Decision Diagrams in Combinatorics

Andrew Conway

Conference in honor of Tony Guttmann's 80th birthday

Talk Contents

- What is a binary decision diagram (BDD)?
  - How is it useful in combinatorics?
  - How about a other variants?
- Use in pattern avoiding permutations
  - Generalization to multiset – MBDD, MZDD, etc.
  - Paper with Tony:

    “Counting Occurrences of Patterns in Permutations”

    January 2025 The Electronic Journal of Combinatorics 32(1)

    DOI:10.37236/12963

What is a decision diagram?

- A data structure
- Represents a binary function
  - e.g. f(a,b,c) = (a xor b) and (not c)

The Art of Computer Programming, Volume 4, Fascicle 1,

Bitwise Tricks & Techniques; Binary Decision Diagrams

Donald Knuth

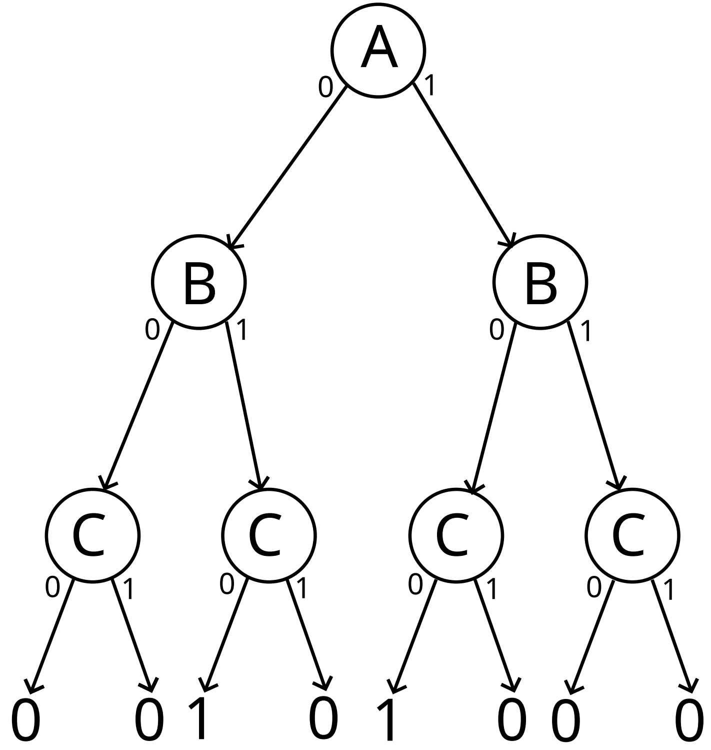
Combinatorics motivation

- A binary function can be considered to define a set S = {x:f(x)=true}
- e.g. variables are bonds on a fixed size lattice, function is true if they form a self avoiding walk.
- Number of true elements = number of self avoiding walks.

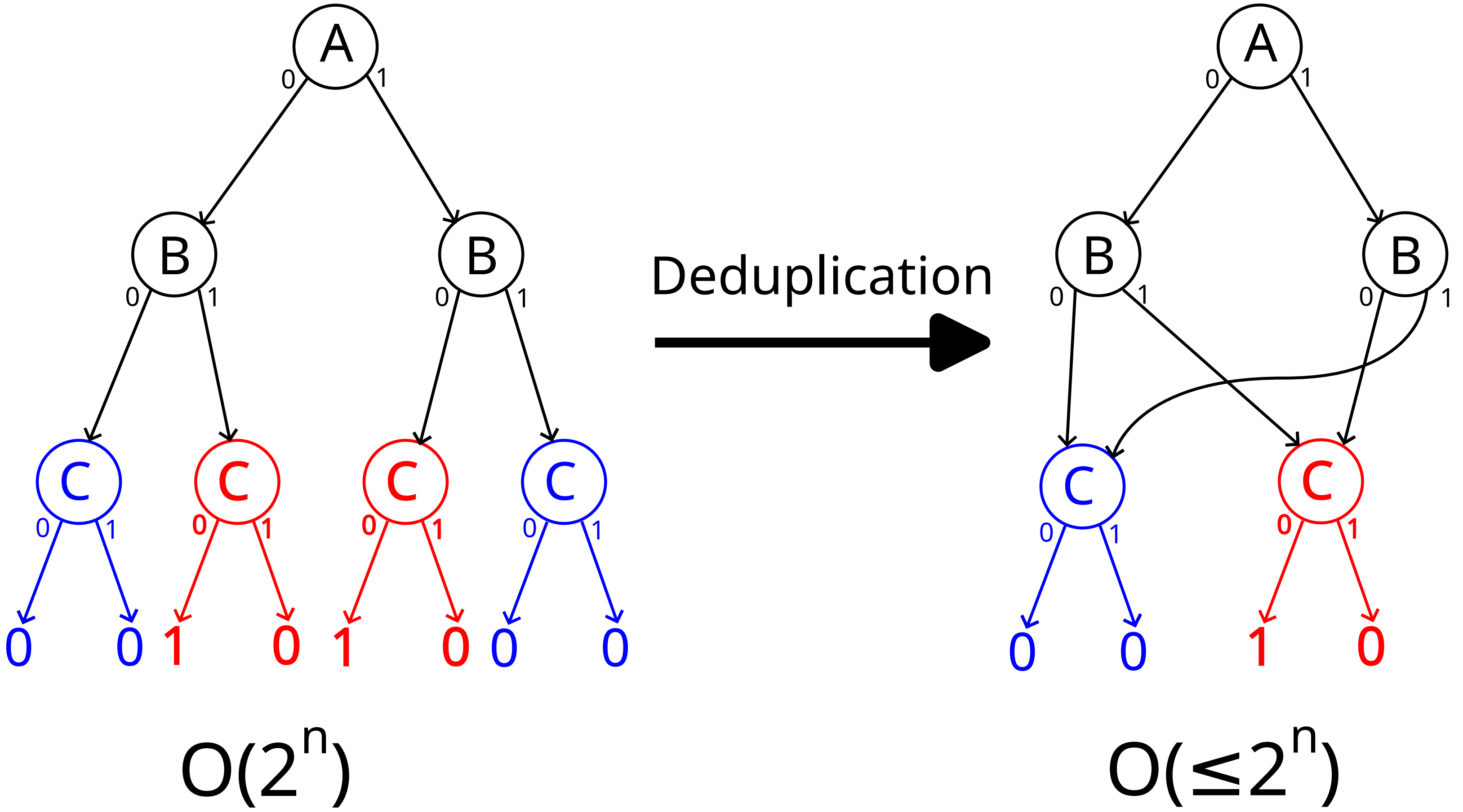
Truth Table

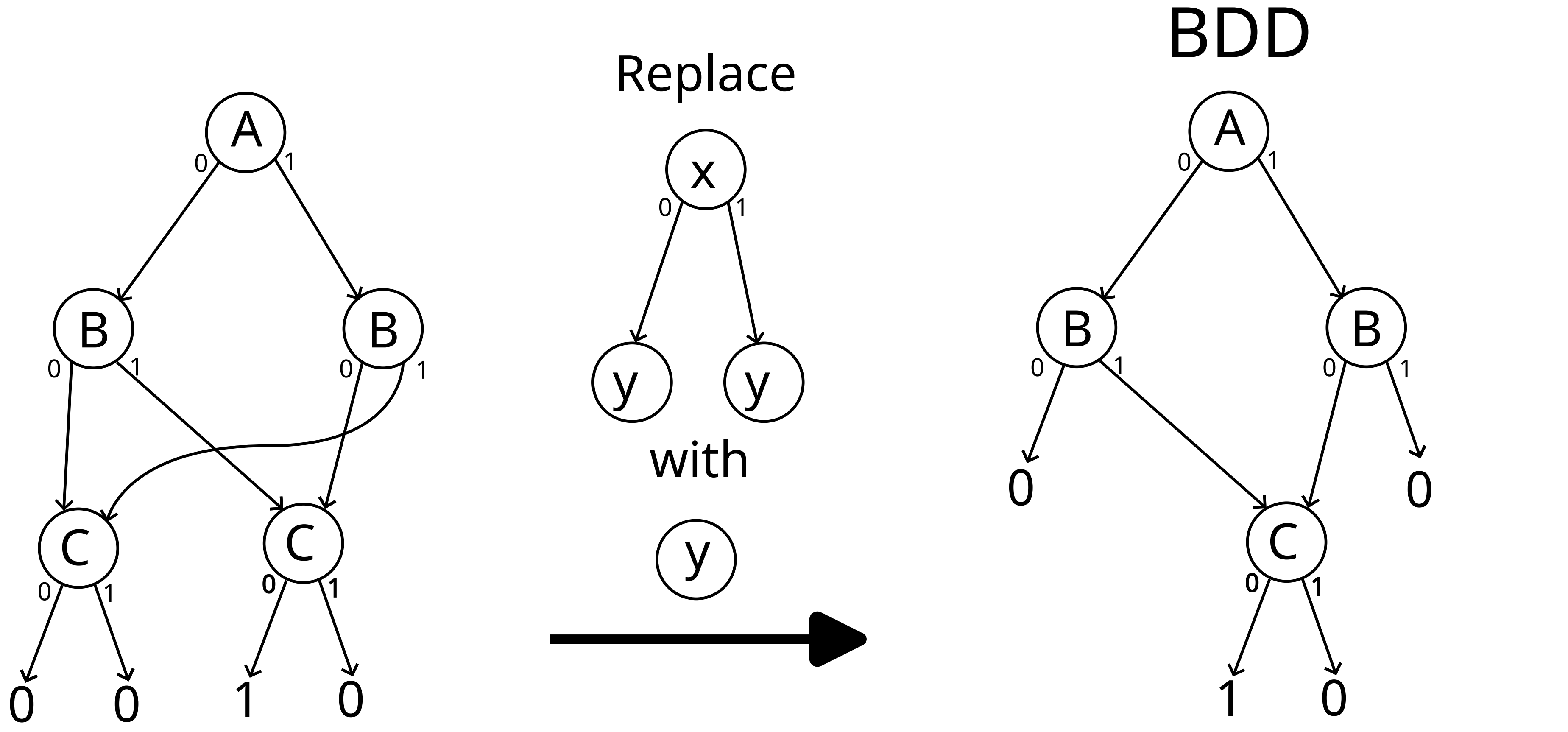
A	B	C	f(A,B,C)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

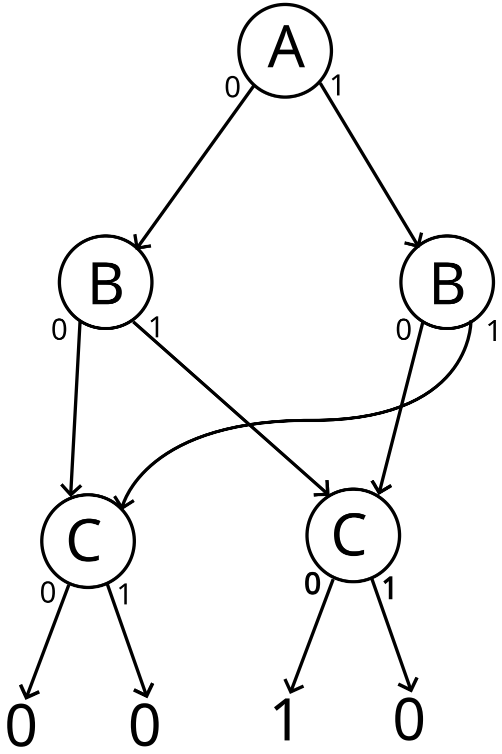
Decision Diagram



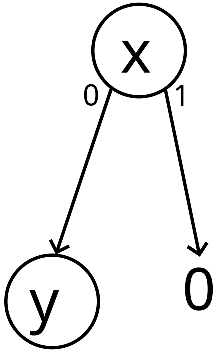
MOST IMPORTANT SLIDE OF THE TALK



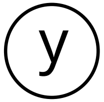

Replace
with
BDD



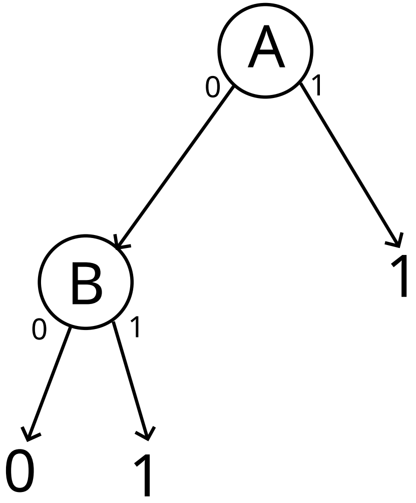
Replace



with



ZDD



Not what it naively looks like!

What can you do with an xDD?

- Often store a complex function of n variables in much less space than $O(2^n)$.
- Logical operations (and or not) of sizes s and t take worst case O(st) and often much better
- Fast to count, find smallest number of satisfying elements, get an arbitrary solution

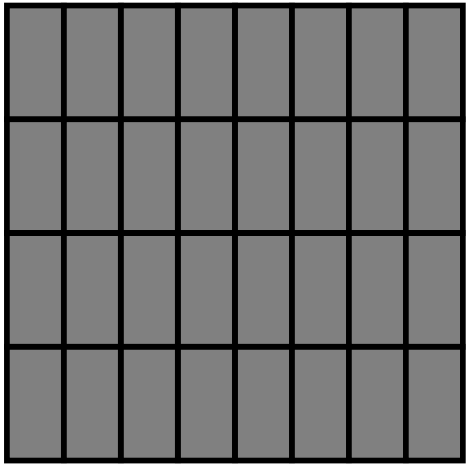
Application : SAWs

- Self avoiding walk on a finite rectangular lattice
- Each bond is a variable
- Function is true if it is a SAW
- Same algorithm as to count them, just with a more complex answer than an integer.

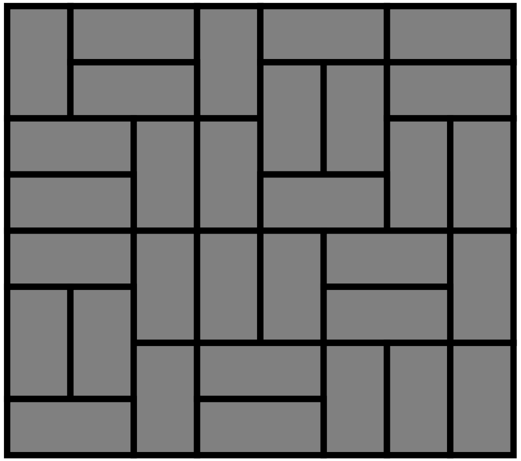
Application : Tilings

- Knuth suggest the problem of tiling a chessboard with dominoes – how many ways?

1 of 12988816



5134724 of 12988816



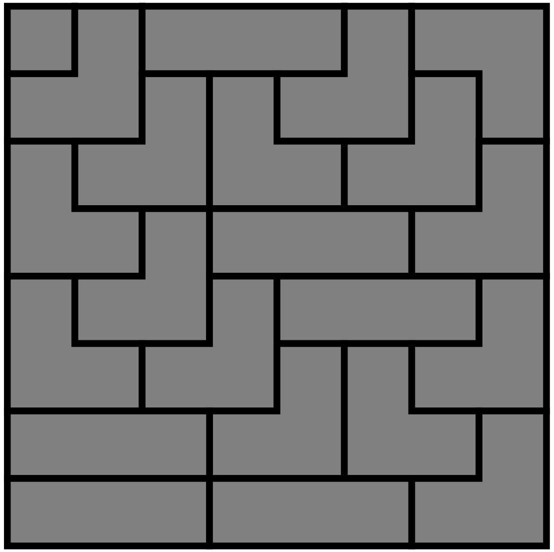
Speed

- BDD: 2.1 ms 5679 nodes (11712 to generate)
  - 41 lines of code including comments, not counting xDD library
- ZDD: 3.2 ms 2298 nodes (19790 to generate)
- Simple dynamic programming approach: 6 μs
  - 27 lines of code including comments

Program: https://github.com/AndrewConway/xdd

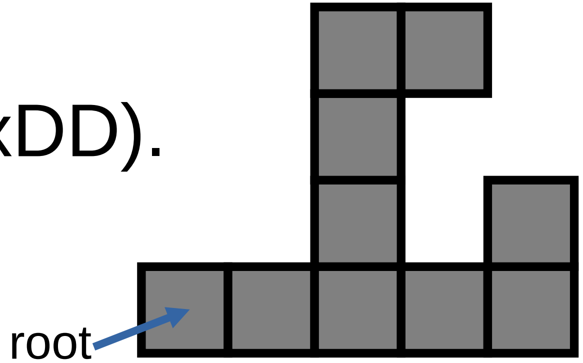
Bigger problem

- Tiling with unimoes, dominoes, trionimoes:
- 92109458286284989468604
- 243057372832 fewest tiles
- BDD: 1.4s, 2,447,347 nodes
- ZDD: 1.7s,   512,225 nodes



Application : Directed animals

- Polyonimoes on a square lattice.
  - Each occupied site other than the root has to have an element directly below or to the left
  - One variable for each site on a sufficiently large lattice.
  - All constraints are local (suits xDD).



Performance

- 20 terms. 86,574,831 animals
- BDD : 7.8s
  - 2,191,868 nodes 23,292,361 used
- ZDD : 6.2s
  - 1,292,942 nodes 13,952,387 used
- Dynamic programming :228µs
  - 5022 cache elements

Not looking great for xDDs.

- Simple program much faster for counting.
- xDDs make it easier to make pictures. Possibly more straight forward?

Pattern Avoiding Permutations

- Let p be a permutation of 1..n
- Let P be a permutation of 1..m $m \leq n$
- p ***contains*** the pattern P if a subset of p of length m has the same relative ordering as P.
  - Otherwise p ***avoids*** the pattern P.
- Examples:
  - 1,5,3,2,4 contains the pattern 1,3,2 as the subsequence 1,5,3 is in the same relative order as 1,3,2. (also 1,5,2 and 1,5,4 and 1,3,2)
  - 1,2,3,4,5 avoids the pattern 1,3,2 (also avoids 2,1)

Enumeration of pattern avoiding permutations : length 1 to 4

- All but one class solved analytically.
- Dedicated algorithms for enumeration of the 1324 class.

Enumeration of PAPs : length 5

- Too many hard patterns for specialized algorithms
- Best general algorithms explicitly generated all solutions, so slow

Permutation Decision Diagrams

- Minato had a novel idea for representing a set of permutations
  - Represent permutations by a set of binary variables
  - Make a new operation on such ZDDs to compose permutations, maintaining canonicity.
  - These are called πDDs.
- Inoue improved the encoding, called Rot-πDDs.

Use of πDDs in PAPs

- Minato defined a set of operations that created a set of all pattern containing permutations.
- Then subtract from n! to get PAPs.
- Much faster than direct enumeration.
- Inoue’s variation is better still.

Pattern containing permutations

- Next step : How many permutations contain a pattern exactly 1 time? Or *r* times?
- For simple patterns, there are some analytic solutions.
- More generally…?

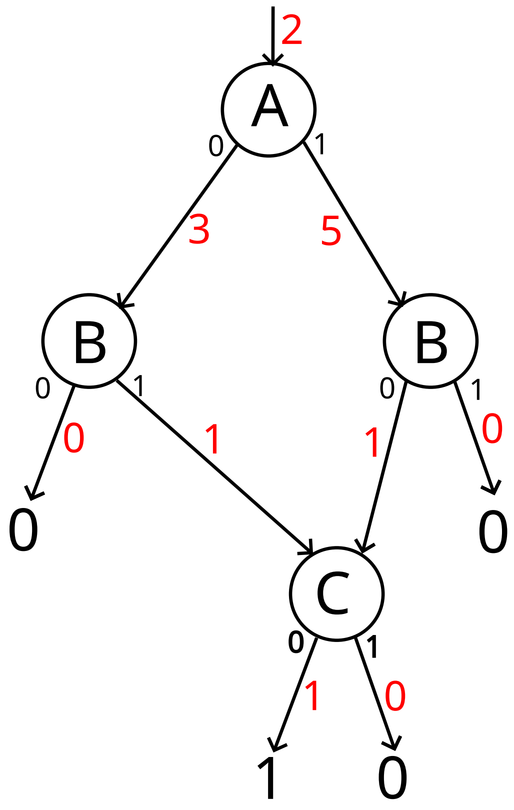
Minato’s algorithm and multisets

- Take Minato’s algorithm for generating all pattern containing permutations
- Use multisets instead of sets.
  - Where $\{a,b,b,c\}\times\{b,b,c\} = \{b,b,b,b,c\}$
- Now the multiplicity of each element is the number of times it contains the pattern.

Multisets and decision diagrams

- Decision diagrams can be extended to represent multisets.
  - f(booleans) $\rightarrow \mathbb{N}$
- Modification: instead of a reference to a node, have an integer multiple and a reference to a node.

MBDD



A	B	C	f(A,B,C)
0	0	0	0
0	0	1	0
0	1	0	2x3=6
0	1	1	0
1	0	0	2x5=10
1	0	1	0
1	1	0	0
1	1	1	0

Canonicalization

- Need to have a unique representation for each multiset.
  - A multiple of 0 iff the reference is to 0.
  - Each node has two references. Their multiples should be coprime (or 1 if the other is 0).
- Details and theory in section 2 of:

“Counting Occurrences of Patterns in Permutations”

January 2025 The Electronic Journal of Combinatorics 32(1)
DOI:10.37236/12963

Implementation

- Rust library at https://github.com/AndrewConway/xdd
- Can enumerate by multiplicity efficiently.
- Multiset BDD is an MBDD.
  - Similarly MZDD, MπDD,Rot-MπDD
- Program used in paper is in that as an example.

References

- Theory and results

  “Counting Occurrences of Patterns in Permutations”

  January 2025 The Electronic Journal of Combinatorics 32(1)

  DOI:10.37236/12963

- Library and program
  - https://github.com/AndrewConway/xdd
- Program to make pictures & benchmarks in this talk
  - https://github.com/AndrewConway/xdd_pictures