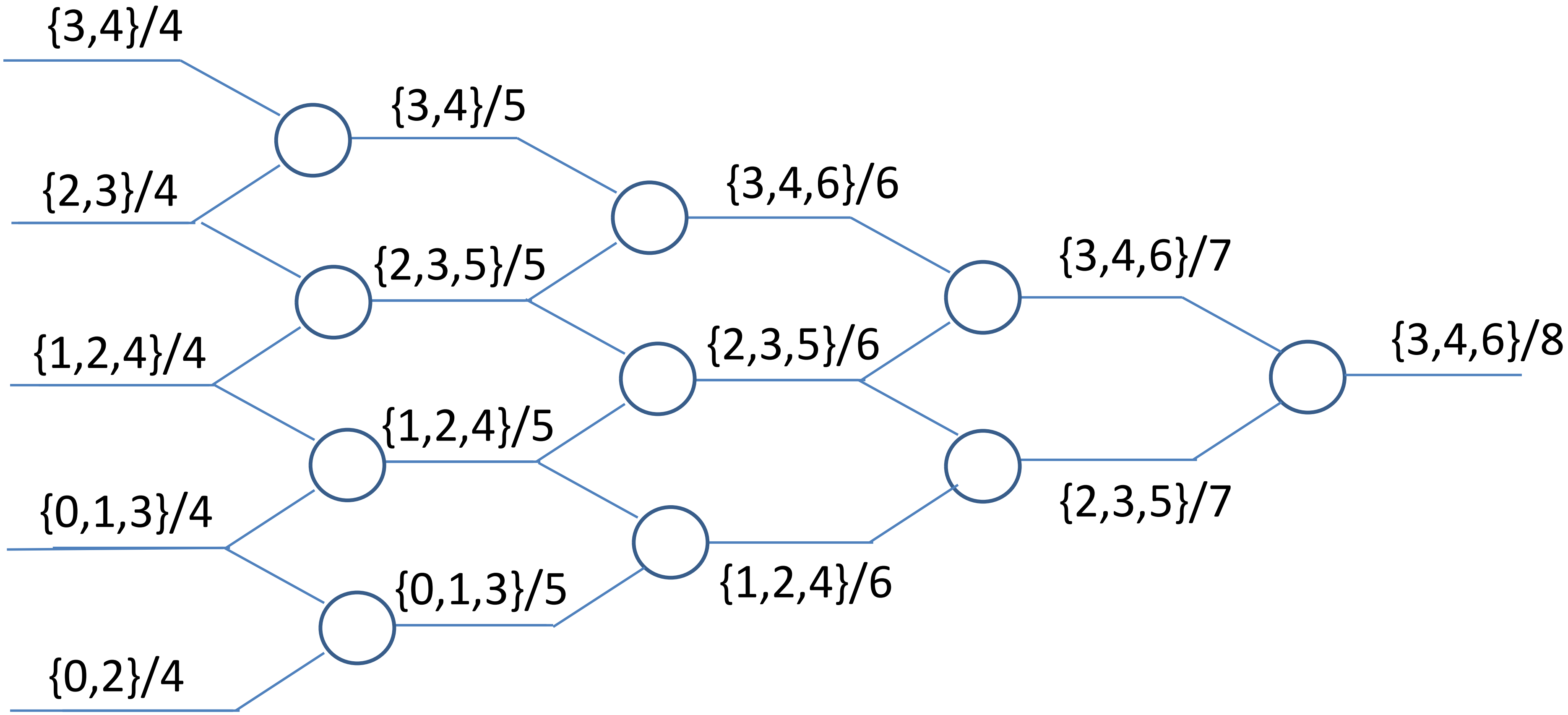




Membership Checkers

{3,4,6}/8 membership checker

Negative terms and terms larger than *n* are dropped



Questions?

parhami@ece.ucsb.edu
PDF files of B. Parhami's papers are available at:
www.ece.ucsb.edu/~parhami/publications.htm