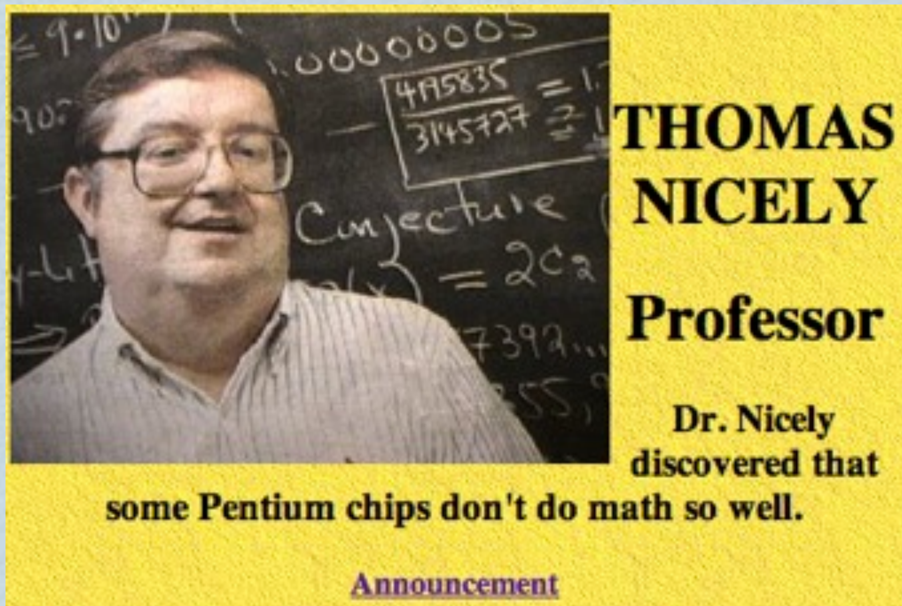
Circuit Verification: The BDD Revolution

Jean Goubault-Larrecq

Bugs: The Intel FDIV Bug



The correct value is

$$\frac{4195835}{3145727} = 1.333820449136241002$$

However, the value returned by the flawed Pentium is incorrect beyond four significant digits[9]:

$$\frac{4195835}{3145727} = 1.333739068902037589$$

Bugs: The Intel FDIV Bug

The correct value is

$$\frac{4195835}{3145727} = 1.333820449136241002$$

However, the value returned by the flawed Pentium is incorrect beyond four significant digits[9]:

$$\frac{4195835}{3145727} = 1.333739068902037589$$

Intel has to recall all flawed chips.
Cost: $475 million

Bugs: Are they Hard to Find?

- Which chip below has the FDIV bug?

Bugs: Are they Hard to Find?

- Which chip below has the FDIV bug?

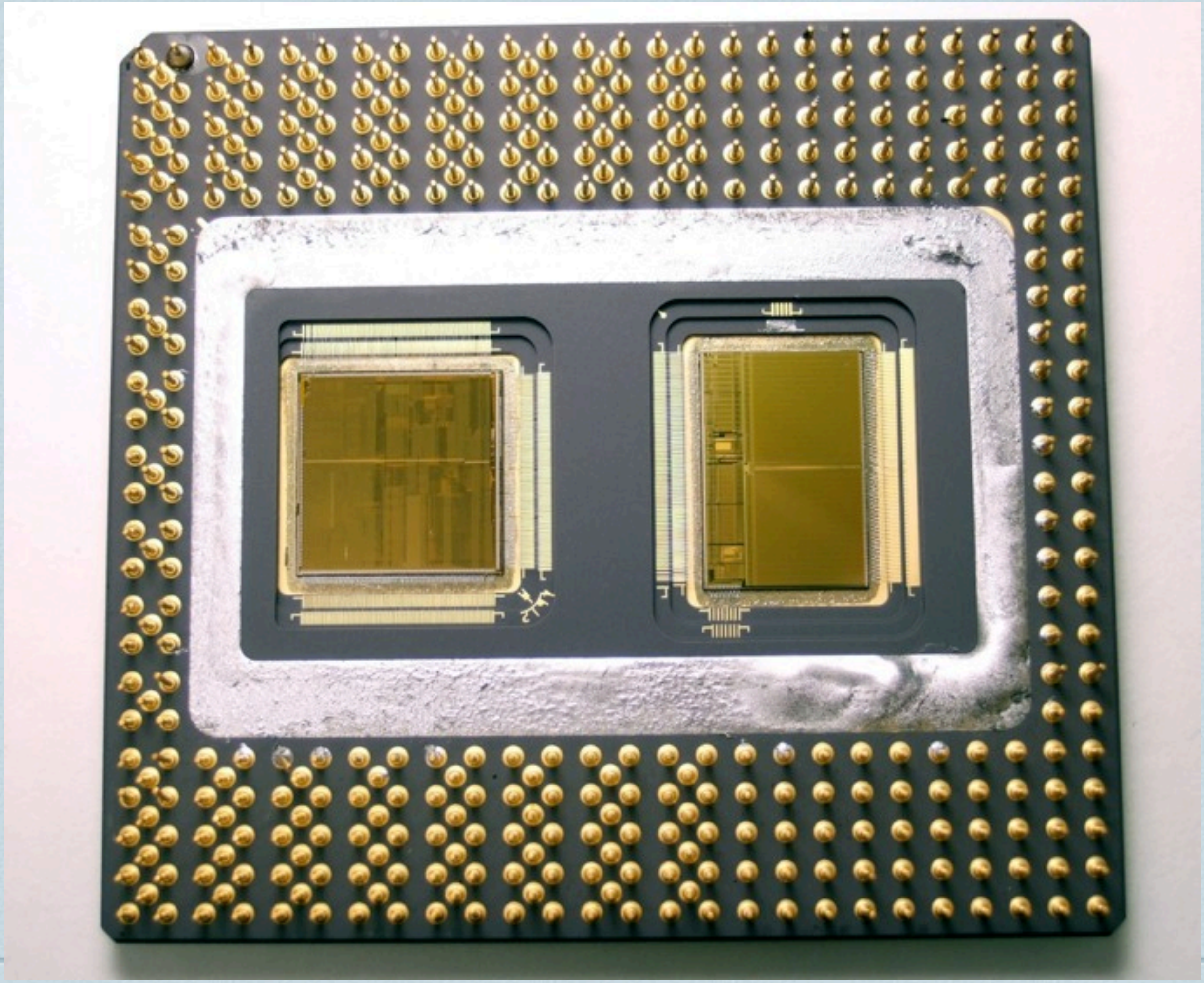
P5

P54CS

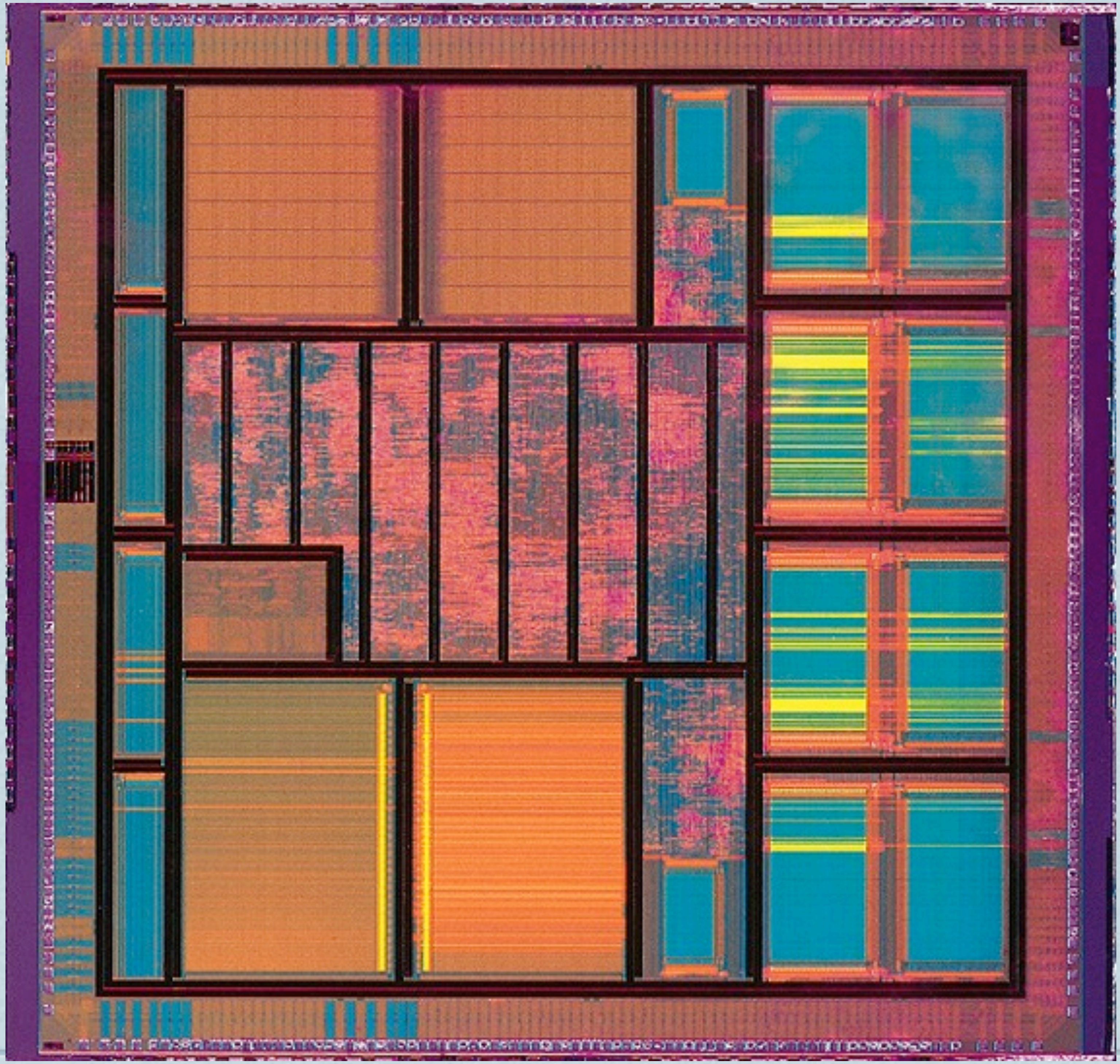
Pentium Pro

Let's Look Inside

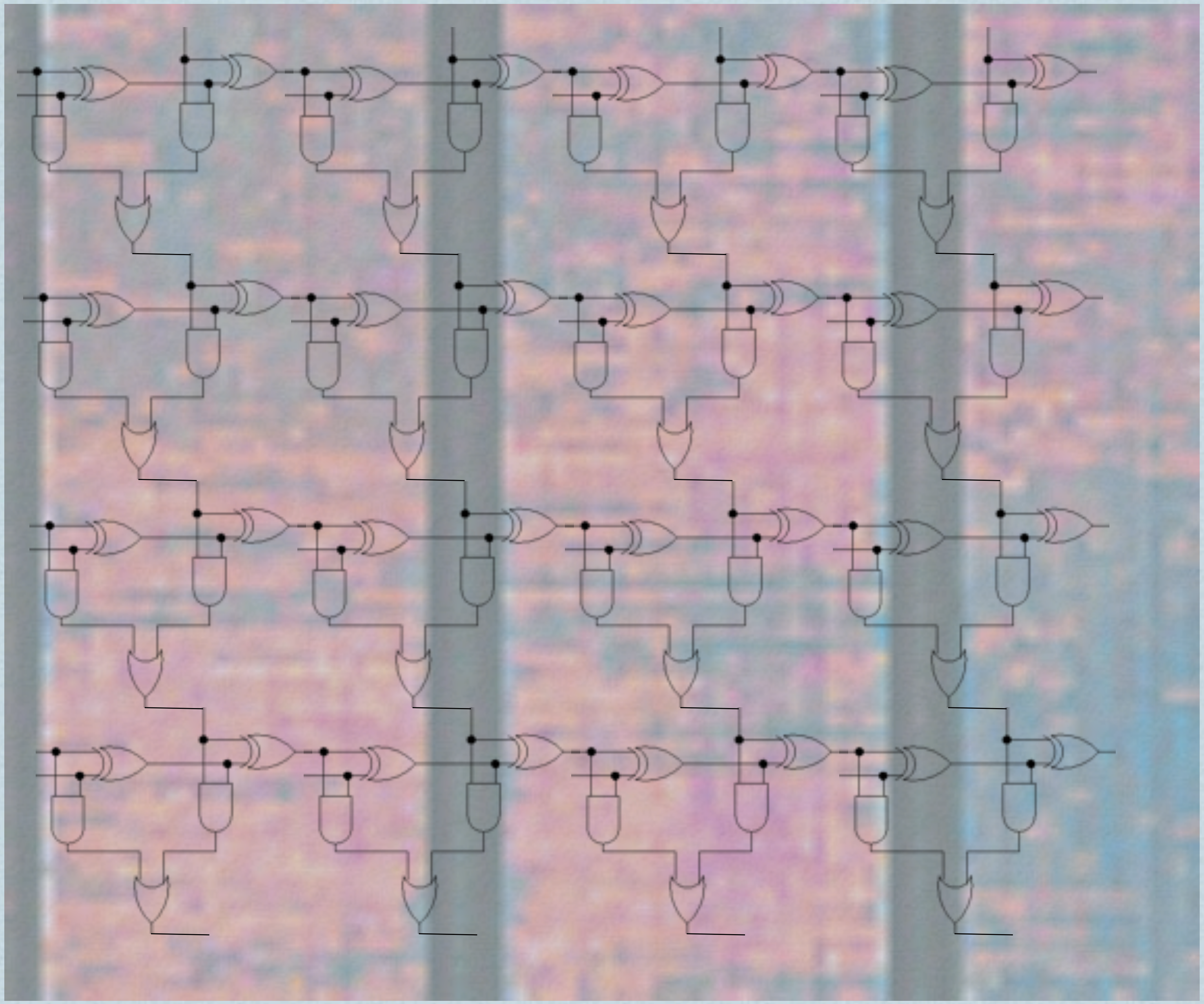
Let's Look Inside



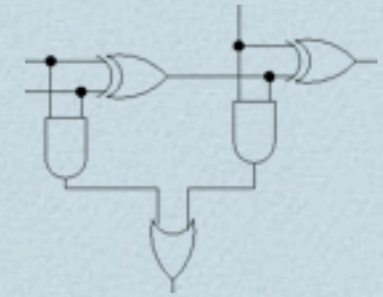
Let's Look Inside



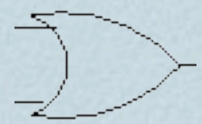
Let’s Look Inside



Circuits



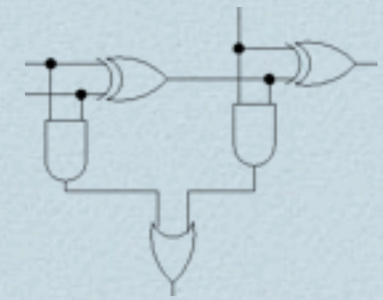
- Made of simple gates
- connected by wires

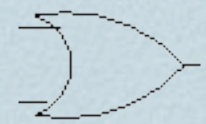


A	*B*	*A* or *B*
0	0	0
1	0	1
0	1	1
1	1	1

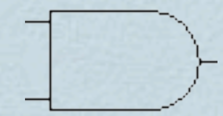
A	*B*	*A* and
0	0	0
1	0	0
0	1	0
1	1	1

Circuits





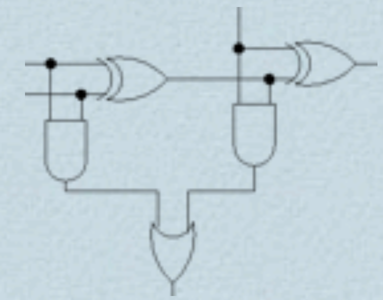
A	*B*	*A* or *B*
0	0	0
1	0	1
0	1	1
1	1	1



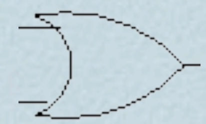
A	*B*	*A* and
0	0	0
1	0	0
0	1	0
1	1	1

- Made of simple gates
- connected by wires
- In the Pentium
  - ~ 1 million gates
  - ~ 3 million wires, each having value 0 or 1
  - $2^{3,000,000}$ possible distinct configurations to check
  - This is 1 followed by 900,000 zeros!
    (number of atoms in the universe: only 1 followed by 82 zeros...)

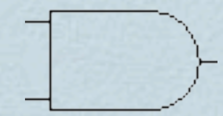
Circuits



- Made of simple gates
- connected by wires



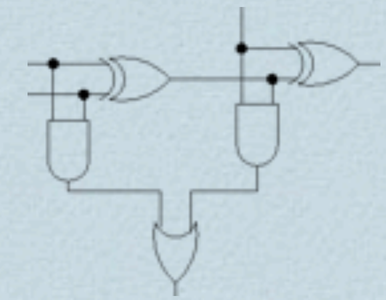
A	*B*	*A* or *B*
0	0	0
1	0	1
0	1	1
1	1	1



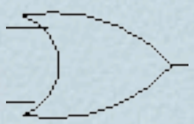
A	*B*	*A* and
0	0	0
1	0	0
0	1	0
1	1	1

- In practice, we would be happy to check elementary units (e.g., a divisor, a multiplier, an adder)
  - with a few thousand wires
  - Before 1986, this was out of reach of all known methods

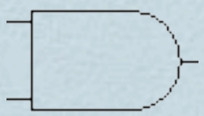
Circuits



- Made of simple gates
- connected by wires



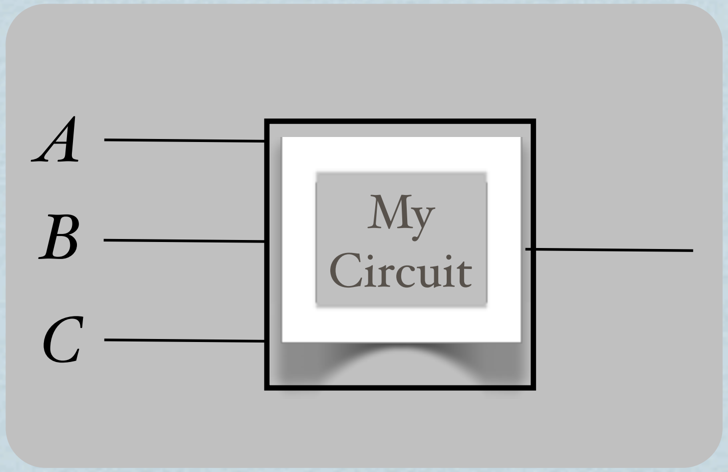
A	*B*	*A* or *B*
0	0	0
0	0	1
0	1	1
0	1	1



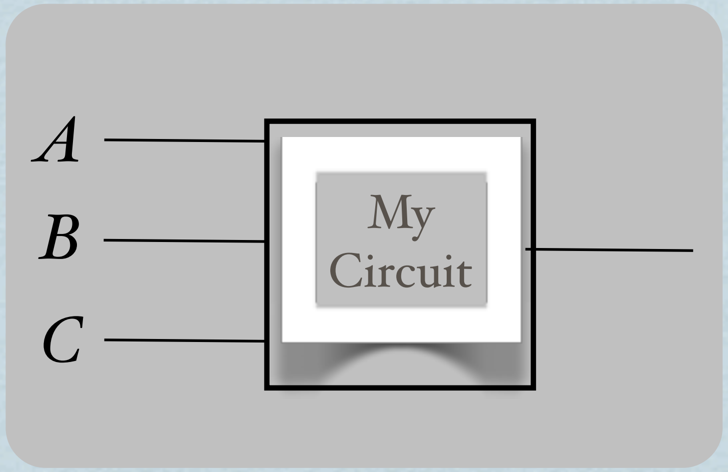
A	*B*	*A* and
0	0	0
0	0	0
0	1	0
0	1	1

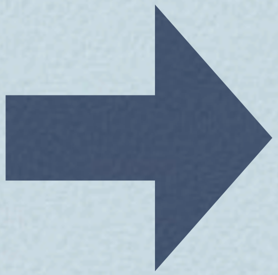
- Then **R.L. Bryant** (1986) found a way: BDDs (Binary Decision Diagrams) [improving on Sh. B. Akers (1978)]
- «One of the only really fundamental data structures that came out in the last twenty-five years» — Donald E. Knuth

Circuits...



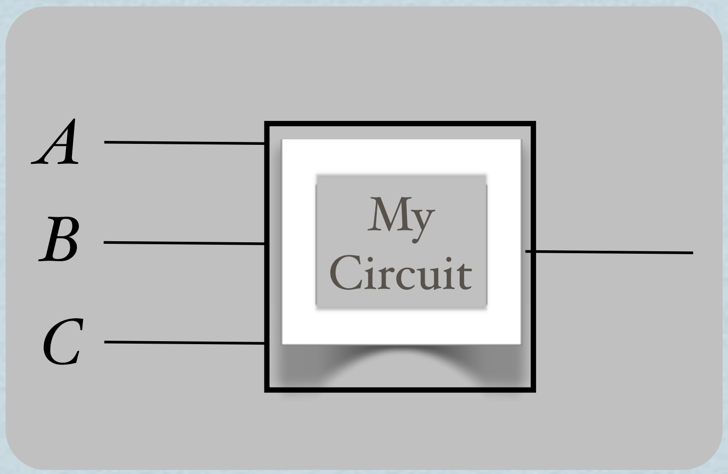
… and Truth Tables



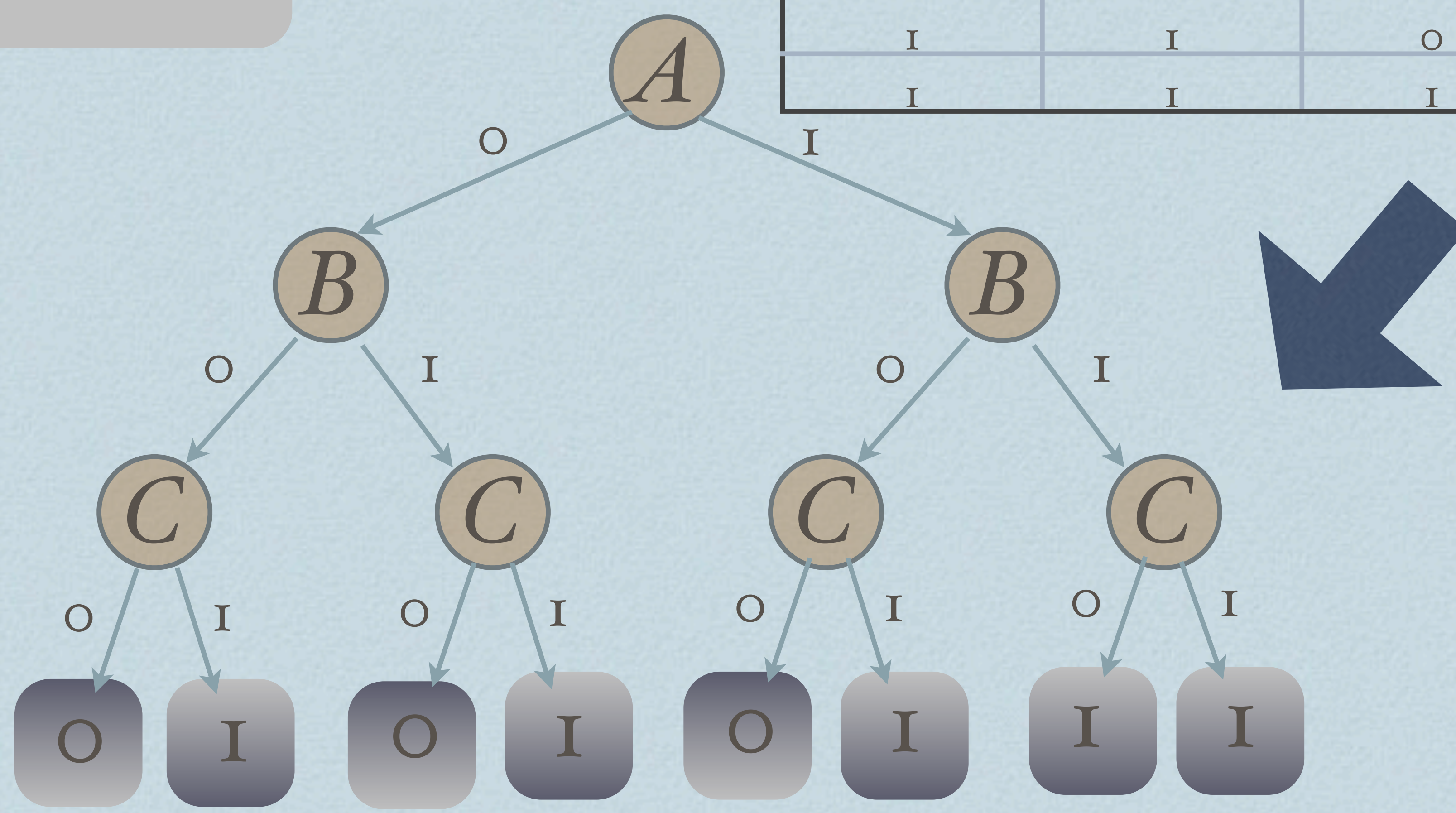


A	B	C	
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

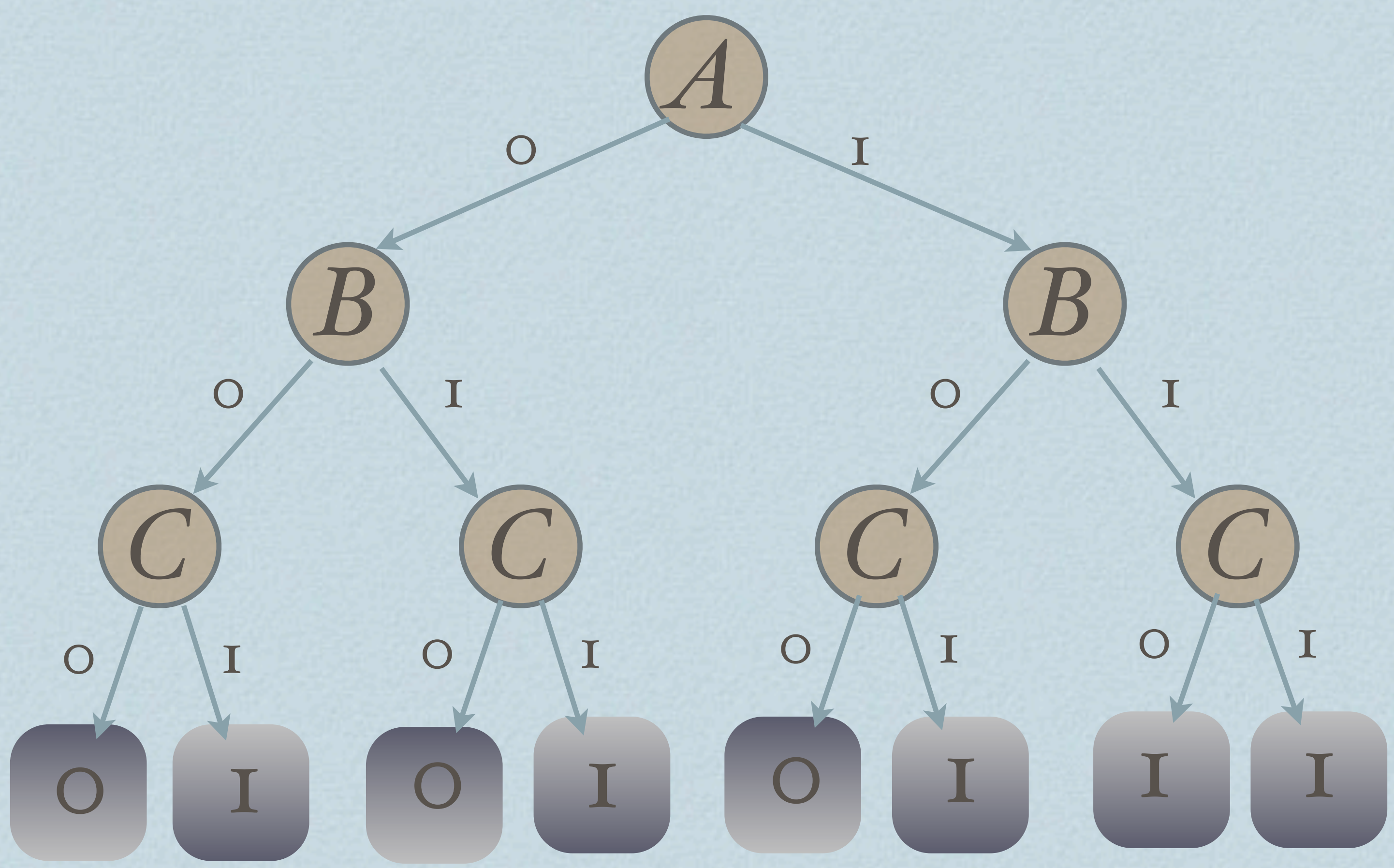
Decision Trees



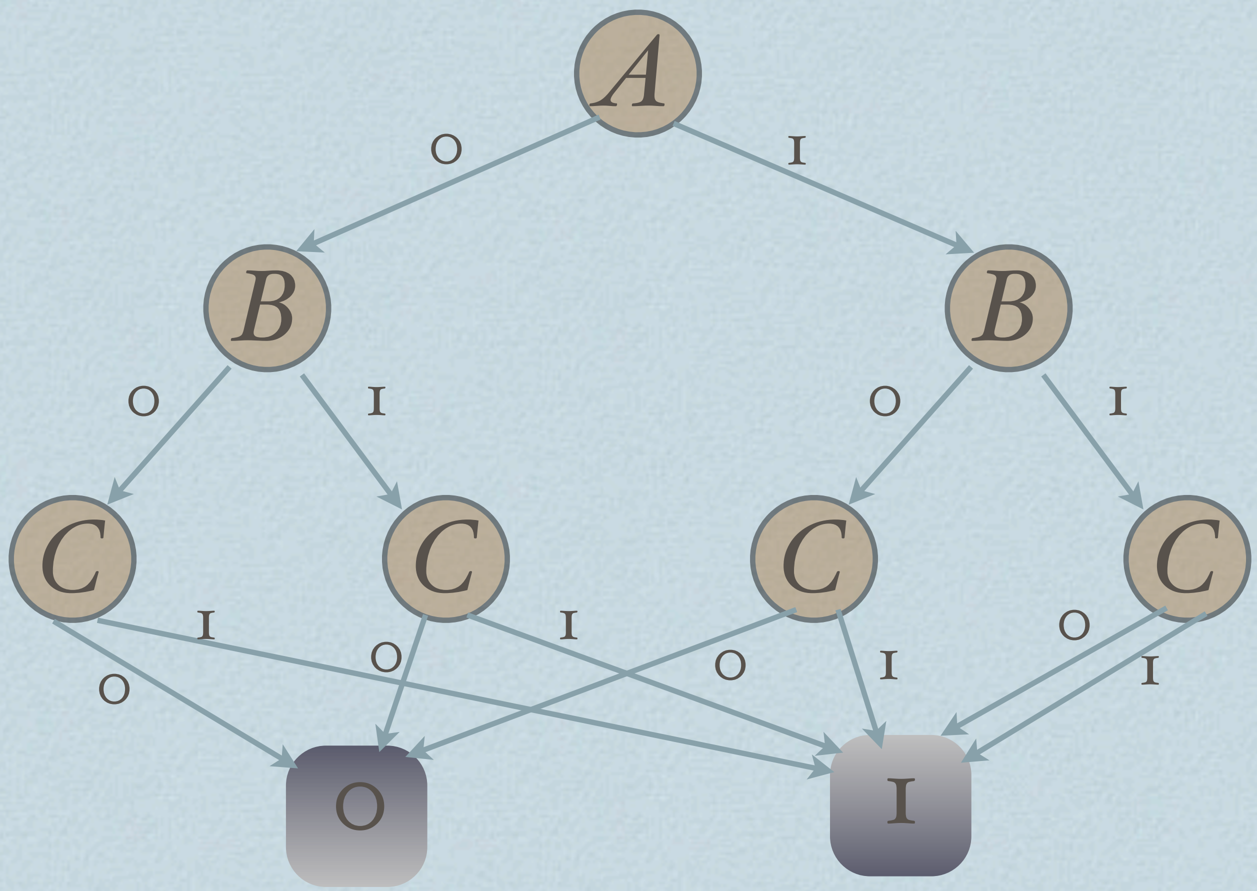
A	B	C	
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



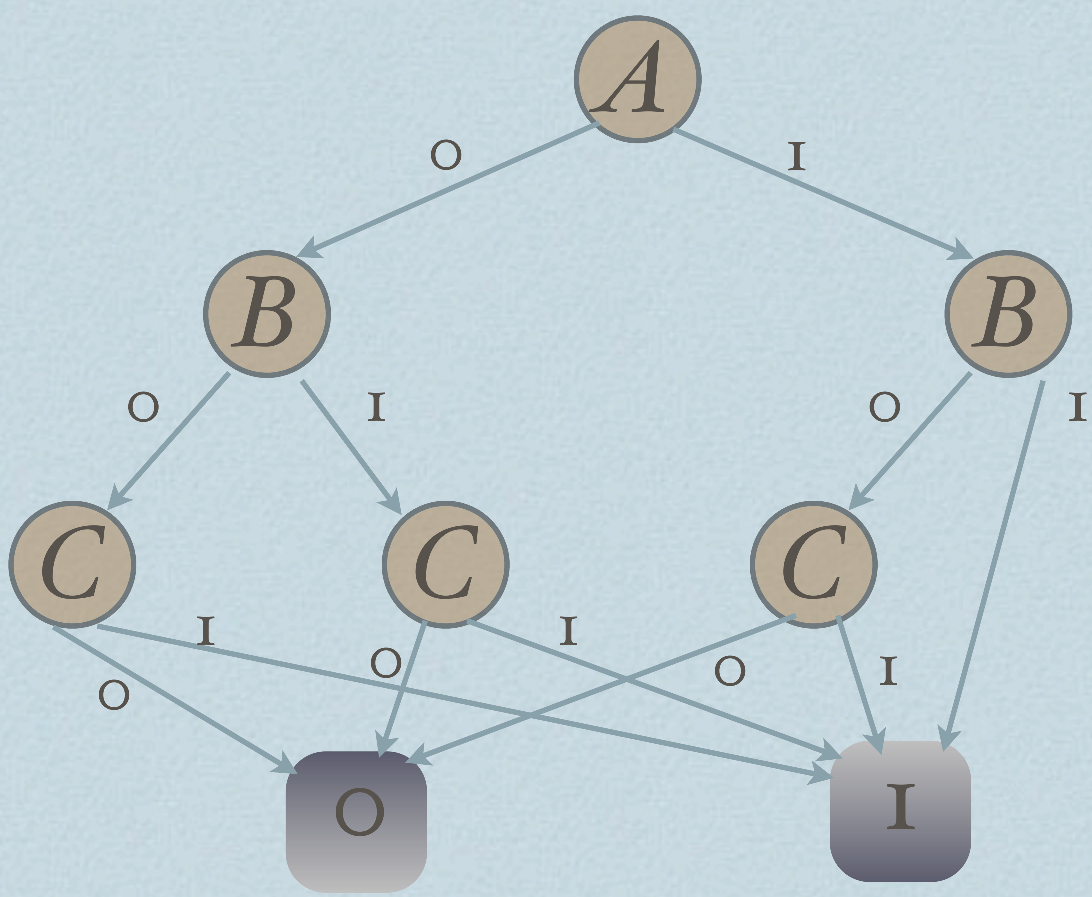
BDDs: Sharing



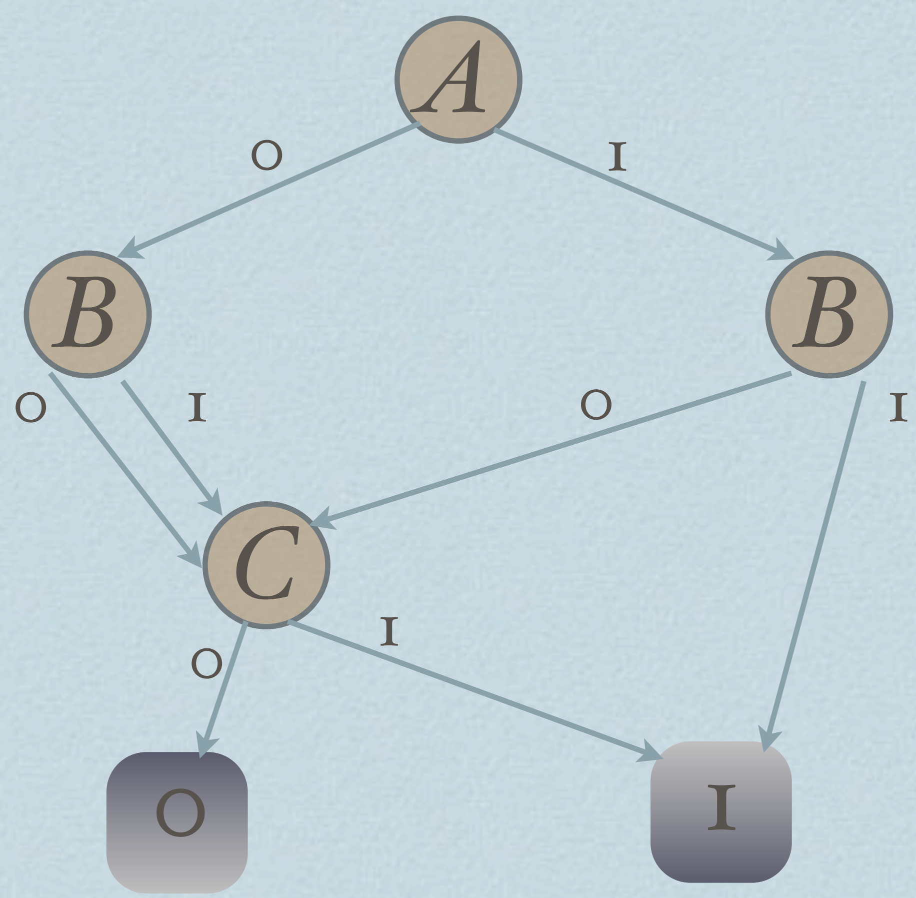
BDDs: Sharing



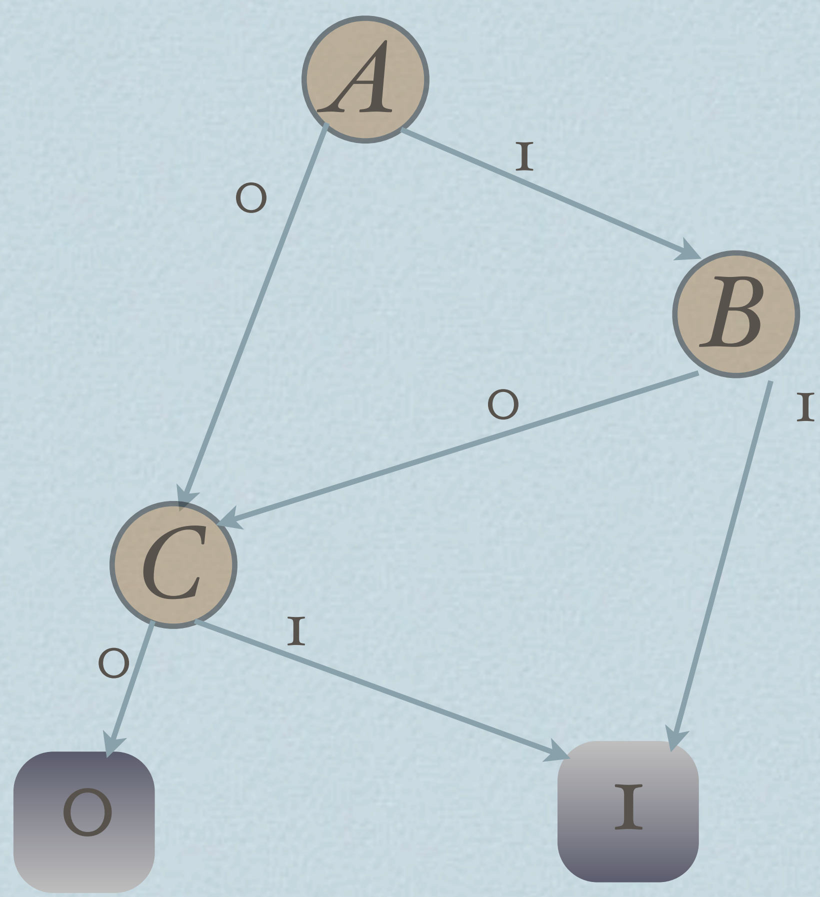
BDDs: Reduction



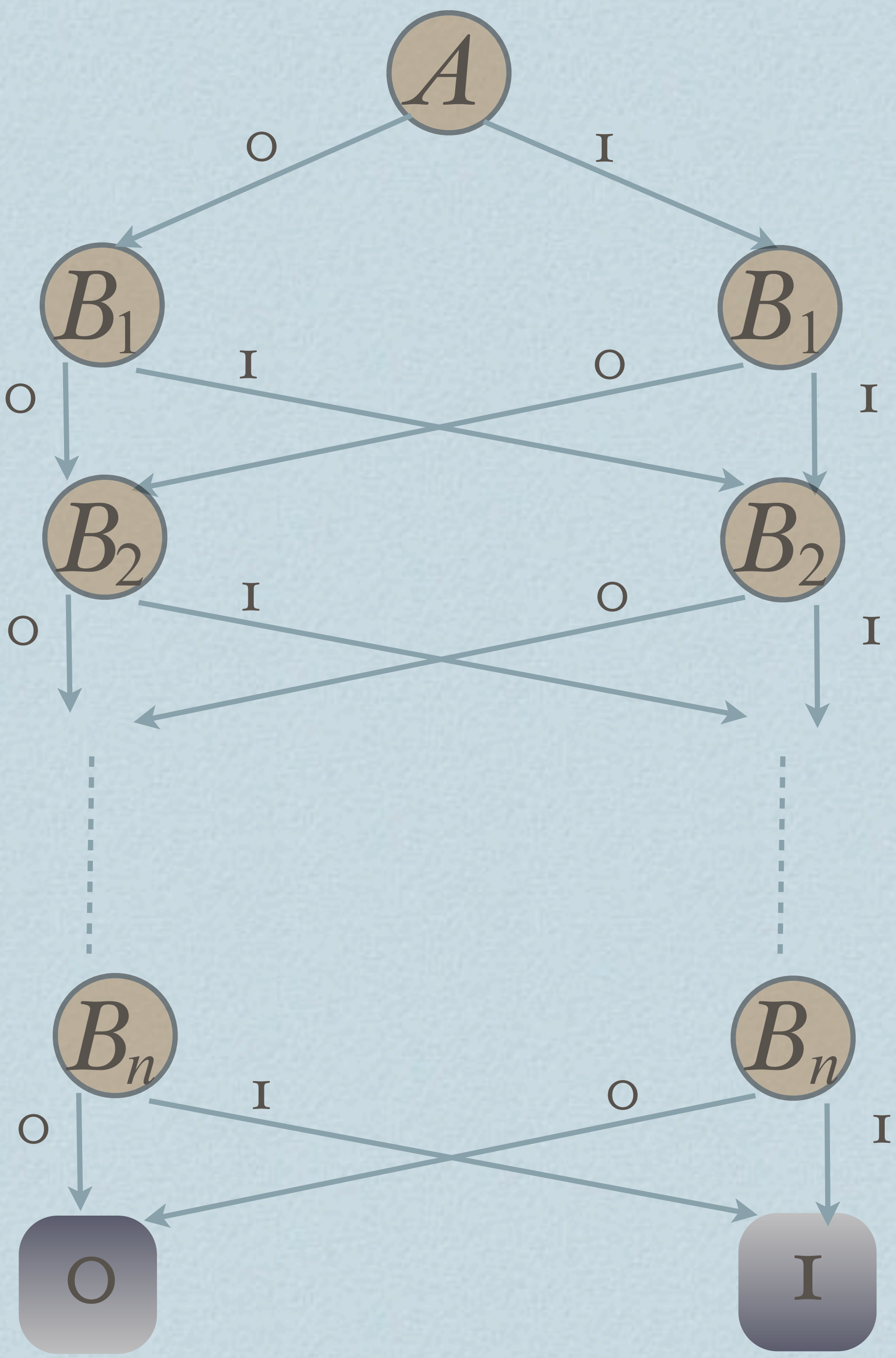
BDDs: Sharing



BDDs: Reduction

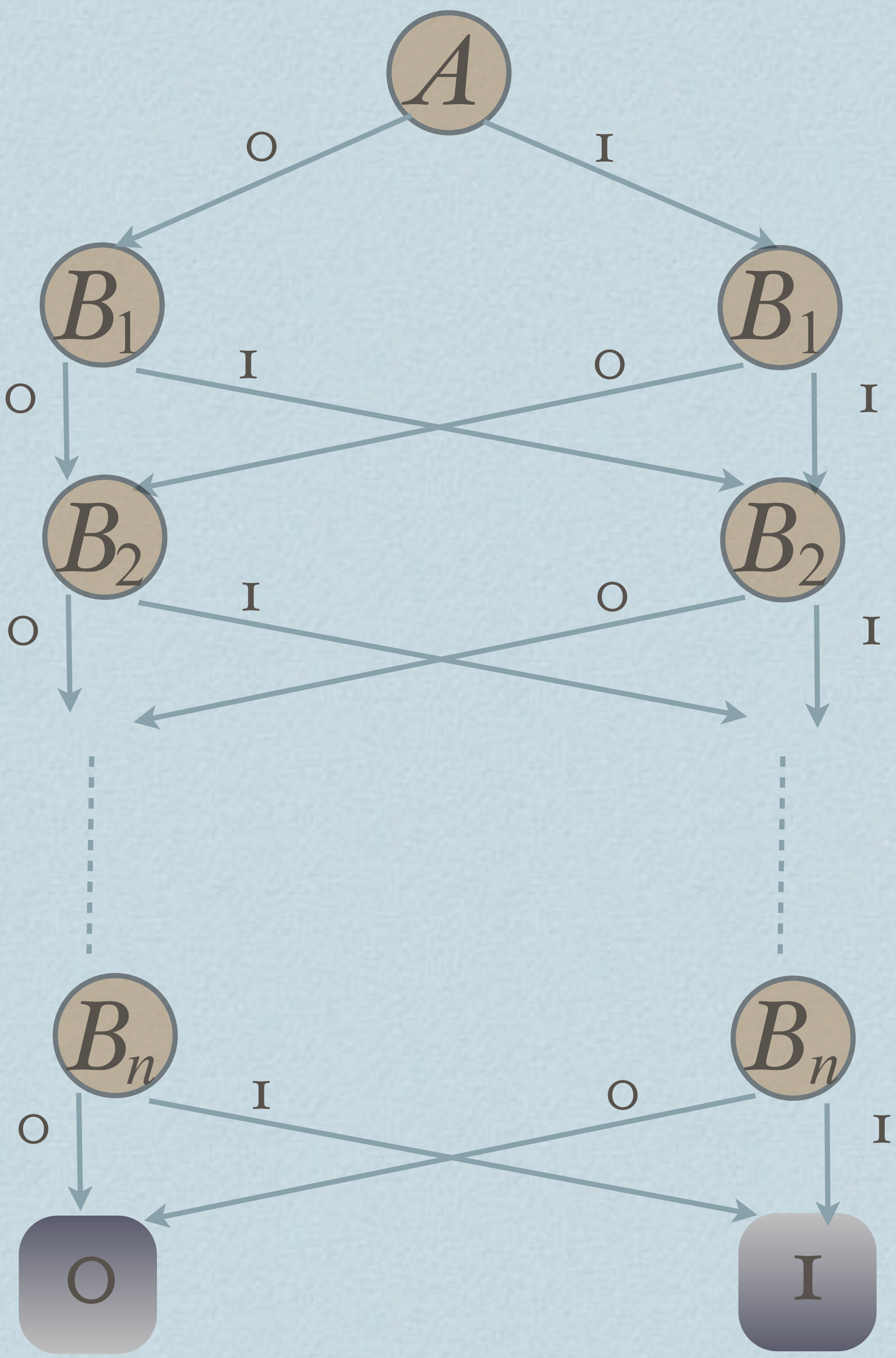


BDDs are Compact



- Space used: $2n+3$ nodes
- #configurations (paths): 2^{n+1}

BDDs are Compact



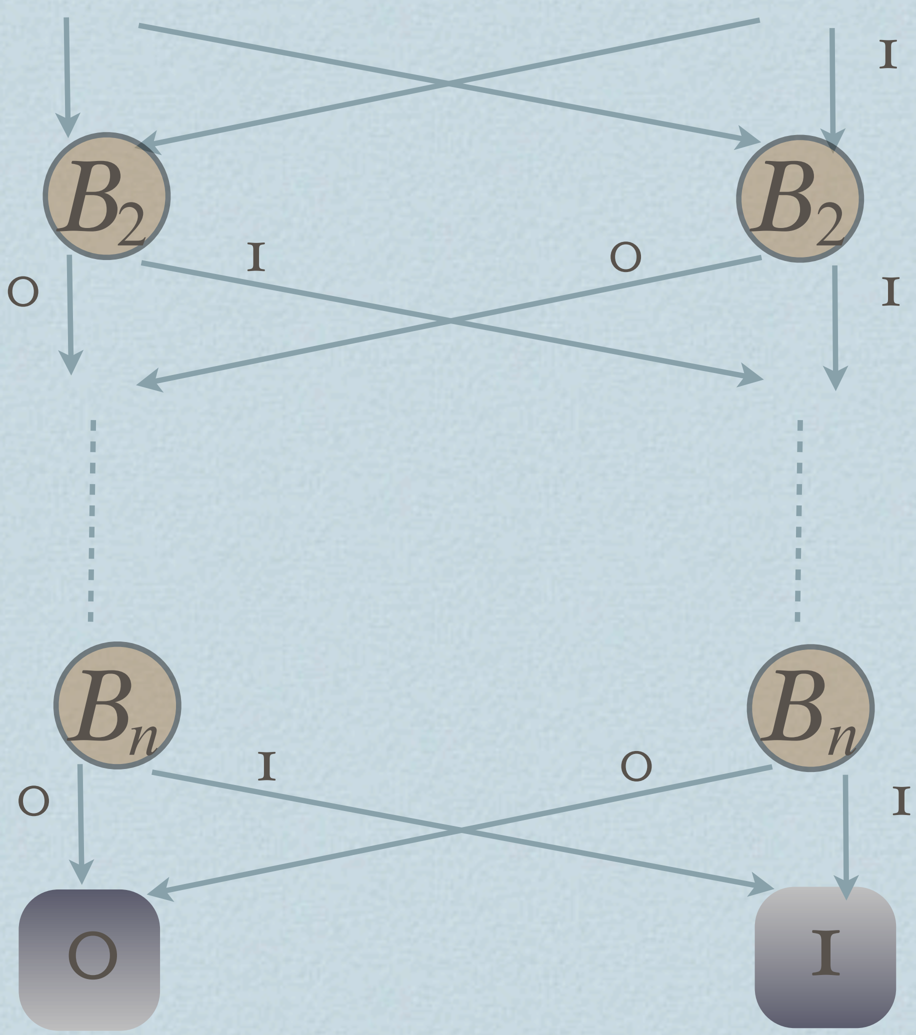
- Space used: $2n+3$ nodes
- #configurations (paths): 2^{n+1}

n	space used	# configs
50	103	2 10^{15}
100	203	2 10^{30}
150	303	2 10^{45}
200	403	2 10^{60}
250	503	2 10^{75}
300	603	2 10^{90}
350	703	2 10^{105}
400	803	2 10^{120}
450	903	2 10^{135}

* Each node typically takes 16 bytes

BDDs are Compact

- Space used: $2n+3$ nodes
- #configurations (paths): 2^{n+1}



n	space used	# configs
50	103	$2\ 10^{15}$
100	203	$2\ 10^{30}$
150	303	$2\ 10^{45}$
200	403	$2\ 10^{60}$
250	503	$2\ 10^{75}$
300	603	$2\ 10^{90}$
350	703	$2\ 10^{105}$
400	803	$2\ 10^{120}$
450	903	$2\ 10^{135}$

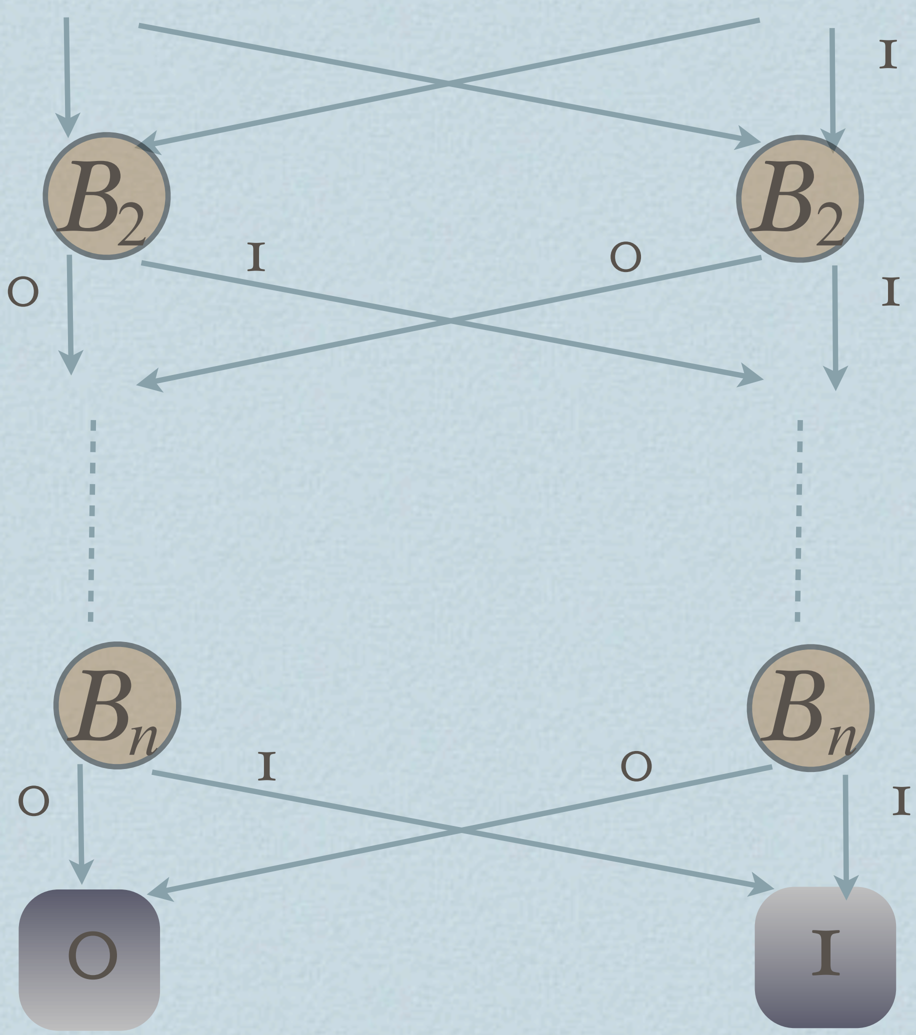
* Each node typically takes 16 bytes

BDDs are Compact

Smaller than your typical Word document

❖ Space used: $2n+3$ nodes

❖ #configurations (paths): 2^{n+1}



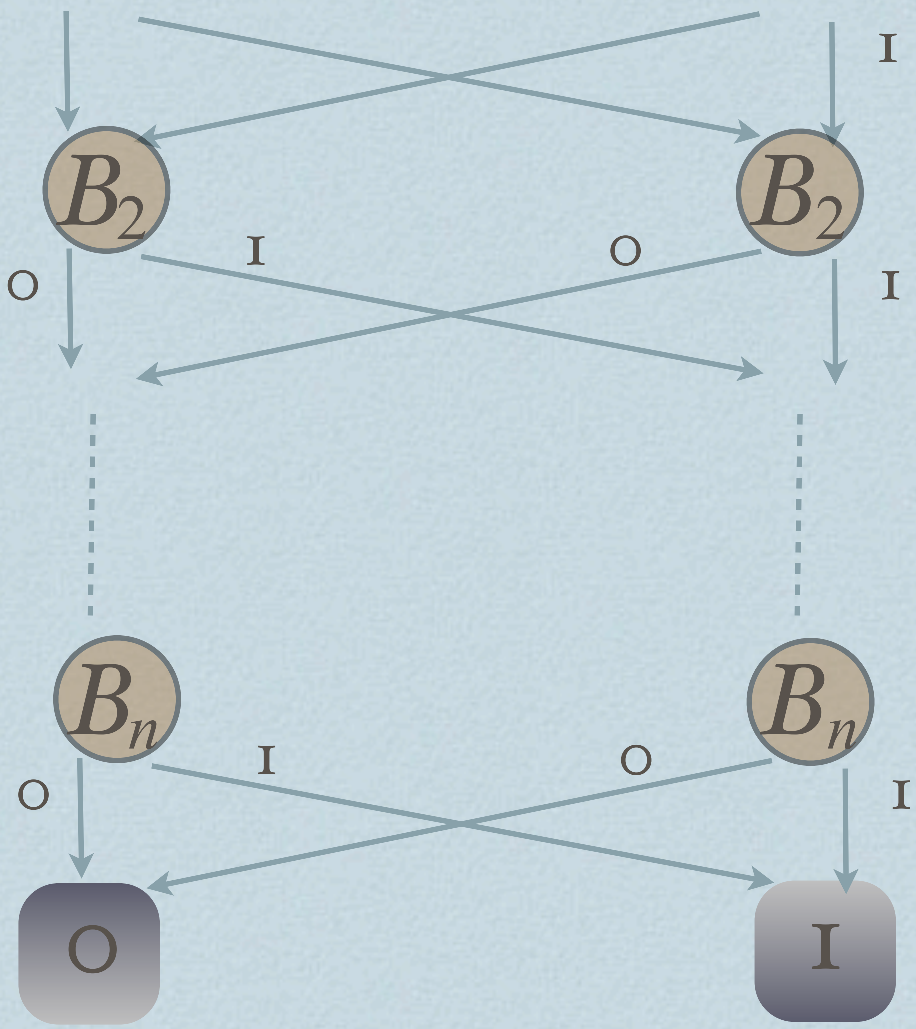
n	space used	# configs
50	103	$2\ 10^{15}$
100	203	$2\ 10^{30}$
150	303	$2\ 10^{45}$
200	403	$2\ 10^{60}$
250	503	$2\ 10^{75}$
300	603	$2\ 10^{90}$
350	703	$2\ 10^{105}$
400	803	$2\ 10^{120}$
450	903	$2\ 10^{135}$

* Each node typically takes 16 bytes

BDDs are Compact

❖ Space used: $2n+3$ nodes

❖ #configurations (paths): 2^{n+1}



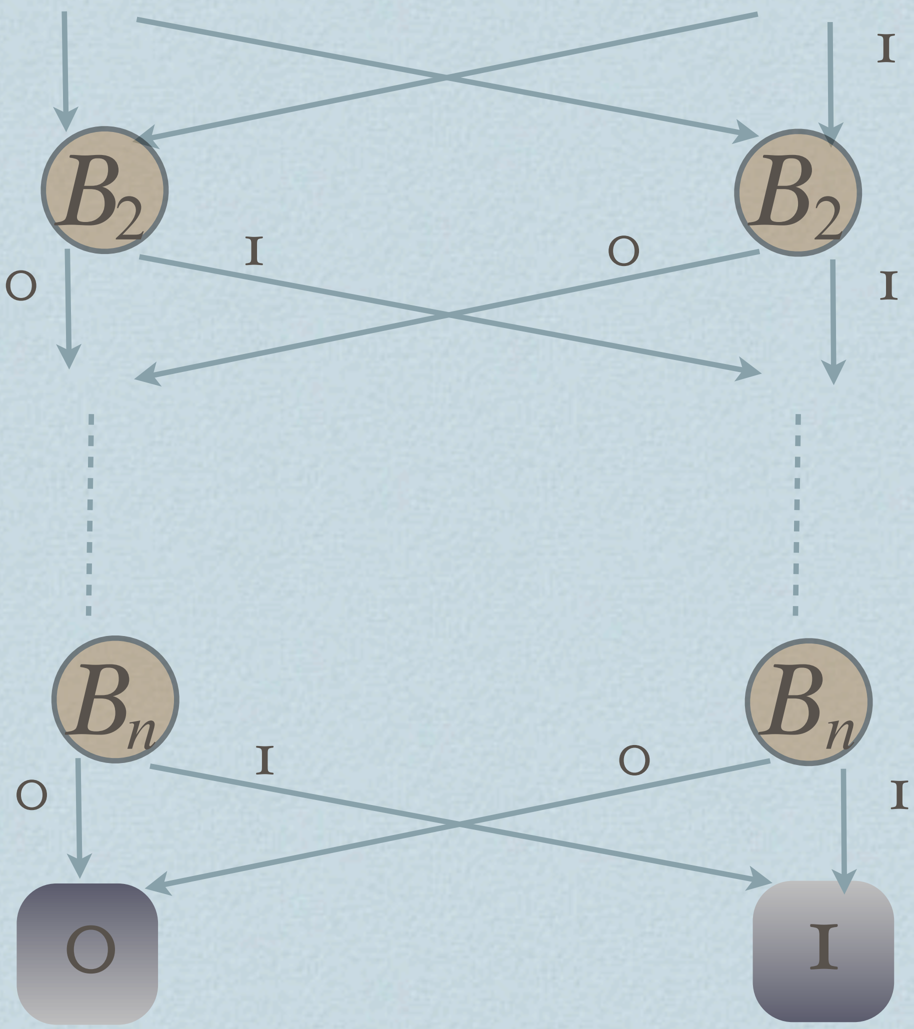
n	space used	# configs
50	103	$2\ 10^{15}$
100	203	$2\ 10^{30}$
150	303	$2\ 10^{45}$
200	403	$2\ 10^{60}$
250	503	$2\ 10^{75}$
300	603	$2\ 10^{90}$
350	703	$2\ 10^{105}$
400	803	$2\ 10^{120}$
450	903	$2\ 10^{135}$

* Each node typically takes 16 bytes

BDDs are Compact

Larger than your computer's memory

- Space used: $2n+3$ nodes
- #configurations (paths): 2^{n+1}



n	space used	# configs
50	103	$2\ 10^{15}$
100	203	$2\ 10^{30}$
150	303	$2\ 10^{45}$
200	403	$2\ 10^{60}$
250	503	$2\ 10^{75}$
300	603	$2\ 10^{90}$
350	703	$2\ 10^{105}$
400	803	$2\ 10^{120}$
450	903	$2\ 10^{135}$

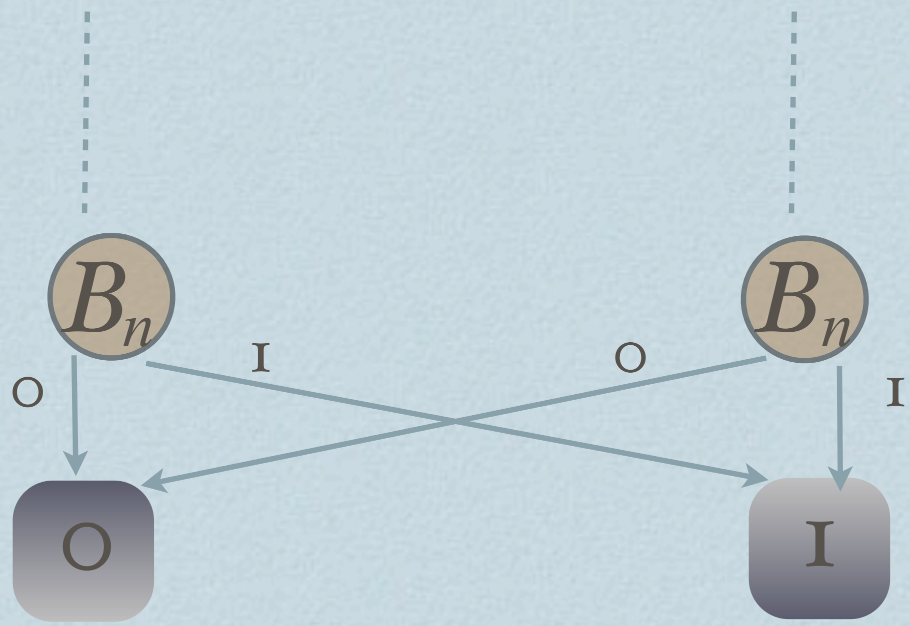
* Each node typically takes 16 bytes

BDDs are Compact

Larger than your computer's memory

- Space used: $2n+3$ nodes
- #configurations (paths): 2^{n+1}

n	space used	# configs
50	103	$2\ 10^{15}$
100	203	$2\ 10^{30}$
150	303	$2\ 10^{45}$
200	403	$2\ 10^{60}$
250	503	$2\ 10^{75}$
300	603	$2\ 10^{90}$
350	703	$2\ 10^{105}$
400	803	$2\ 10^{120}$
450	903	$2\ 10^{135}$

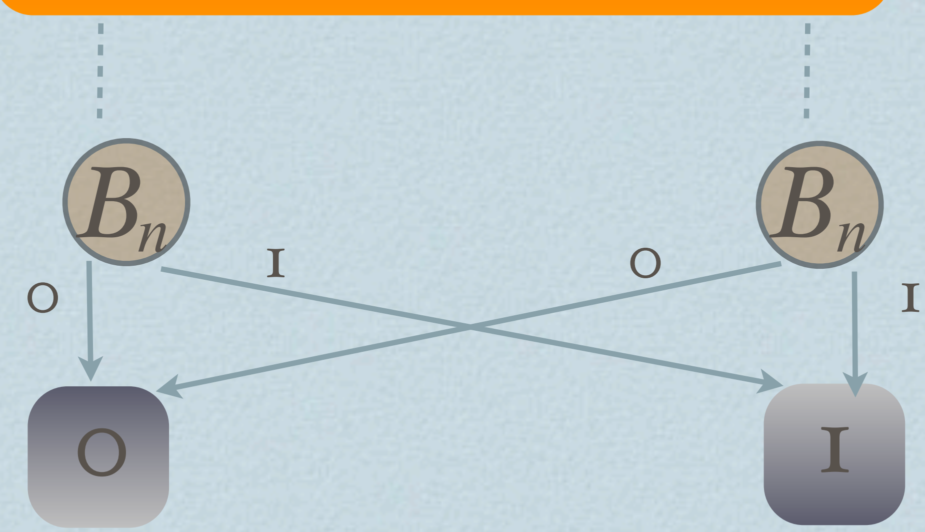


* Each node typically takes 16 bytes

BDDs are Compact

Larger than your computer's memory

Larger than the universe



- Space used: $2n+3$ nodes
- #configurations (paths): 2^{n+1}

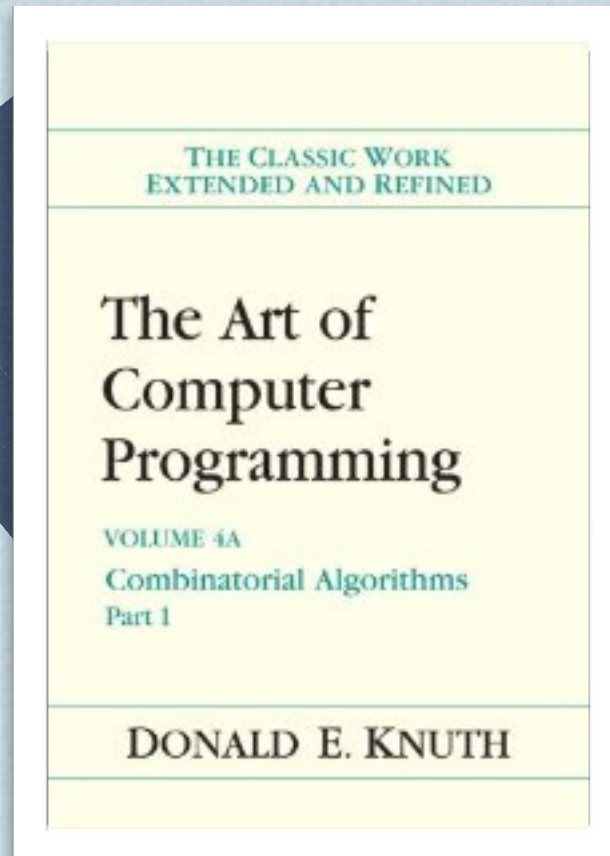
n	space used	# configs
50	103	$2\ 10^{15}$
100	203	$2\ 10^{30}$
150	303	$2\ 10^{45}$
200	403	$2\ 10^{60}$
250	503	$2\ 10^{75}$
300	603	$2\ 10^{90}$
350	703	$2\ 10^{105}$
400	803	$2\ 10^{120}$
450	903	$2\ 10^{135}$

* Each node typically takes 16 bytes

BDDs Today

- Circuit verification
- Model-checking: verification, beyond circuits
- Circuit design
- Fault diagnosis
- Production configuration
- Etc.

Want to Know More?



Volume 4A – Combinatorial Algorithms, Part 1

- Chapter 7 – Combinatorial Searching
  - 7.1. Zeros and Ones
    - 7.1.1. Boolean Basics
    - 7.1.2. Boolean Evaluation
    - 7.1.3. Bitwise Tricks and Techniques
    - 7.1.4. Binary Decision Diagrams
  - 7.2. Generating All Possibilities
    - 7.2.1. Generating Basic Combinatorial Patterns
      - 7.2.1.1. Generating all n-tuples
      - 7.2.1.2. Generating all permutations
      - 7.2.1.3. Generating all combinations
      - 7.2.1.4. Generating all partitions
      - 7.2.1.5. Generating all set partitions
      - 7.2.1.6. Generating all trees
      - 7.2.1.7. History and further references

Quick reference: http://en.wikipedia.org/wiki/Binary_decision_diagram

References

- **Sh. B. Akers**, "Binary decision diagrams," IEEE Transactions on Computers, Vol. C-27, No. 6 (June, 1978), pp. 509-516
- **R.E. Bryant**,"Graph-based algorithms for Boolean function manipulation," *IEEE Transactions on Computers*, Vol. C-35, No. 8 (August, 1986), pp. 677–691
- J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, L.J. Hwang, "Symbolic model-checking: 10^{20} states and beyond," *Information and Computation*, 98 (1992), pp. 142-170