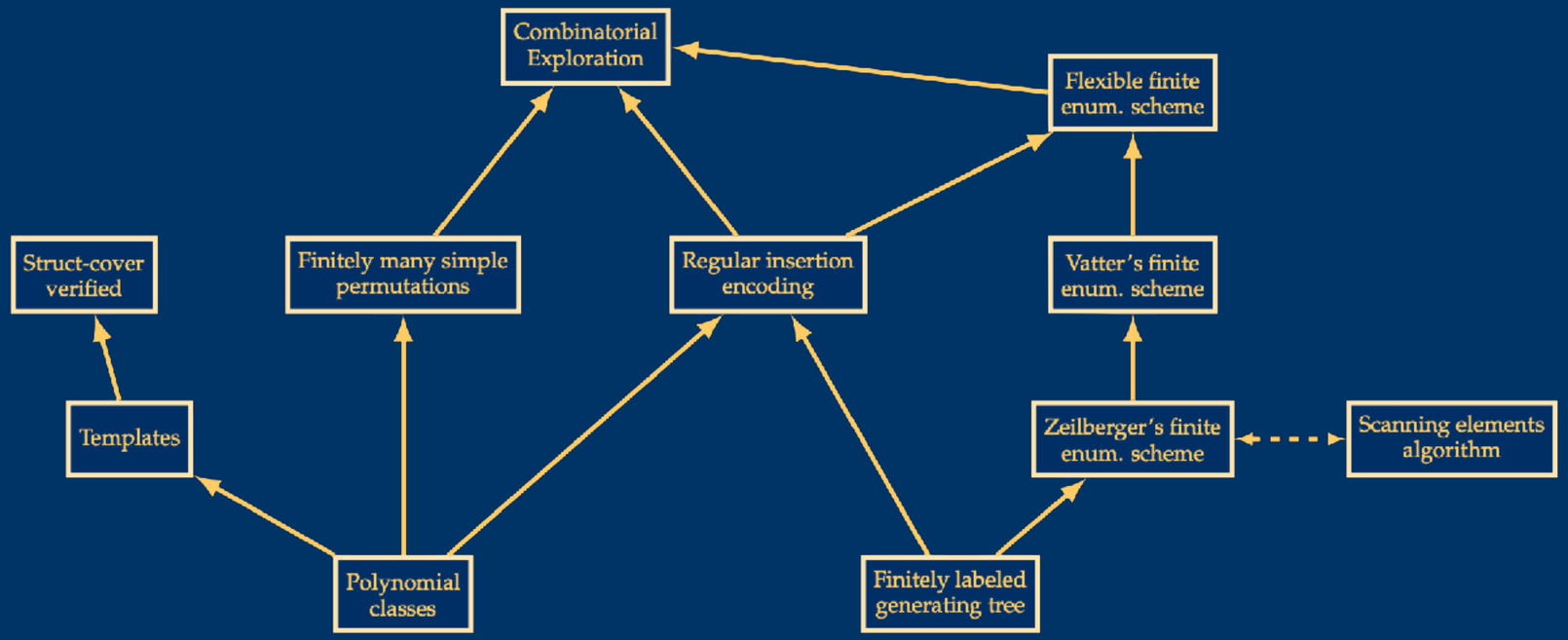
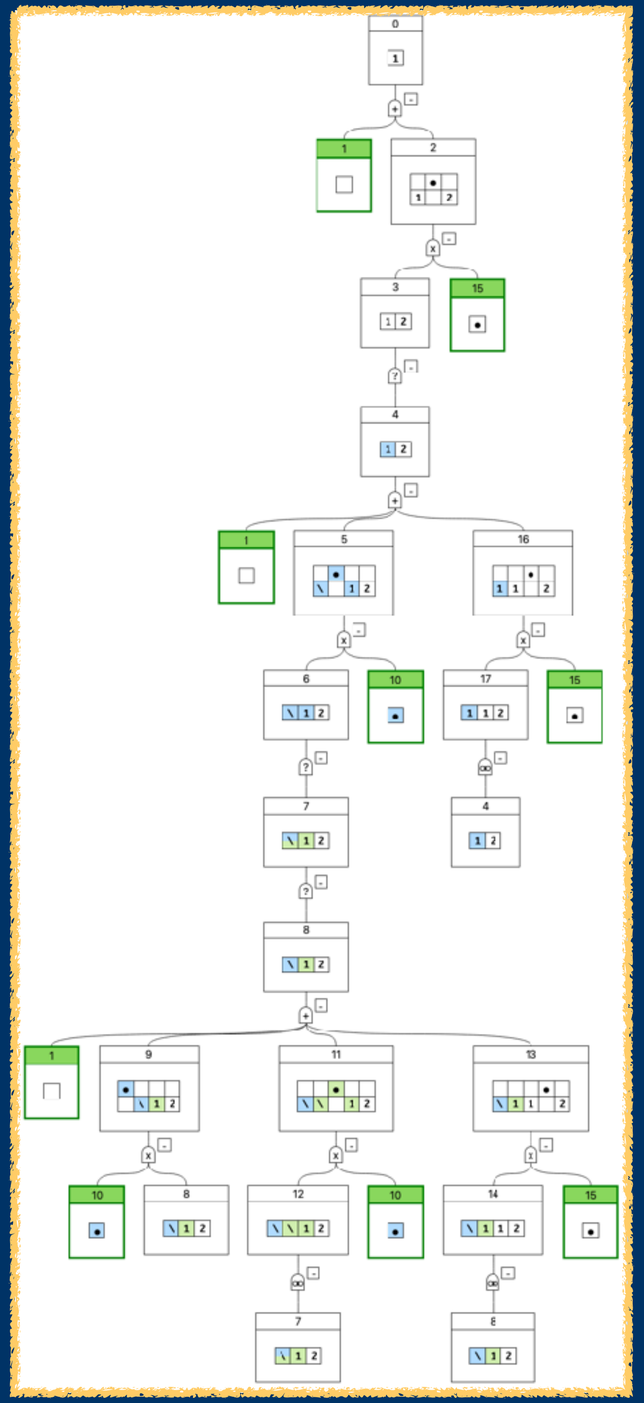
Computational and Experimental Methods in Permutation Patterns

Jay Pantone
Marquette University

Rutgers Experimental Mathematics Seminar





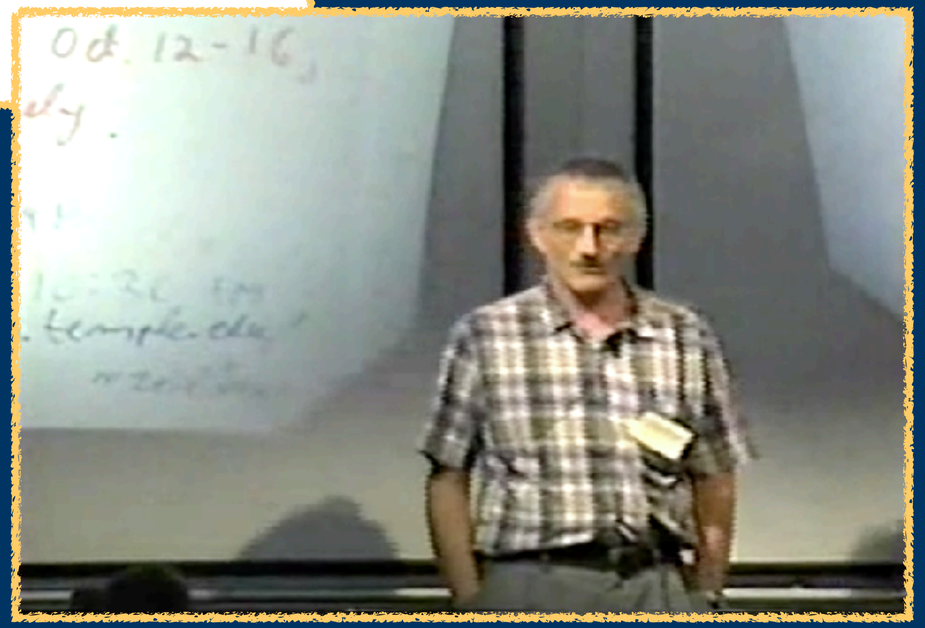
28 years ago...

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998



28 years ago...

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

It is way too soon to teach our computers how to become full-fledged *humans*. It is even premature to teach them how to become *mathematicians*; it is even unwise, at present, to teach them how to become *combinatorialists*. But the time is ripe to teach them how to become experts in a suitably defined and narrowly focused subarea of combinatorics. In this article, the author will describe his efforts in teaching his beloved computer, Shalosh B. Ekhad, how to be an enumerator of Wilf classes.

28 years ago...

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

With all due respect to Wilf classes and enumeration, and even to combinatorics, the main point of this article is not to enhance our understanding of Wilf classes, but to *illustrate* how much (if not all) of mathematical research will be conducted in a few years. It goes as follows. Suppose a (as of now, human) mathematician has a brilliant idea. Teach that idea to a computer and let the computer 'do research' using that idea.

28 years ago…

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

With all due respect to Wilf classes and enumeration, and even to combinatorics, the main point of this article is not to enhance our understanding of Wilf classes, but to *illustrate* how much (if not all) of mathematical research will be conducted in a few years. It goes as follows. Suppose a (as of now, human) mathematician has a brilliant idea. Teach that idea to a computer and let the computer 'do research' using that idea.

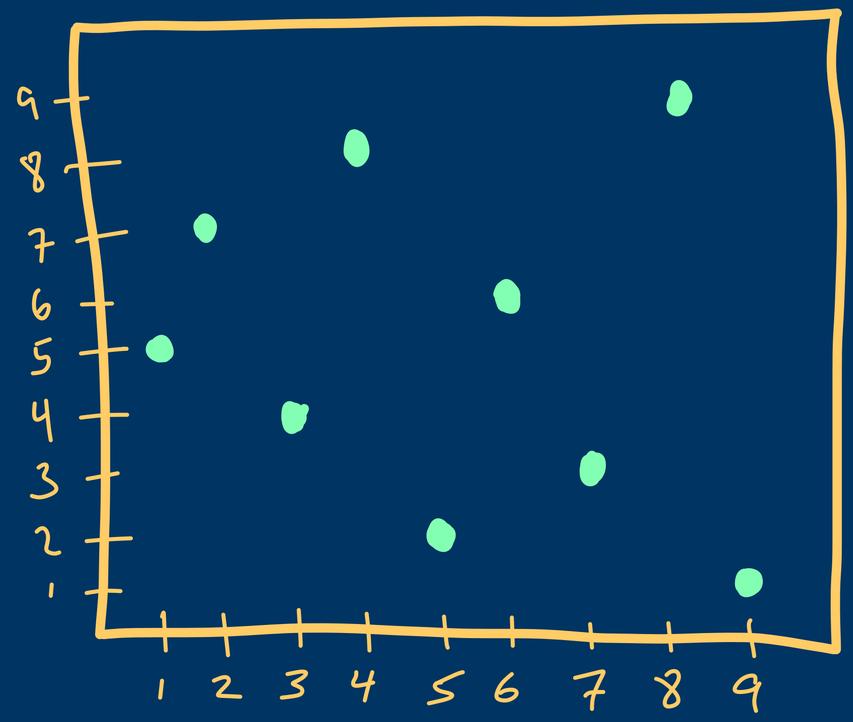
What's the state of computation and experimentation in Permutation Patterns?

Permutation Patterns

We write permutations in one-line notation:

574826391

And we plot them by plotting the points $(i, \pi(i))$:

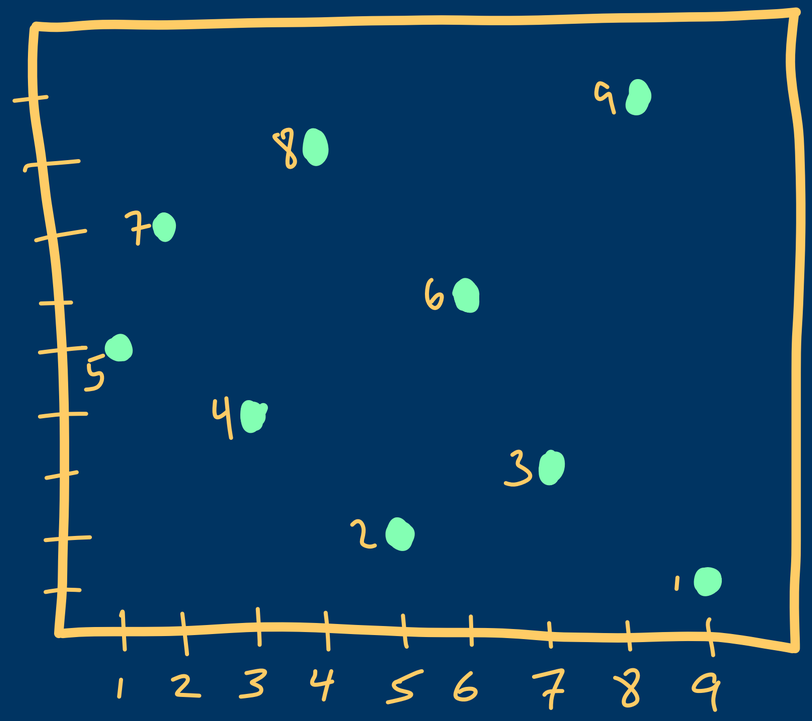


Permutation Patterns

We write permutations in one-line notation:

574826391

And we plot them by plotting the points $(i, \pi(i))$:

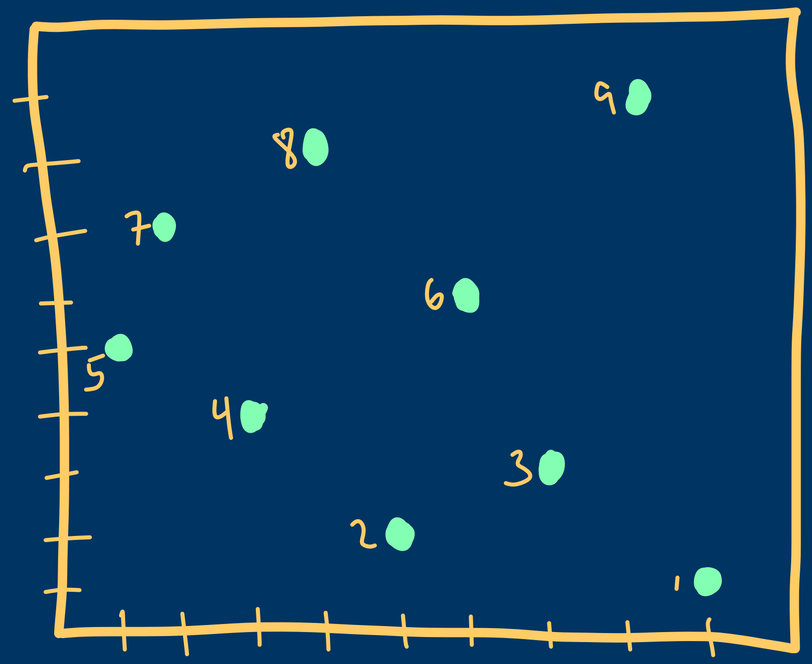


Permutation Patterns

We write permutations in one-line notation:

574826391

And we plot them by plotting the points $(i, \pi(i))$:

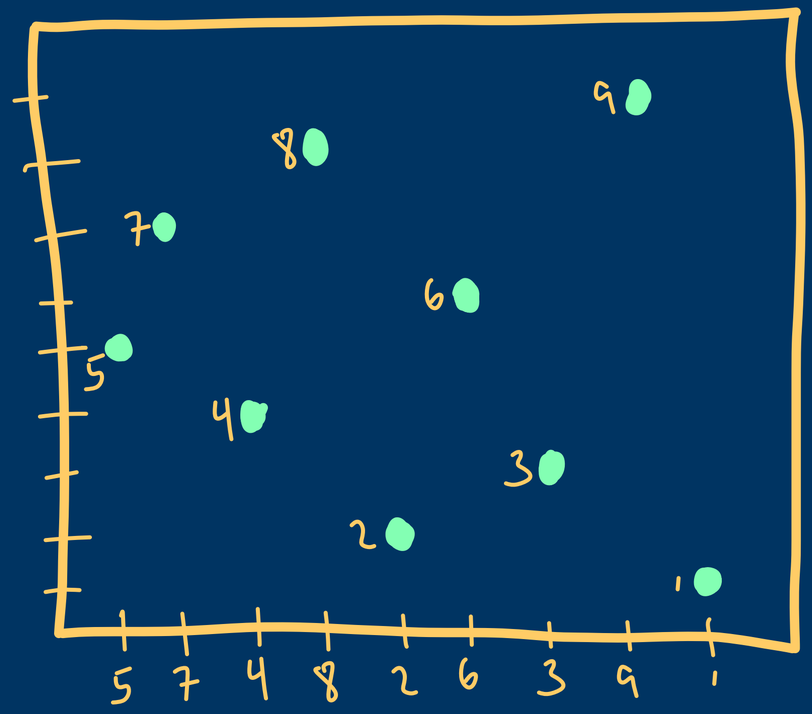


Permutation Patterns

We write permutations in one-line notation:

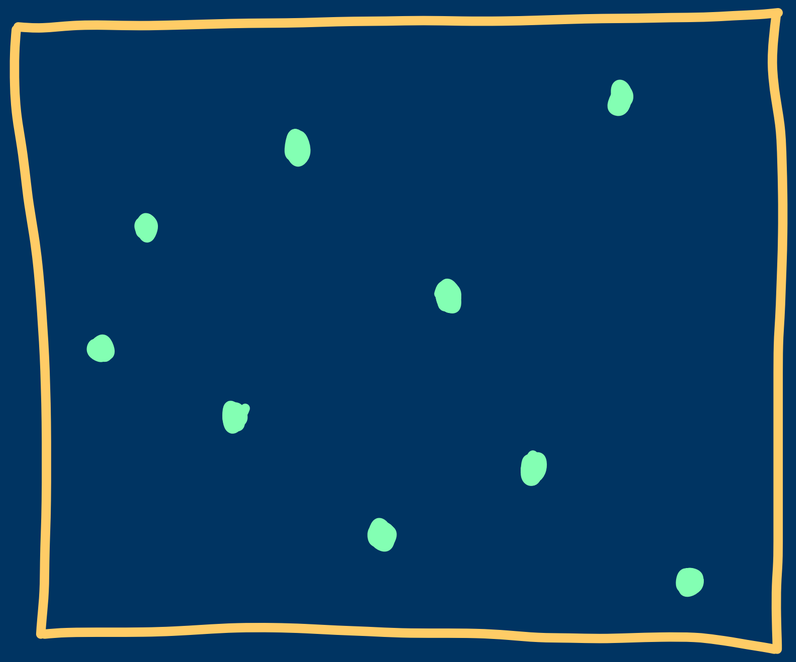
574826391

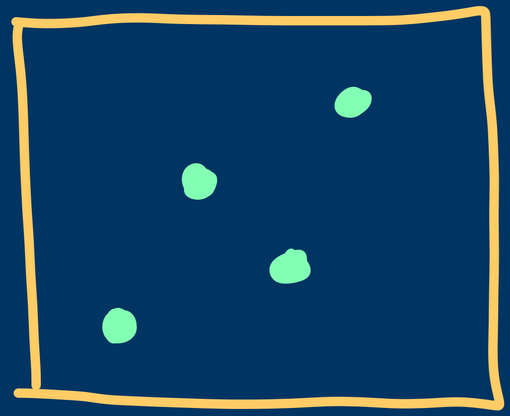
And we plot them by plotting the points $(i, \pi(i))$:



Permutation Patterns

Permutation π contains permutation σ if π has a subsequence of entries that is order-isomorphic to σ .



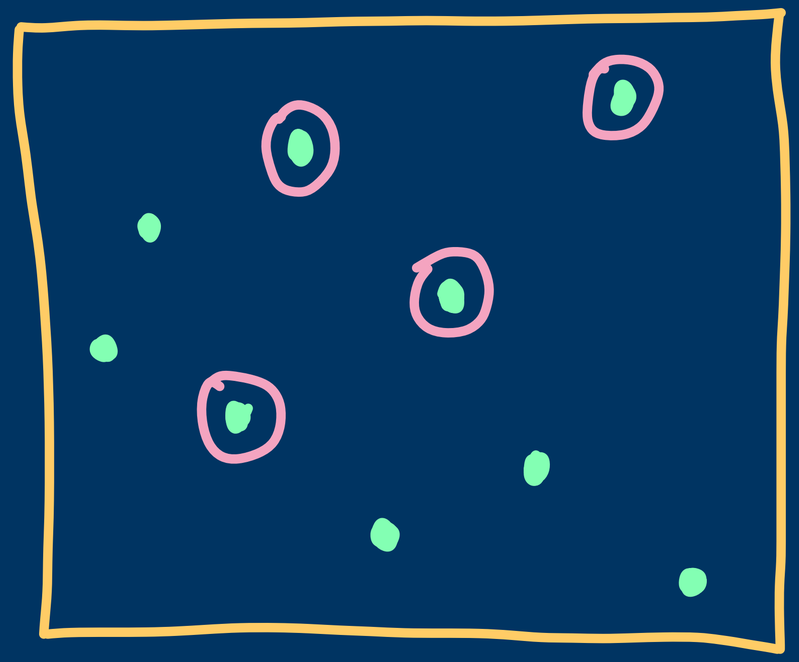


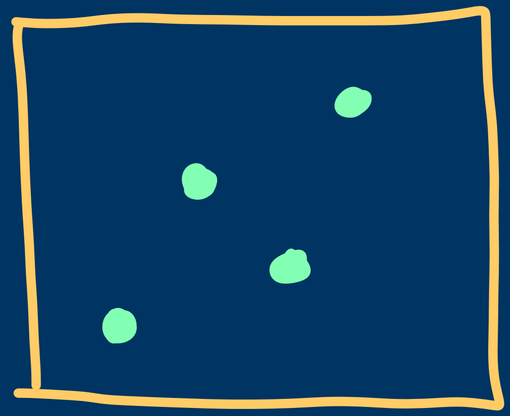
574826391 contains 1324

Otherwise we say π avoids σ.

Permutation Patterns

Permutation π contains permutation σ if π has a subsequence of entries that is order-isomorphic to σ .





574826391 contains 1324

Otherwise we say π avoids σ.

Permutation Patterns

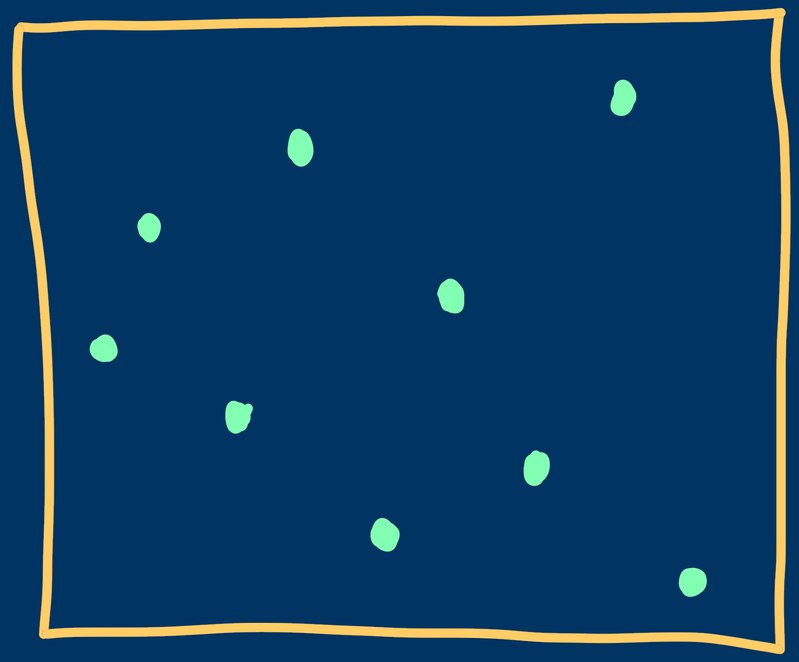
Permutation π contains permutation σ if π has a subsequence of entries that is order-isomorphic to σ .

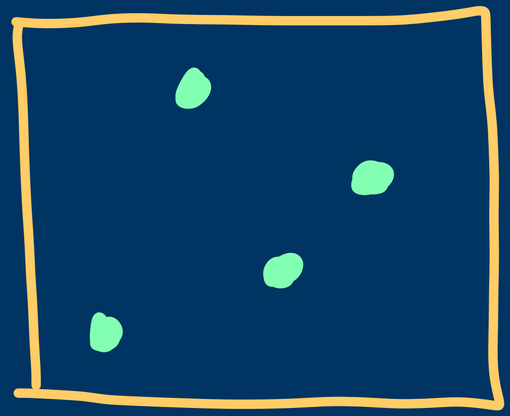
574826391 contains 1324

Otherwise we say π avoids σ.

Permutation Patterns

Permutation π contains permutation σ if π has a subsequence of entries that is order-isomorphic to σ .





574826391 avoids 1423

Otherwise we say π avoids σ.

Permutation Patterns

A permutation class is a set of permutations defined by avoiding a set of patterns (permutations).

Permutation Patterns

A permutation class is a set of permutations defined by avoiding a set of patterns (permutations).

For a set of permutations B, we write $\mathrm{Av}(B)$ to mean the class of permutations avoiding all the patterns in B.

Permutation Patterns

A permutation class is a set of permutations defined by avoiding a set of patterns (permutations).

For a set of permutations B, we write $\mathrm{Av}(B)$ to mean the class of permutations avoiding all the patterns in B.

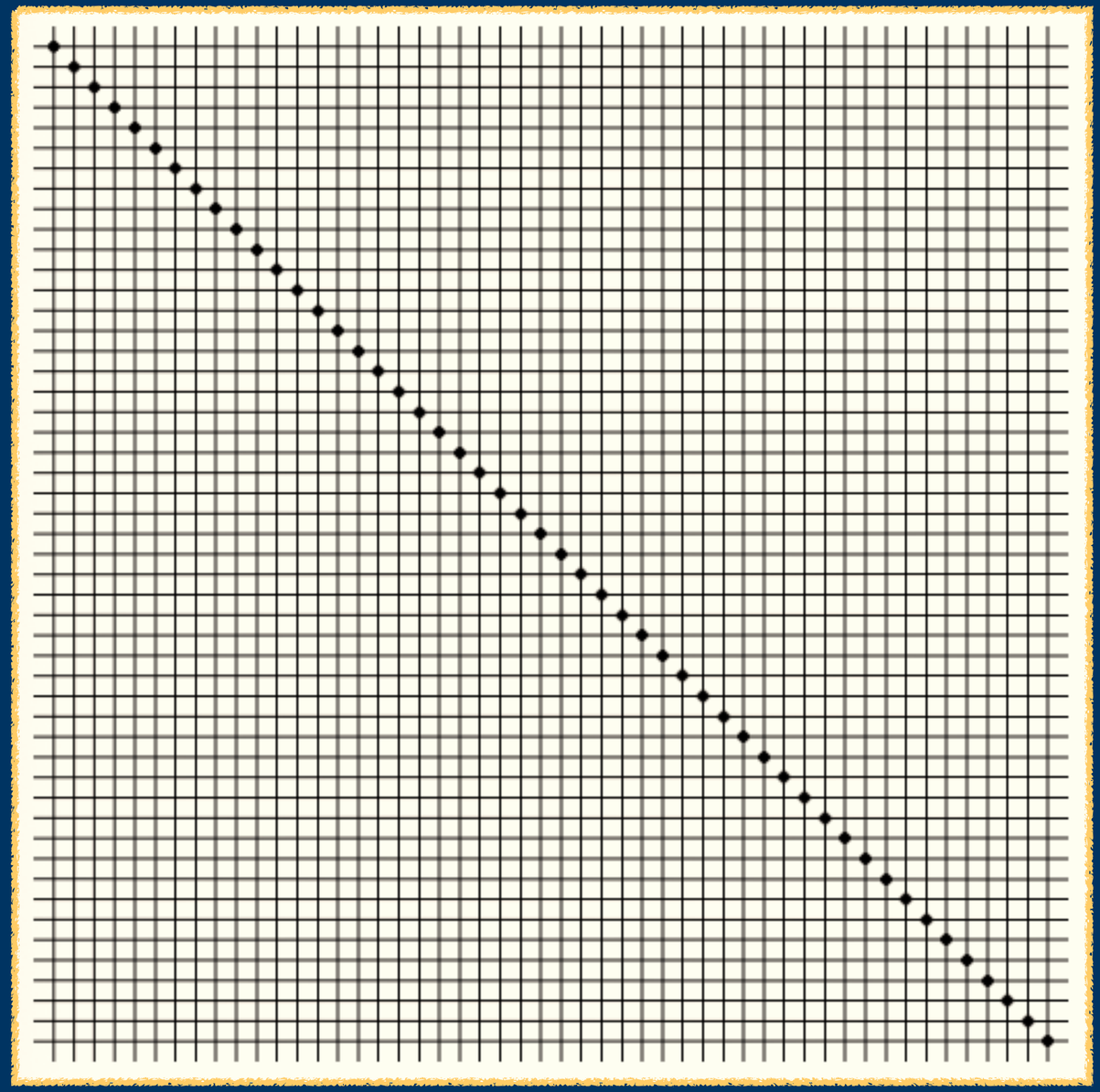
The permutation containment relation forms a poset on the set of all permutations of all lengths, and permutation classes are the downward closed sets in this poset.

Permutation Patterns

$$\mathrm{Av}(12)$$

$$a_n = 1$$

$$\frac{1}{1-x}$$

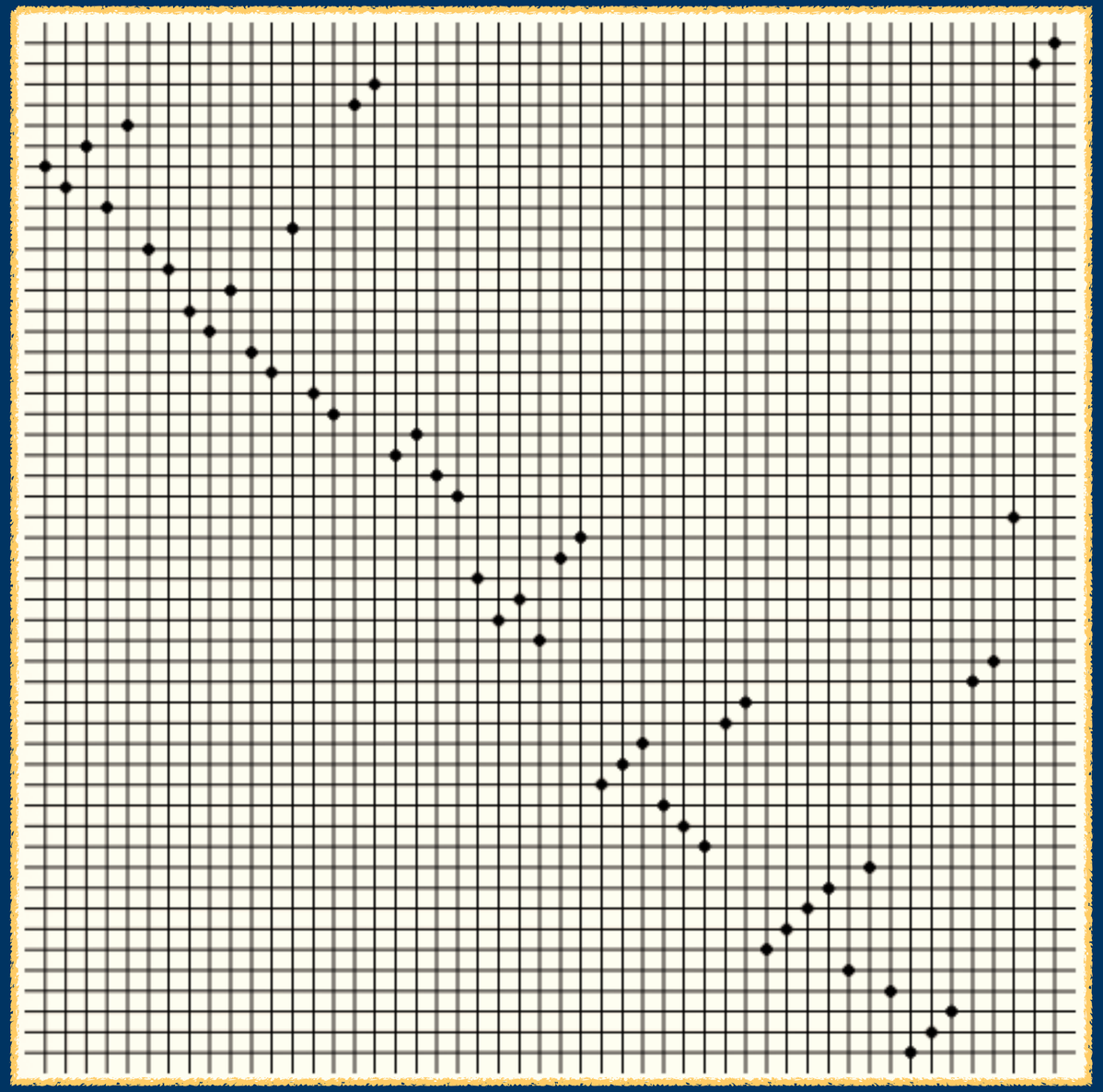


Permutation Patterns

$$\mathrm{Av}(132)$$

$$a_n = \frac{1}{n+1}\binom{2n}{n}$$

$$\frac{1-\sqrt{1-4x}}{2x}$$

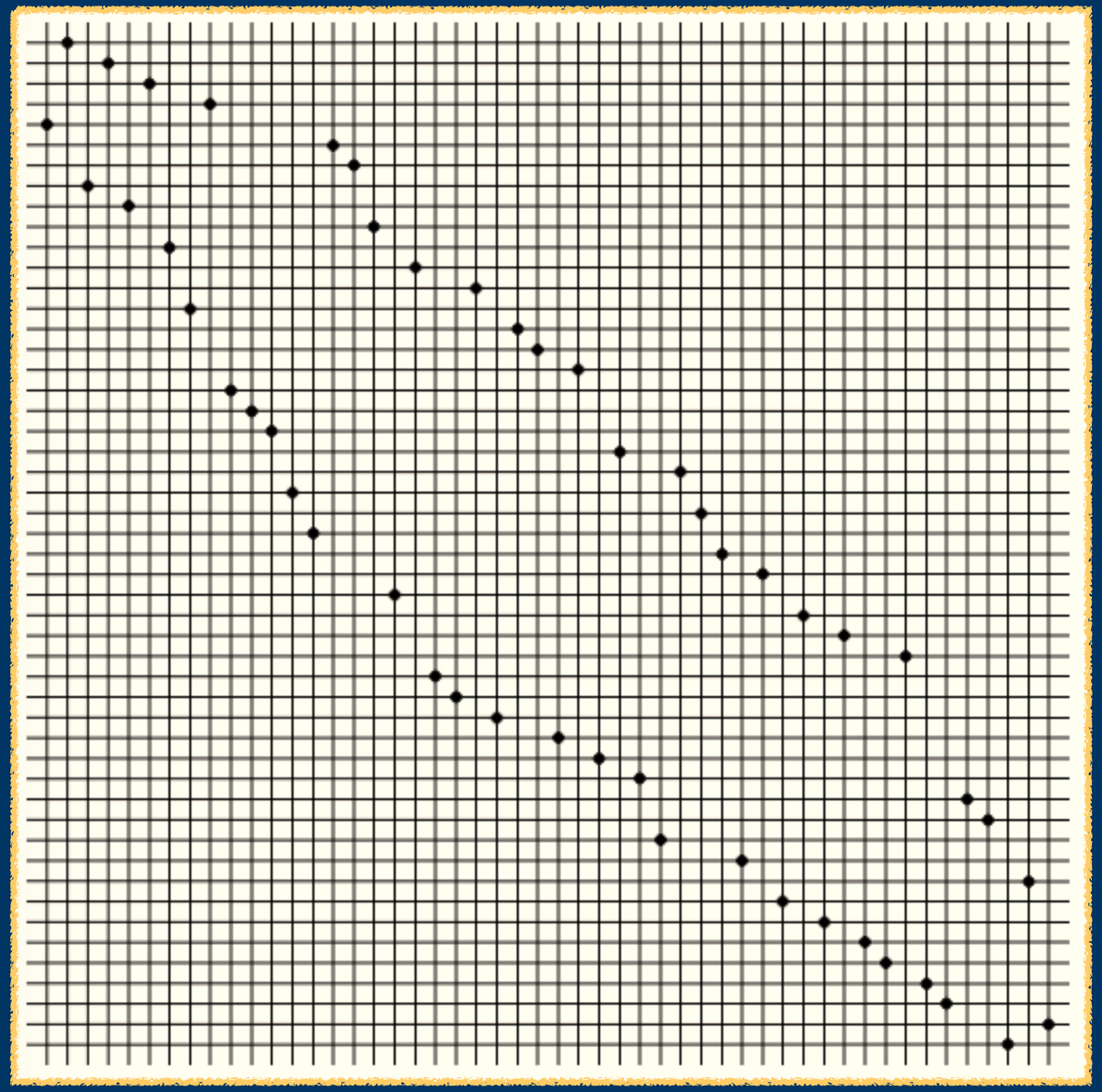


Permutation Patterns

$$\mathrm{Av}(123)$$

$$a_n = \frac{1}{n+1}\binom{2n}{n}$$

$$\frac{1-\sqrt{1-4x}}{2x}$$

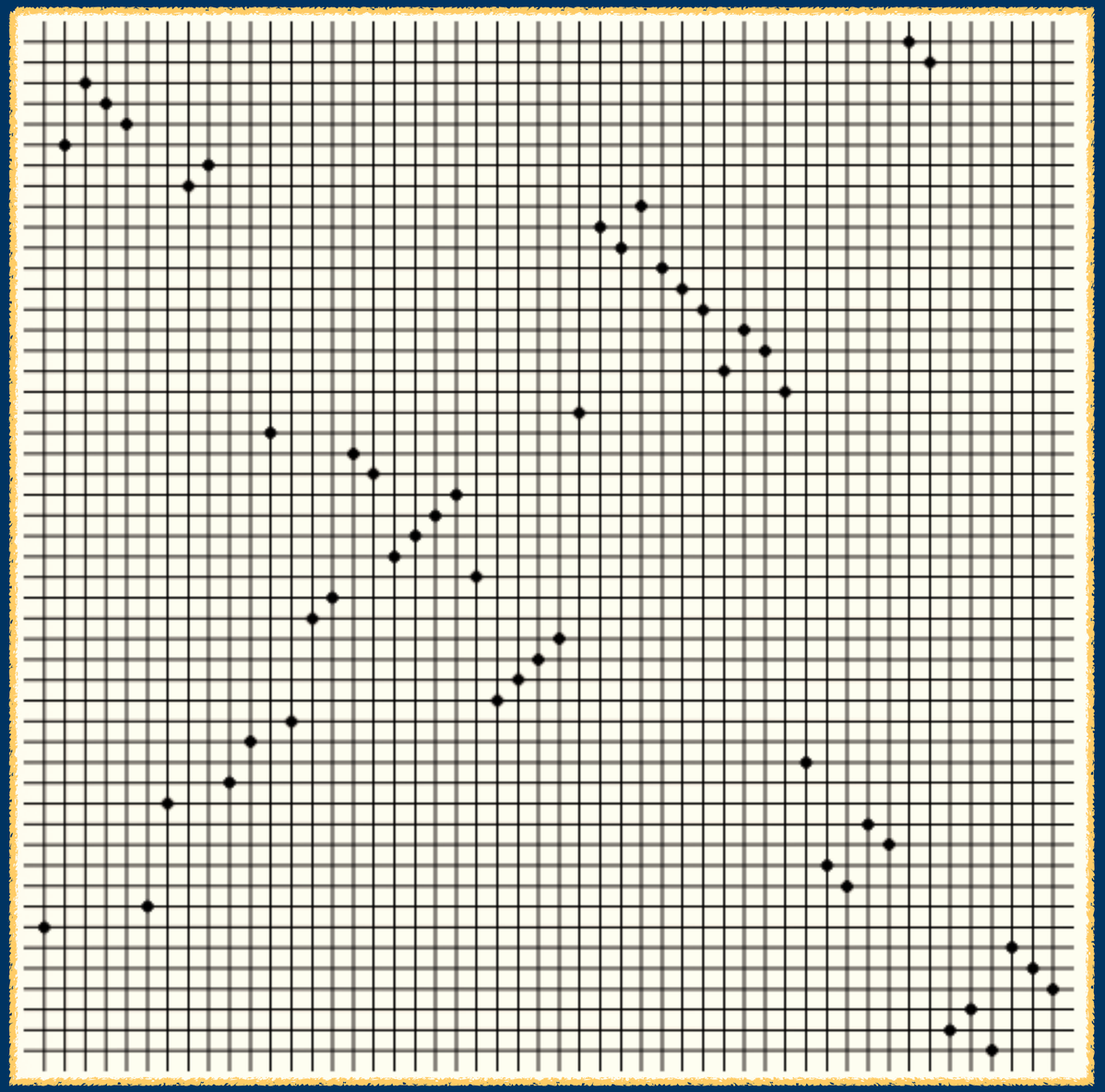


Permutation Patterns

$$\mathrm{Av}(2413, 3142)$$

$$a_n = \sum_{k=0}^{n-1} \frac{1}{k+1} \binom{n-1}{k} \binom{n+k}{k}$$

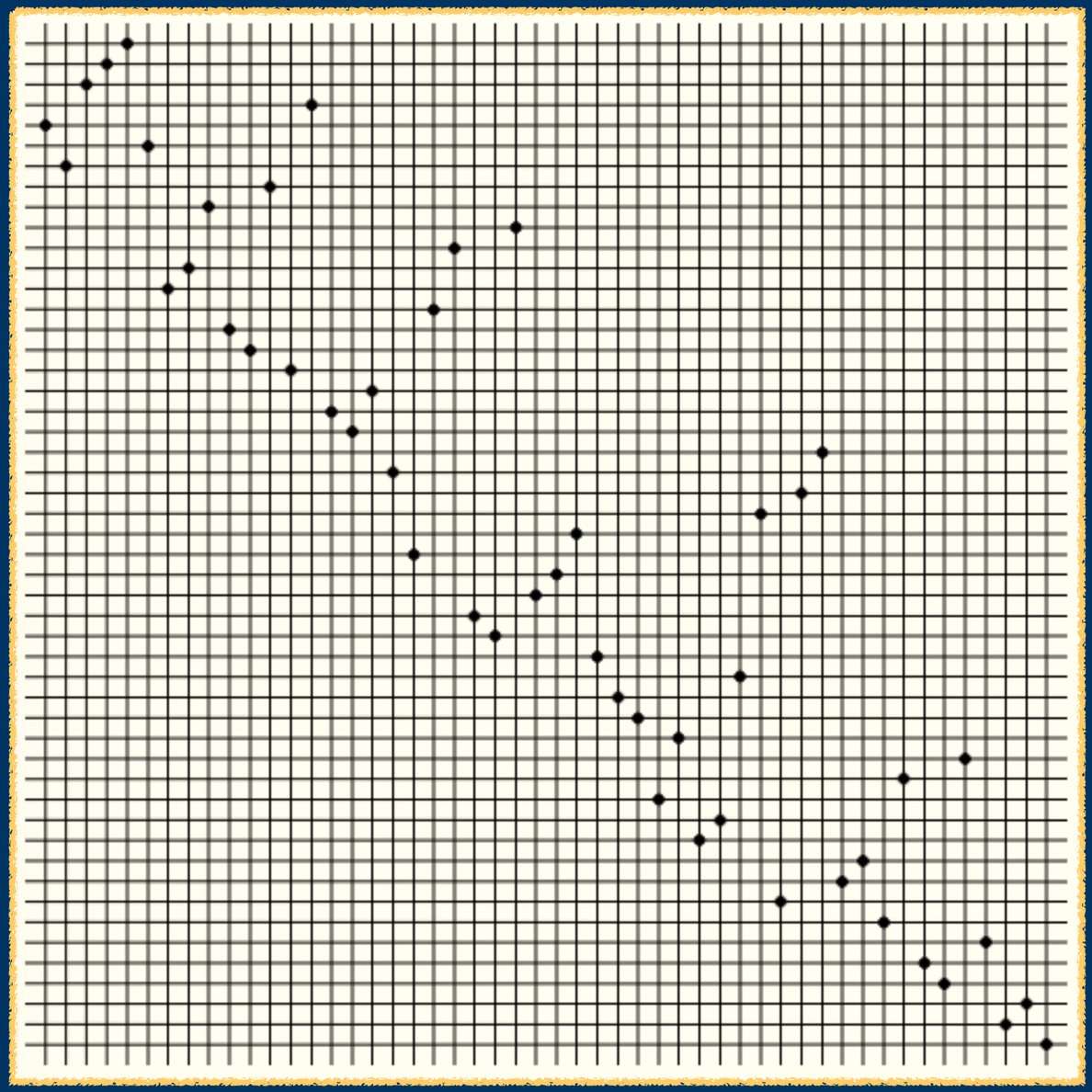
$$\frac{3 - x - \sqrt{1 - 6x + x^2}}{2}$$



Permutation Patterns

Av(1324, 1432)

???



Enumeration Schemes

Goal: Teach the computer how to search for structure in a permutation class.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

Goal: Teach the computer how to search for structure in a permutation class.

Input: A basis for a permutation class.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

Goal: Teach the computer how to search for structure in a permutation class.

Input: A basis for a permutation class.

Output: An “enumeration scheme” that describes how larger permutations in the class can be recursively described in terms of smaller permutations in the class.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

Goal: Teach the computer how to search for structure in a permutation class.

Input: A basis for a permutation class.

Output: An "enumeration scheme" that describes how larger permutations in the class can be recursively described in terms of smaller permutations in the class.

Enumeration Scheme ⟶ A polynomial-time algorithm to compute the number of permutations of length n.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Define $A(n) := \mathrm{Av}_n(123)$.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Define $A(n) := \mathrm{Av}_n(123)$.

Every non-empty permutation has a minimum entry somewhere.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

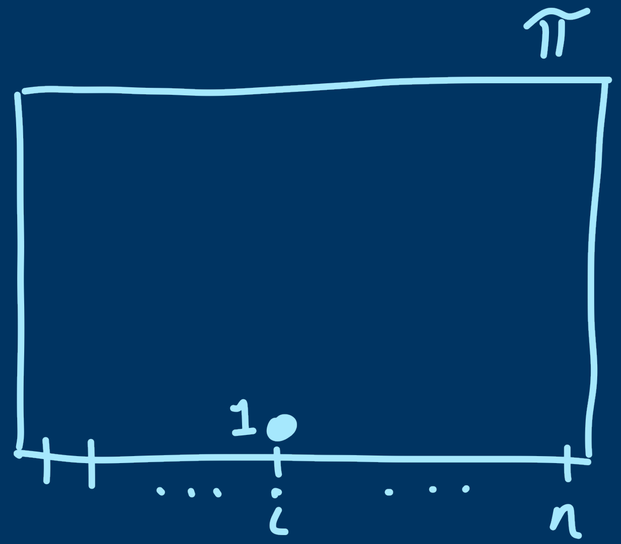
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Define $A(n) := \mathrm{Av}_n(123)$.

Every non-empty permutation has a minimum entry somewhere.

Define $A_1(n,i) := \{\pi \in A(n) : \pi(i) = 1\}$.



Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

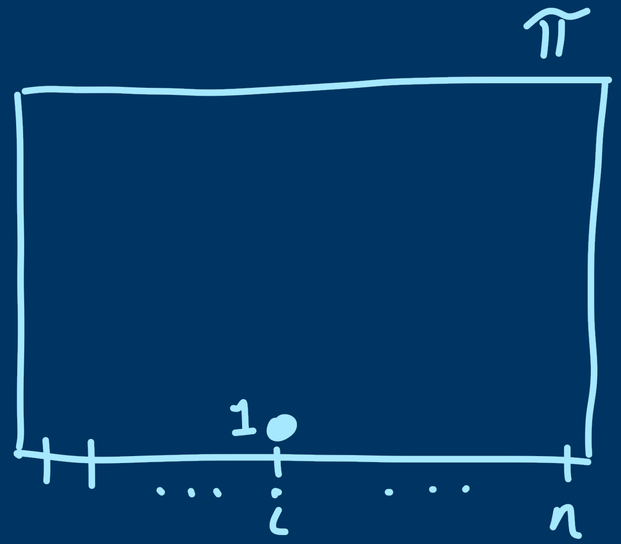
Received May 27, 1998

Define $A(n) := \mathrm{Av}_n(123)$.

Every non-empty permutation has a minimum entry somewhere.

Define $A_1(n,i) := \{\pi \in A(n) : \pi(i) = 1\}$.

Obviously $A(n) = \bigcup_{i=1}^{n} A_1(n,i)$.



Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

If $n \geq 2$, then the entry 2 is either to the left or to the right of 1.

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Enumeration Schemes and, More Importantly, Their Automatic Generation

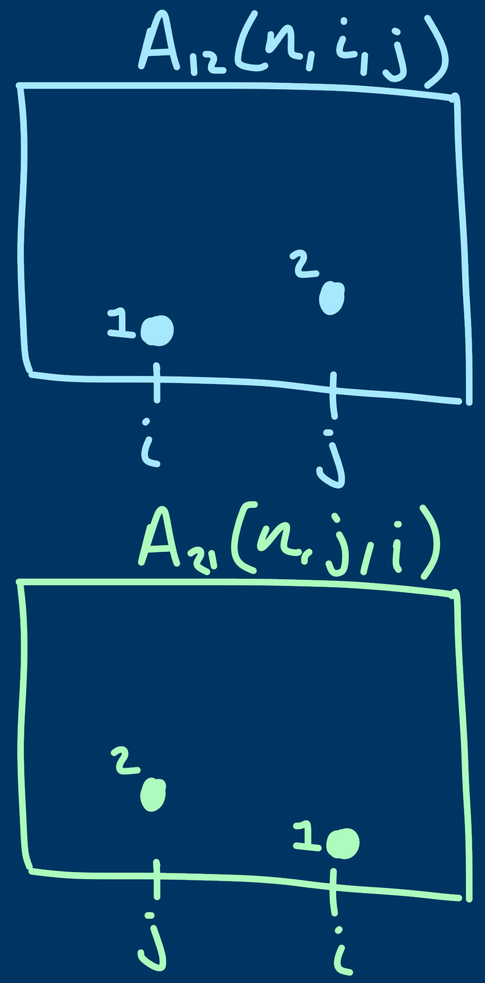
Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

If $n \geq 2$, then the entry 2 is either to the left or to the right of 1.

Define $A_{12}(n,i,j) := \{\pi \in A(n) : \pi(i) = 1,\ \pi(j) = 2,\ i < j\}$
and $A_{21}(n,j,i) := \{\pi \in A(n) : \pi(i) = 1,\ \pi(j) = 2,\ i > j\}$



Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

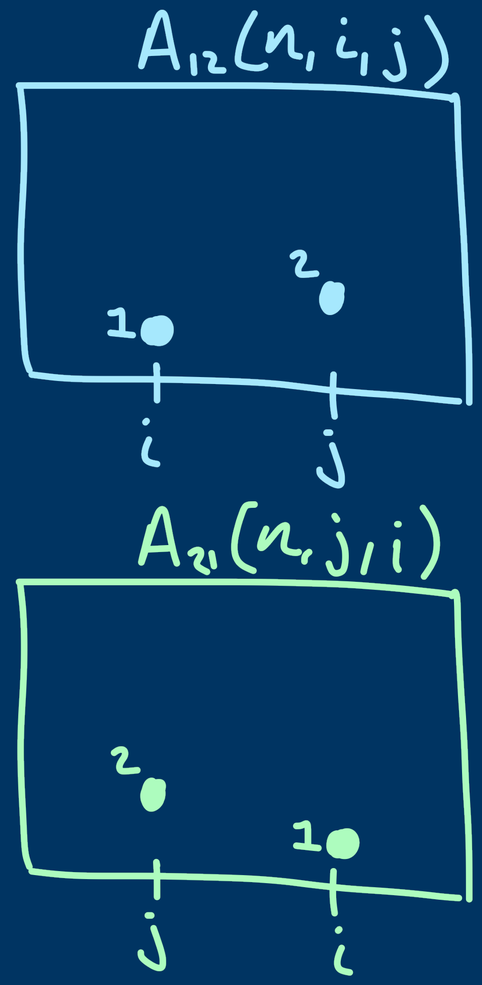
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

If $n \geq 2$, then the entry 2 is either to the left or to the right of 1.

Define $A_{12}(n,i,j) := \{\pi \in A(n) : \pi(i) = 1,\ \pi(j) = 2,\ i < j\}$
and $A_{21}(n,j,i) := \{\pi \in A(n) : \pi(i) = 1,\ \pi(j) = 2,\ i > j\}$

Obviously $A_1(n,i) = \left(\bigcup_{j=1}^{i-1} A_{21}(n,j,i) \right) \cup \left(\bigcup_{j=i+1}^{n} A_{12}(n,i,j) \right).$



Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

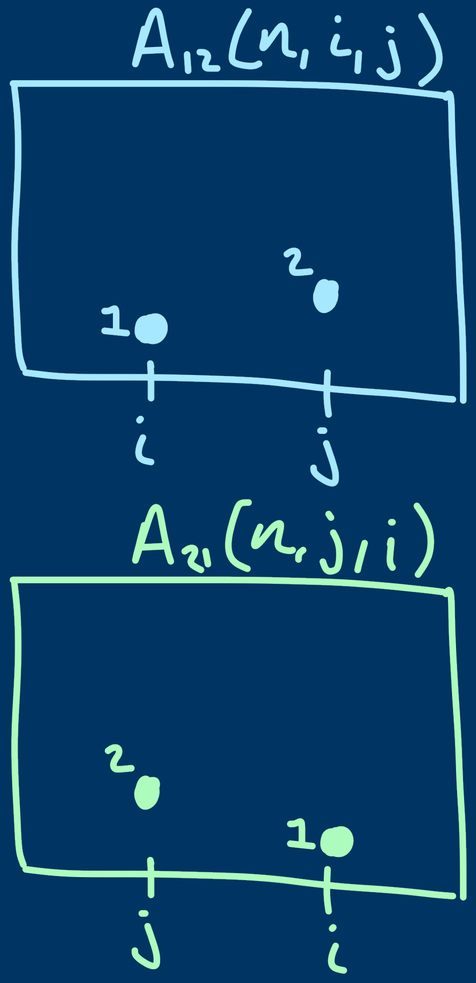
Received May 27, 1998

If $n \geq 2$, then the entry 2 is either to the left or to the right of 1.

Define $A_{12}(n,i,j) := \{\pi \in A(n) : \pi(i) = 1,\ \pi(j) = 2,\ i < j\}$
and $A_{21}(n,j,i) := \{\pi \in A(n) : \pi(i) = 1,\ \pi(j) = 2,\ i > j\}$

Obviously $A_1(n,i) = \left(\bigcup_{j=1}^{i-1} A_{21}(n,j,i) \right) \cup \left(\bigcup_{j=i+1}^{n} A_{12}(n,i,j) \right)$.

We could do this forever...



Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

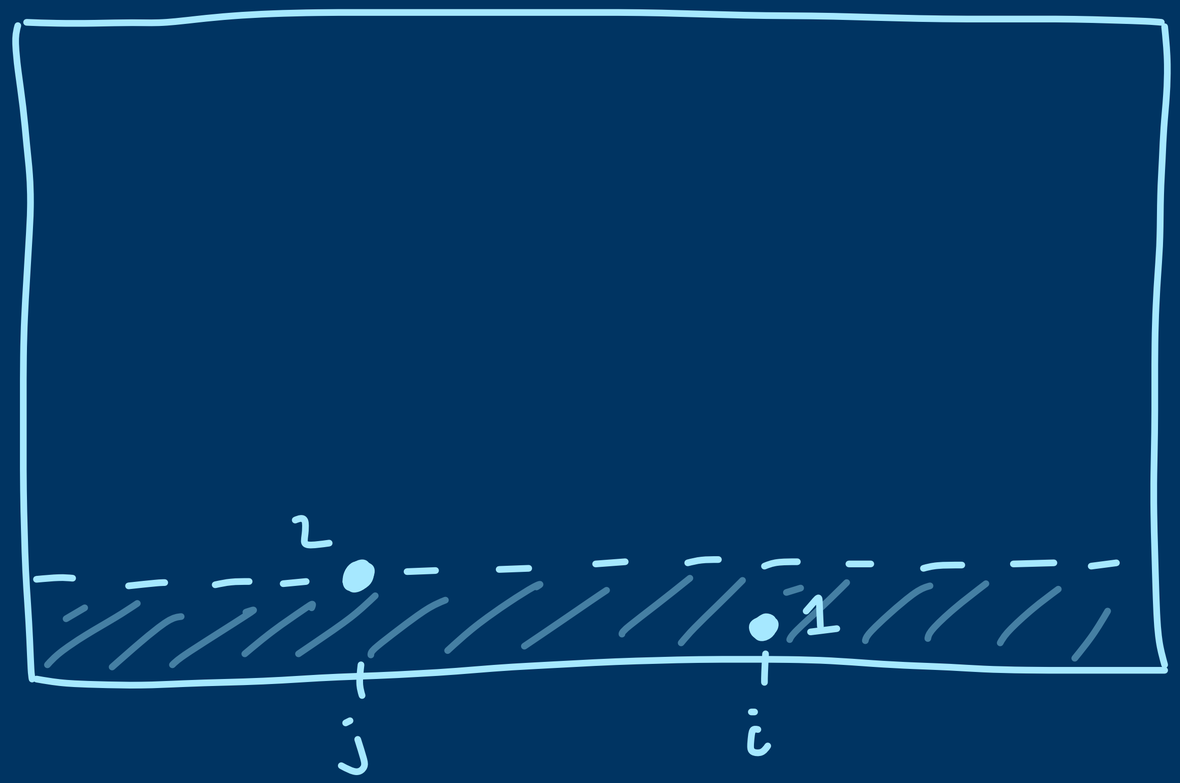
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$



Let π be any permutation with $\pi(j)=2$, $\pi(i)=1$, $j<i$.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

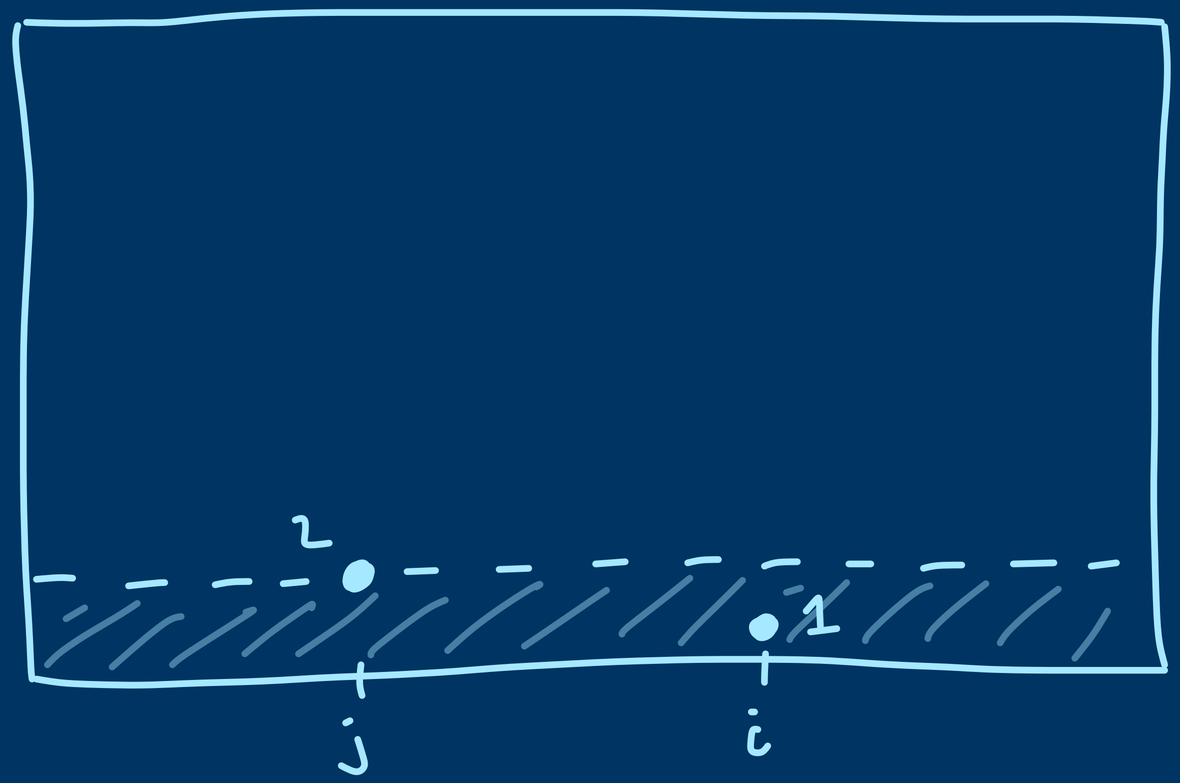
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$



Let π be any permutation with $\pi(j)=2$, $\pi(i)=1$, $j<i$.

π contains 123
iff
$\pi - \pi(i)$ contains 123

Enumeration Schemes and, More Importantly, Their Automatic Generation

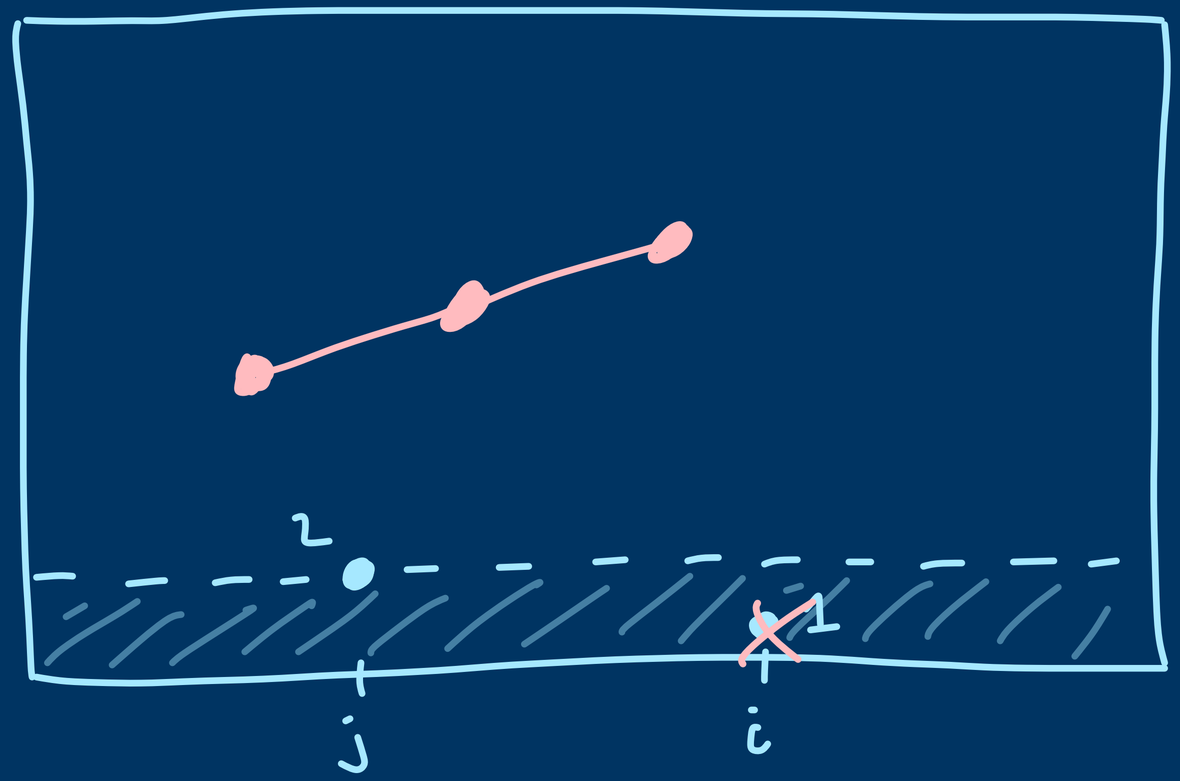
Doron Zeilberger*
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$



π contains 123
iff
$\pi - \pi(i)$ contains 123

Any 123 that doesn't involve $\pi(i)$ is obviously still in $\pi - \pi(i)$.

Enumeration Schemes and, More Importantly, Their Automatic Generation

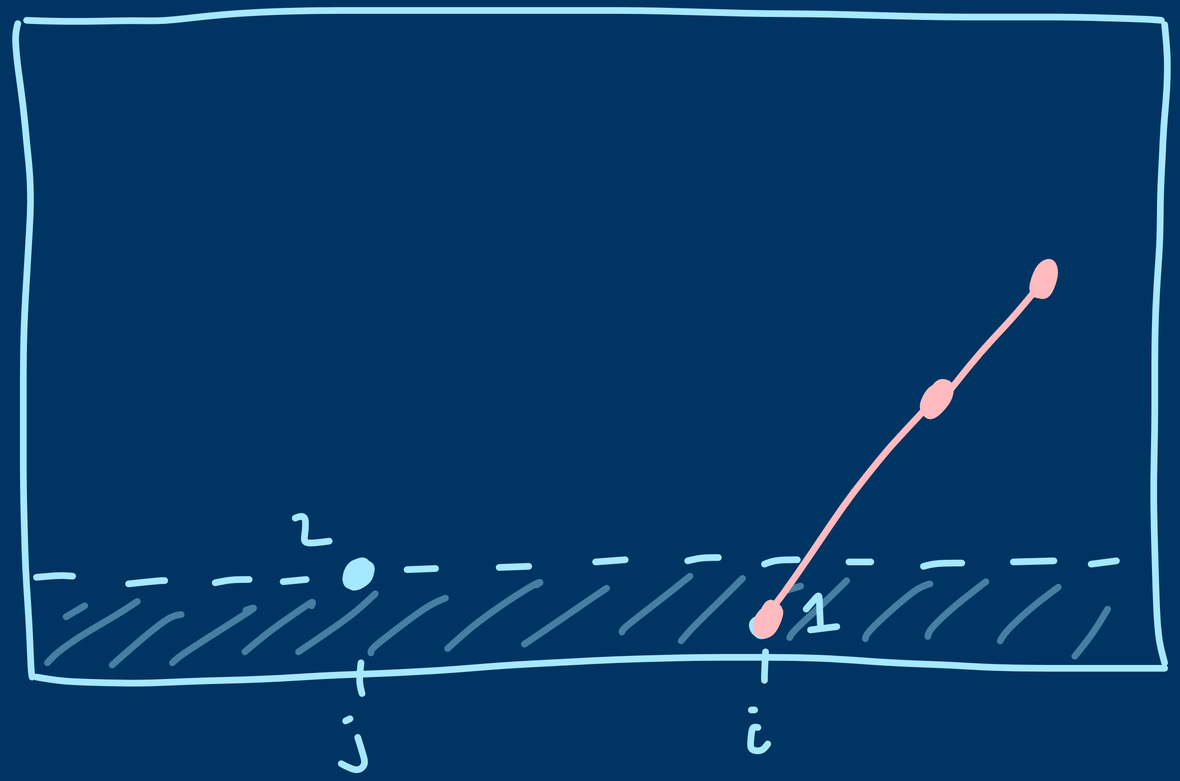
Doron Zeilberger*
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$



Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

π contains 123
iff
$\pi - \pi(i)$ contains 123

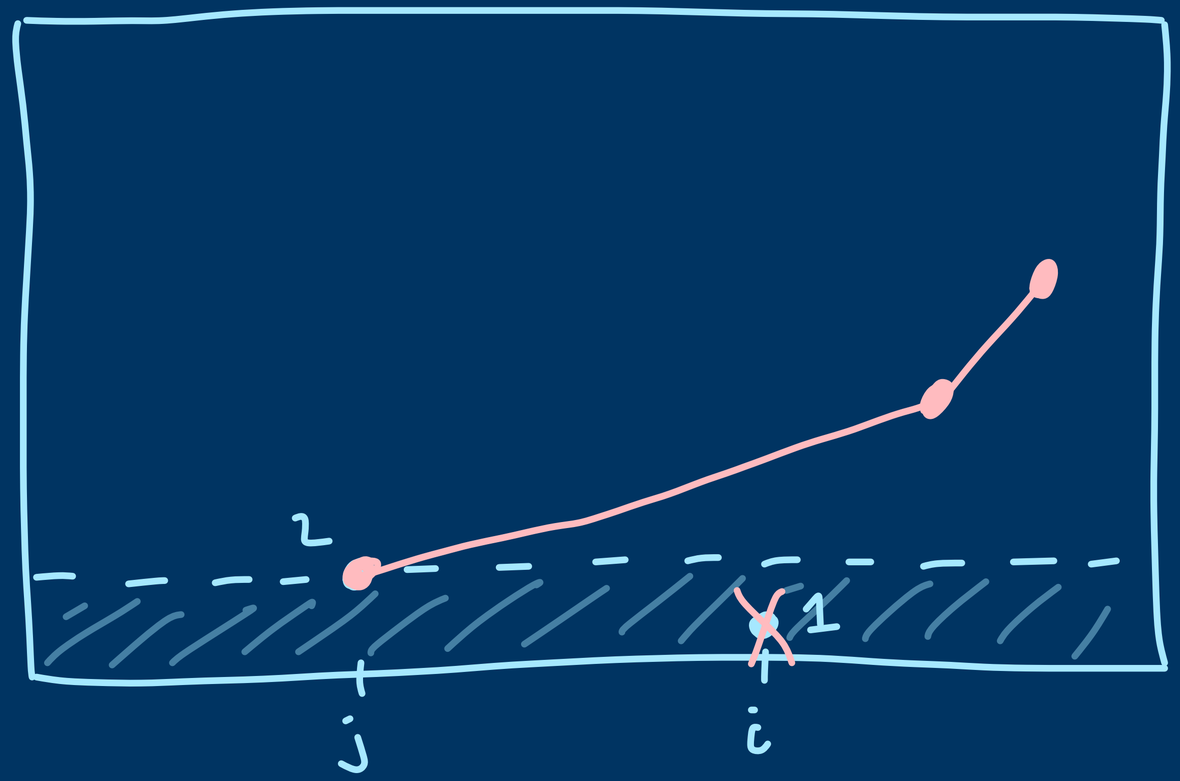
If an occurrence of 123 involves $\pi(i)$, it must be the 1.

$\pi - \pi(i)$ has a 123 with $\pi(j)$ substituting for $\pi(i)$.

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$



π contains 123
iff
$\pi - \pi(i)$ contains 123

If an occurrence of 123 involves $\pi(i)$, it must be the 1.

$\pi - \pi(i)$ has a 123 with $\pi(j)$ substituting for $\pi(i)$.

Enumeration Schemes and, More Importantly, Their Automatic Generation

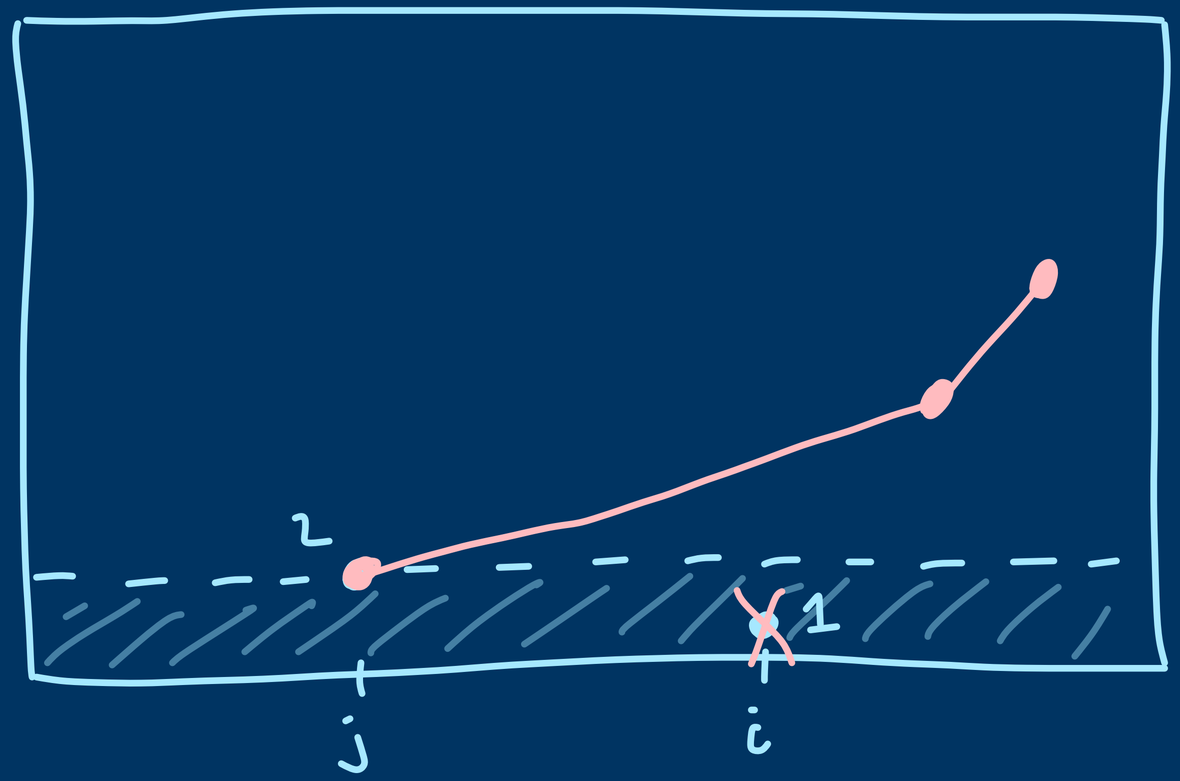
Doron Zeilberger*
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$



π contains 123
iff
$\pi - \pi(i)$ contains 123

So there is a bijection between $A_{21}(n,j,i)$ and $A_1(n-1,j)$.

Enumeration Schemes and, More Importantly, Their Automatic Generation

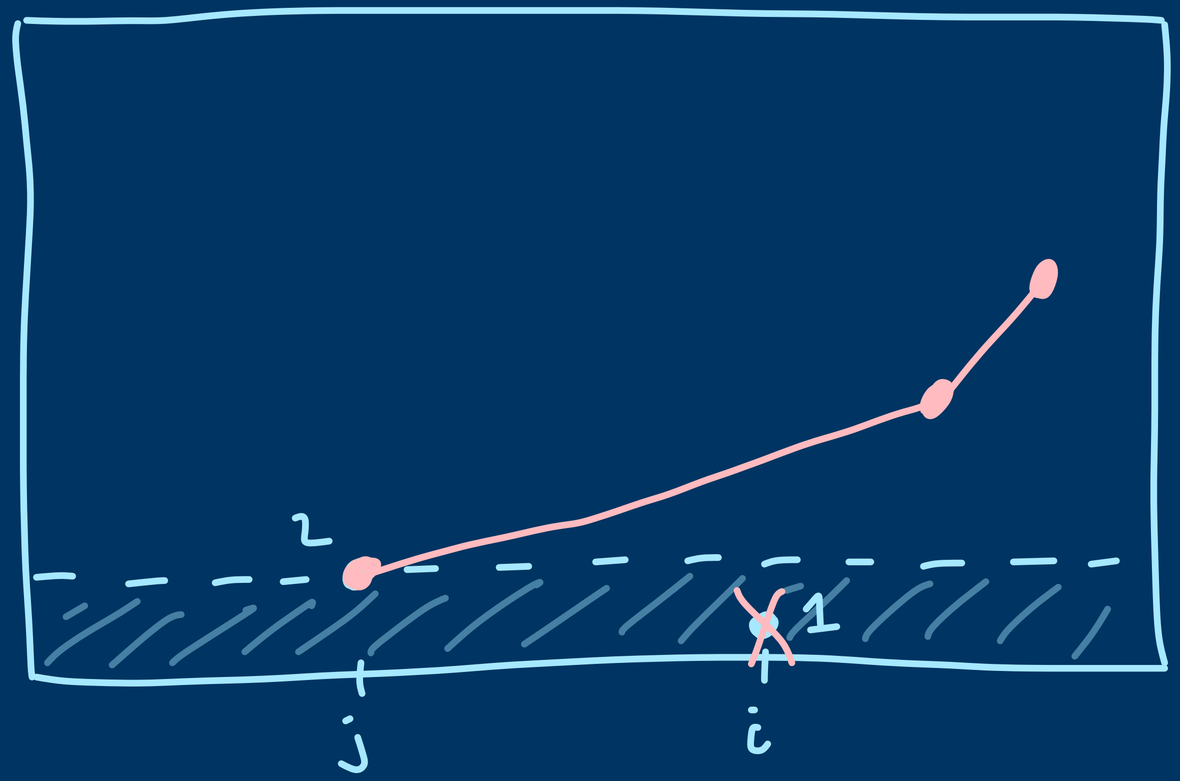
Doron Zeilberger*
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 1: $|A_{21}(n,j,i)| = |A_1(n-1,j)|$ ✓



π contains 123
iff
$\pi - \pi(i)$ contains 123

So there is a bijection between $A_{21}(n,j,i)$ and $A_1(n-1,j)$.

Enumeration Schemes and, More Importantly, Their Automatic Generation

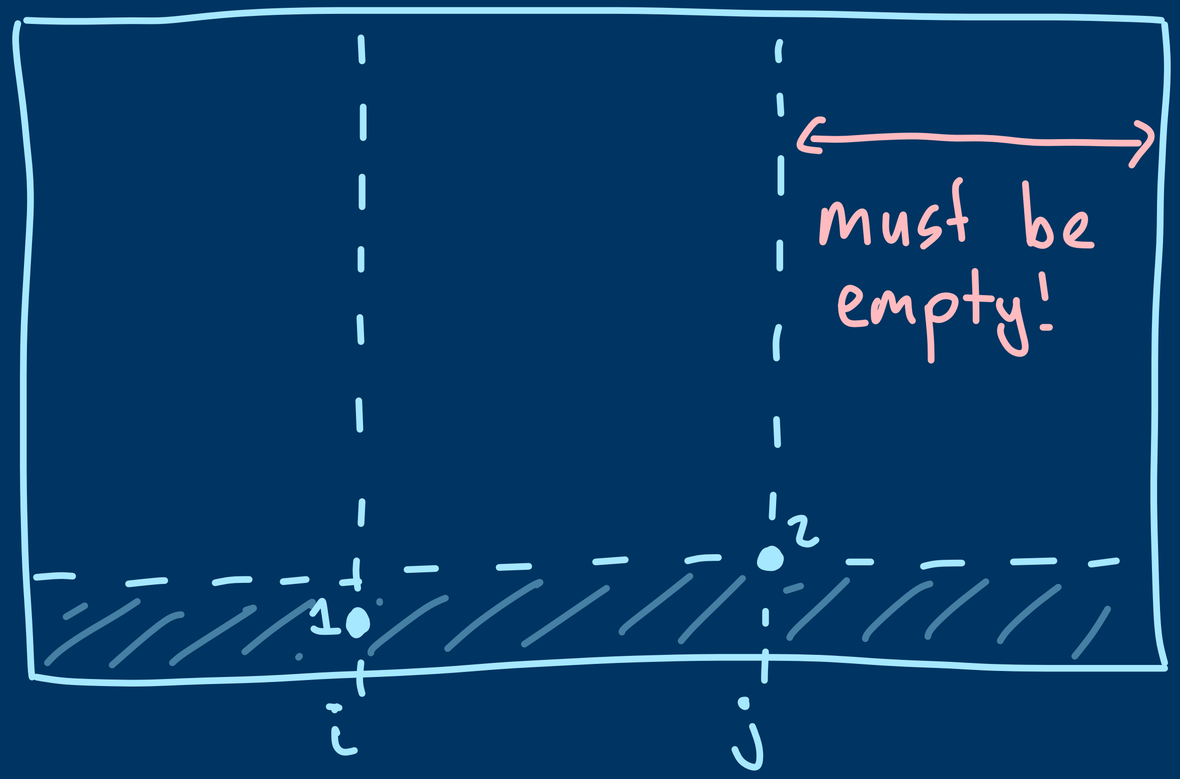
Doron Zeilberger*
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 2: $|A_{12}(n,i,j)| = \begin{cases} 0, & j < n \\ |A_1(n-1,i)|, & j = n \end{cases}$



Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

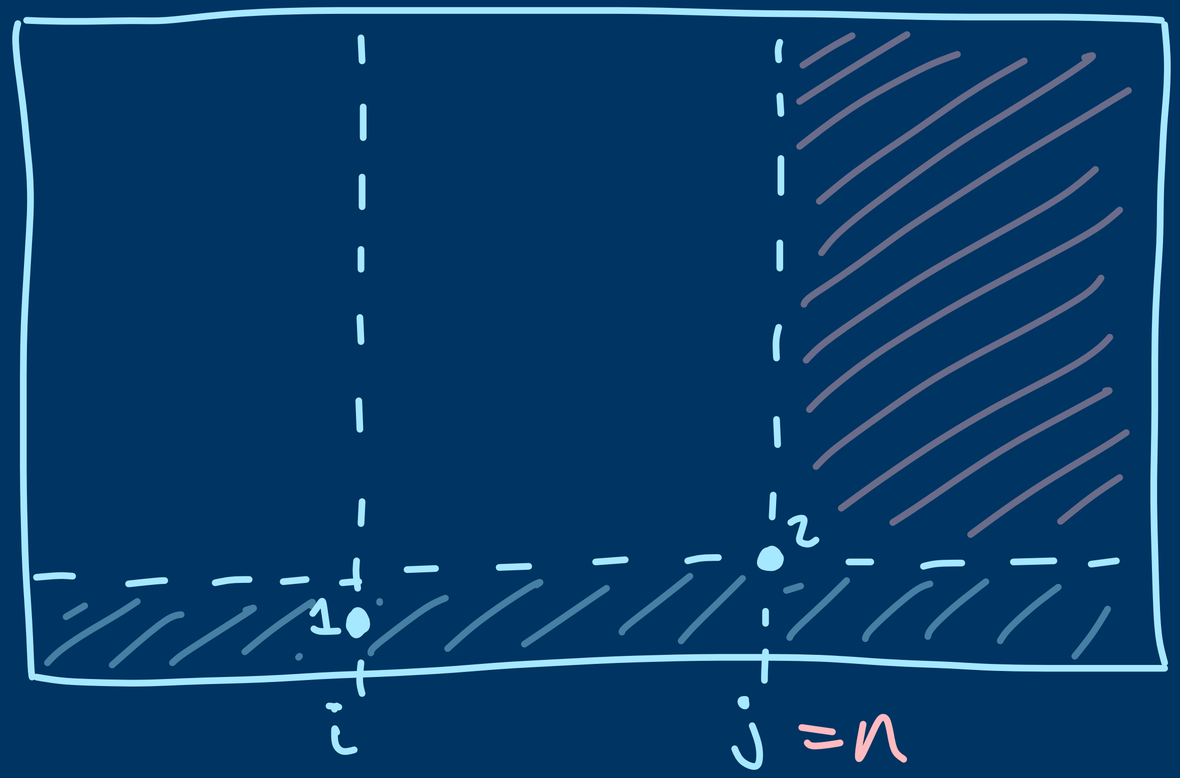
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Claim 2: $|A_{12}(n,i,j)| = \begin{cases} 0, & j < n \\ |A_1(n-1,i)|, & j = n \end{cases}$



Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

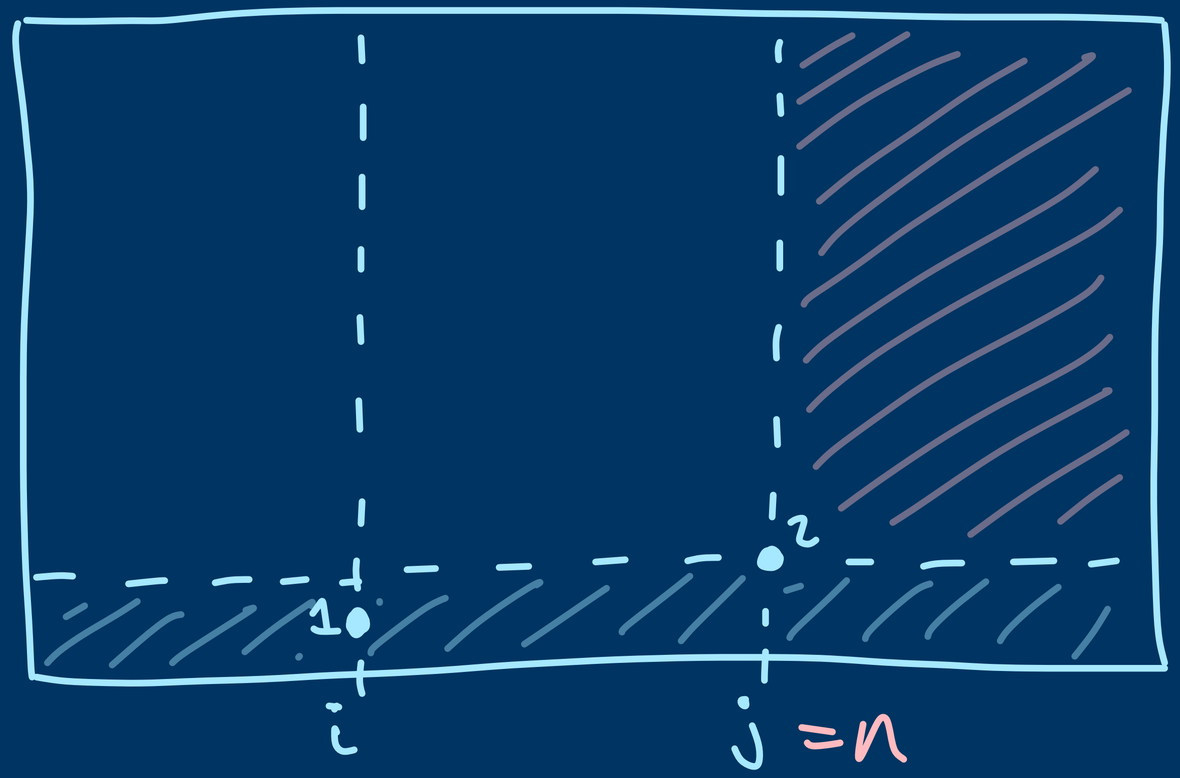
Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Claim 2: $$|A_{12}(n,i,j)| = \begin{cases} 0, & j < n \\ |A_1(n-1,i)|, & j = n \end{cases}$$



$\Rightarrow A_{12}(n,i,j) = \emptyset$ when $j < n$.

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

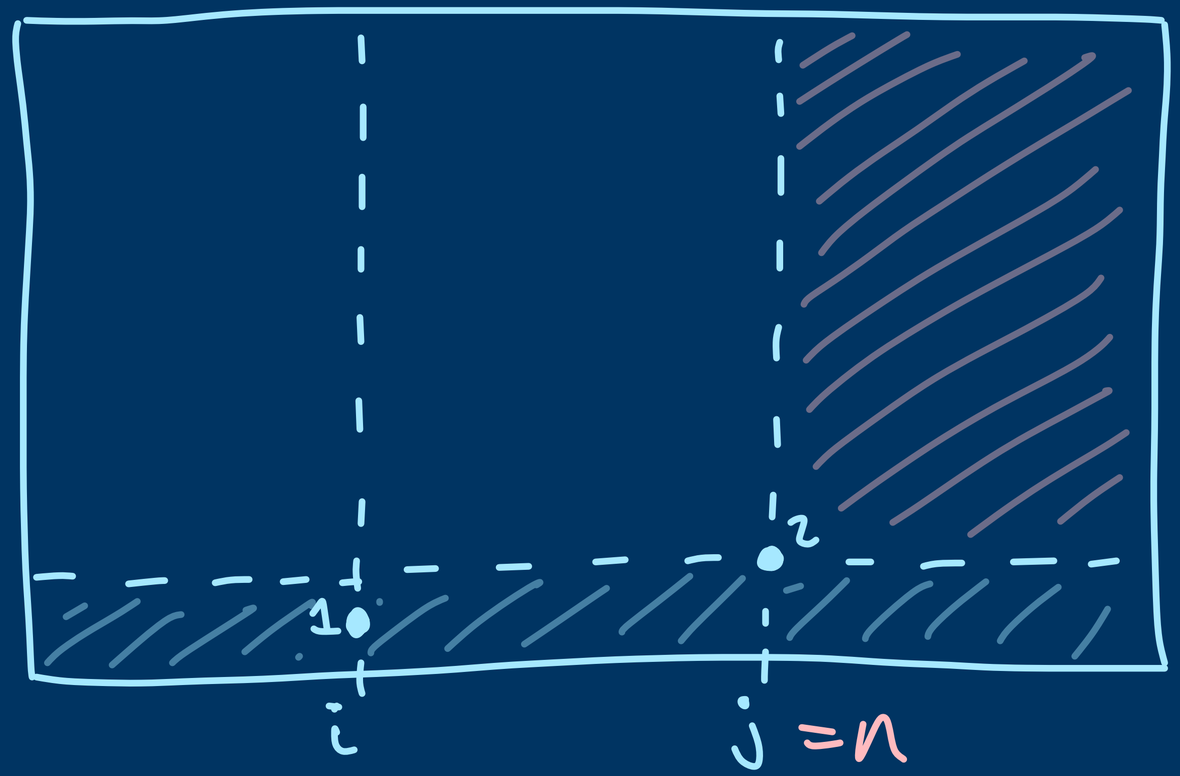
Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Claim 2: $$|A_{12}(n,i,j)| = \begin{cases} 0, & j < n \\ |A_1(n-1,i)|, & j = n \end{cases}$$



$\Rightarrow A_{12}(n,i,j) = \emptyset$ when $j < n$.

When $j = n$, $\pi(j)$ can be deleted without destroying any 123 patterns.

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

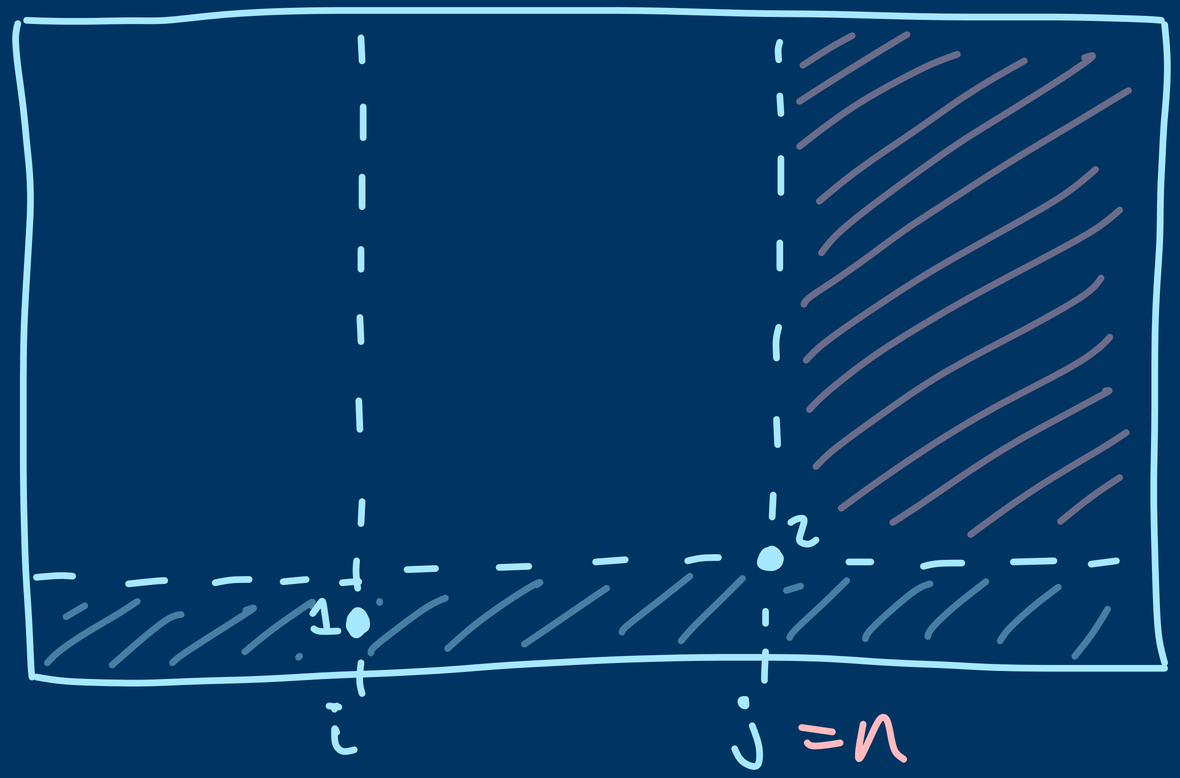
Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Claim 2: $$|A_{12}(n,i,j)| = \begin{cases} 0, & j < n \\ |A_1(n-1,i)|, & j = n \end{cases}$$



$\Rightarrow A_{12}(n,i,j) = \emptyset$ when $j < n$.

When $j = n$, $\pi(j)$ can be deleted without destroying any 123 patterns.

$\Rightarrow A_{12}(n,i,n) \cong A_1(n-1,i)$

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

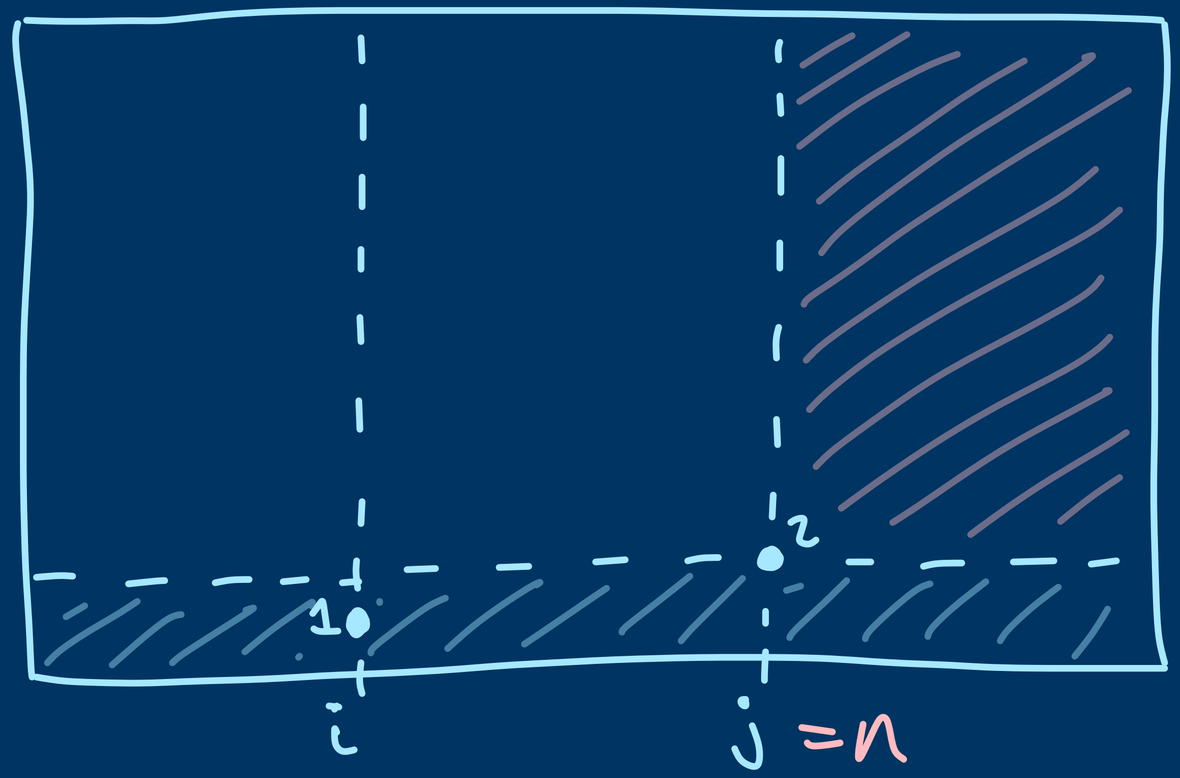
Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Claim 2: $$|A_{12}(n,i,j)| = \begin{cases} 0, & j < n \\ |A_1(n-1,i)|, & j = n \end{cases}$$ ✓



$\Rightarrow A_{12}(n,i,j) = \emptyset$ when $j < n$.

When $j = n$, $\pi(j)$ can be deleted without destroying any 123 patterns.

$\Rightarrow A_{12}(n,i,n) \cong A_1(n-1,i)$

Enumeration Schemes

"The Most Trivial Non-Trivial Example" – $\mathrm{Av}(123)$

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

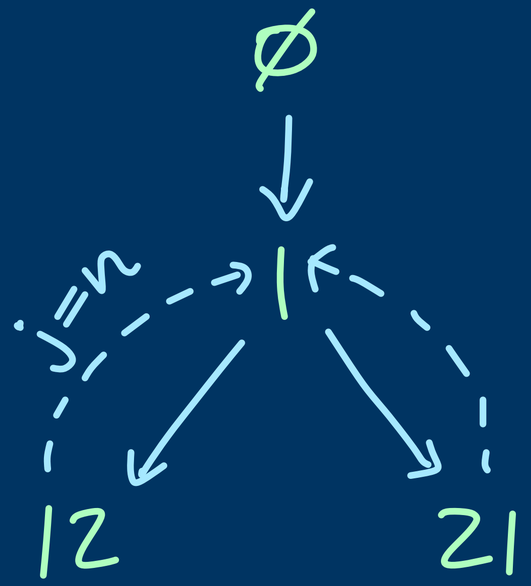
Received May 27, 1998

$$A(n) = \bigcup_{i=1}^{n} A_1(n,i)$$

$$A_1(n,i) = \left(\bigcup_{j=1}^{i-1} A_{21}(n,j,i) \right) \cup \left(\bigcup_{j=i+1}^{n} A_{12}(n,i,j) \right)$$

$$A_{21}(n,j,i) \cong A_1(n-1,j)$$

$$A_{12}(n,i,j) \cong \begin{cases} \varnothing & , j < n \\ A_1(n-1,i), & j = n \end{cases}$$



Enumeration Schemes

Big Picture:

- The computer splits the whole set $A(n)$ further and further based on the pattern formed by the bottom entries.

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Enumeration Schemes

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Big Picture:

- The computer splits the whole set $A(n)$ further and further based on the pattern formed by the bottom entries.
- At each step it checks if any of the entries are “reversibly deletable”. If so, this branch of the search tree doesn’t need to be split further.

Enumeration Schemes

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998

Big Picture:

- The computer splits the whole set $A(n)$ further and further based on the pattern formed by the bottom entries.
- At each step it checks if any of the entries are “reversibly deletable”. If so, this branch of the search tree doesn’t need to be split further.
- If all branches finish, we get an enumeration scheme, which gives us a *polynomial-time algorithm* to count the number of permutations of length n, but does not give us the *generating function*.

Enumeration Schemes

Zeilberger's method is:

Enumeration Schemes and, More Importantly, Their Automatic Generation

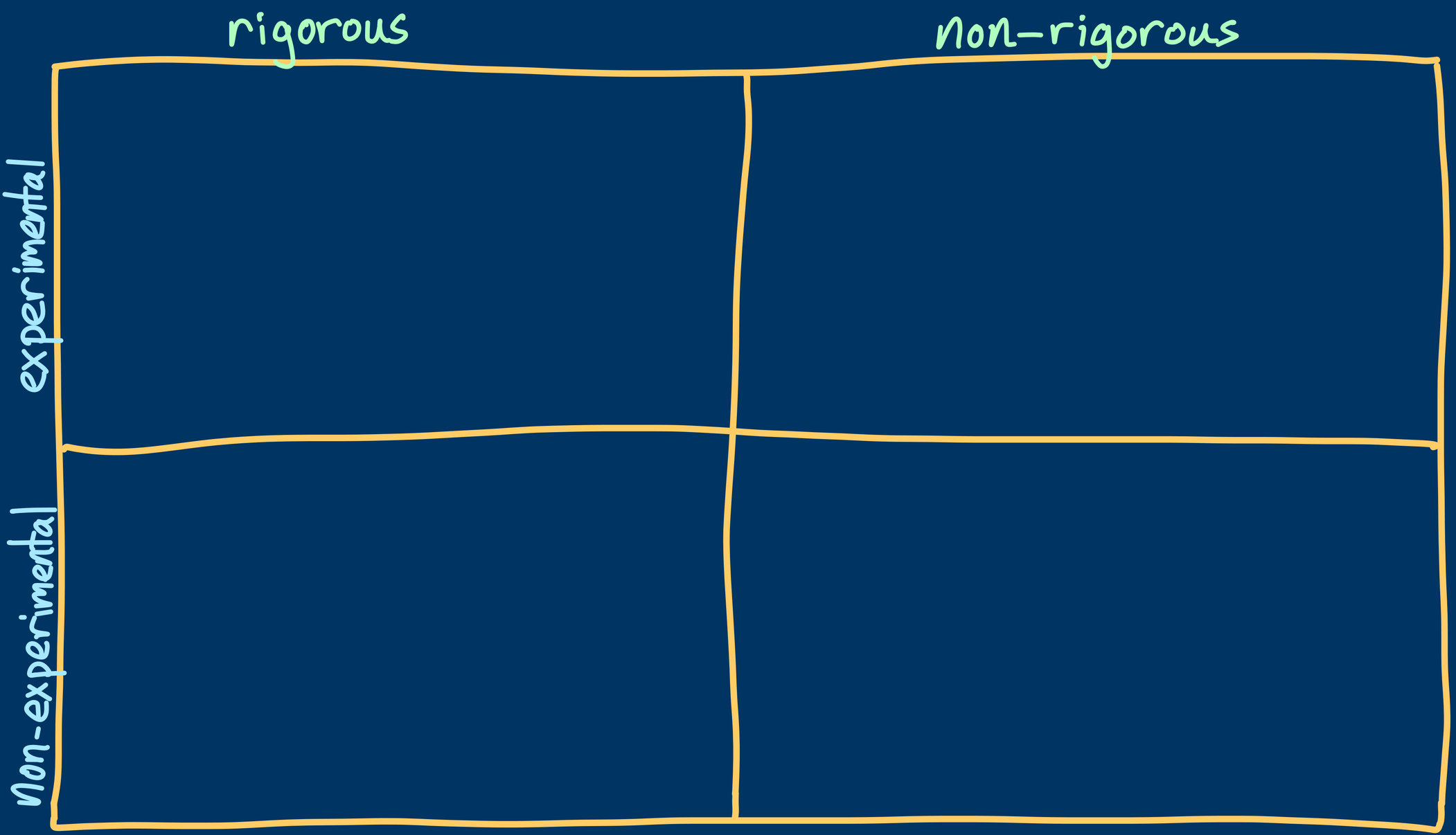
Doron Zeilberger*

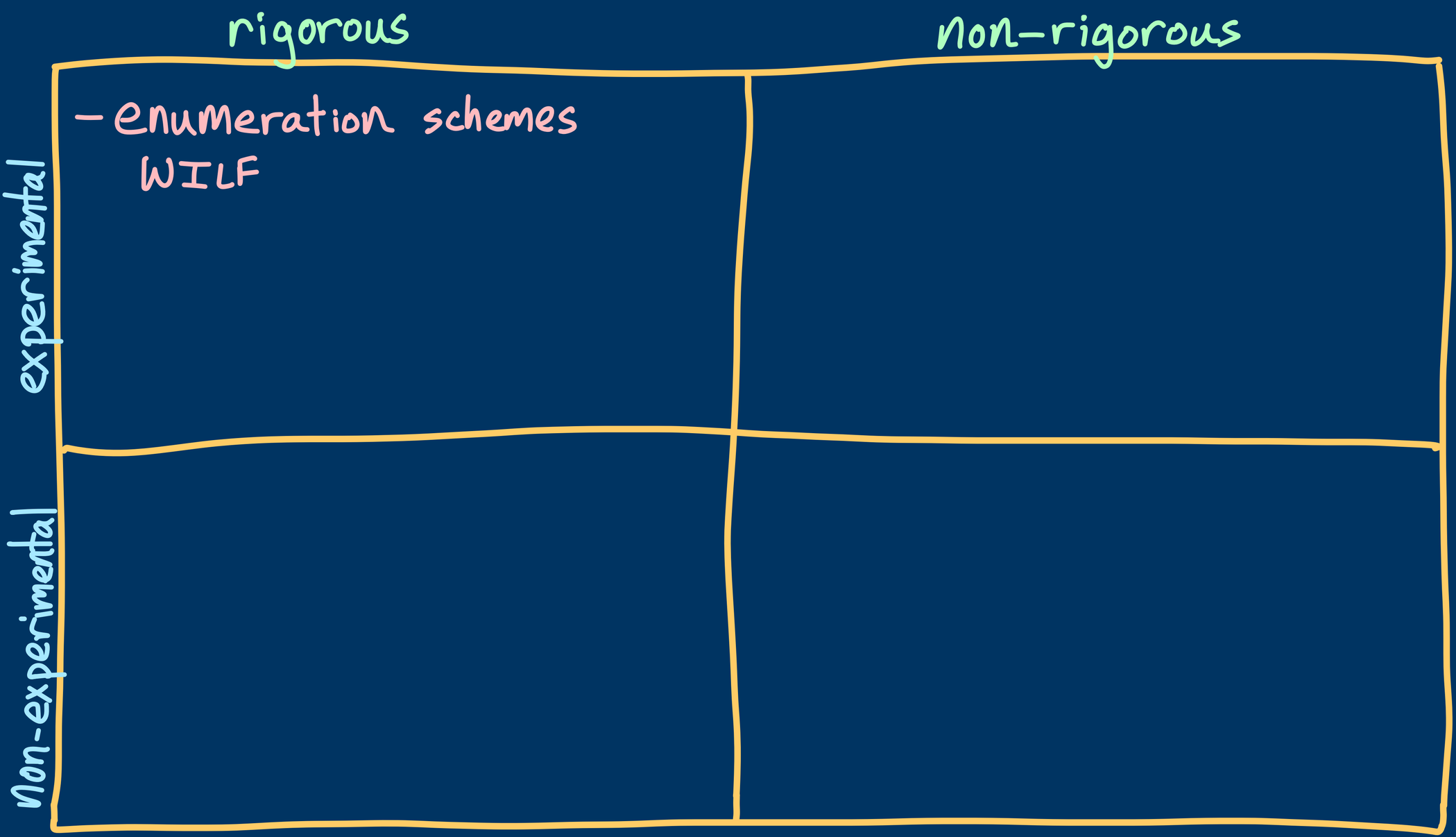
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

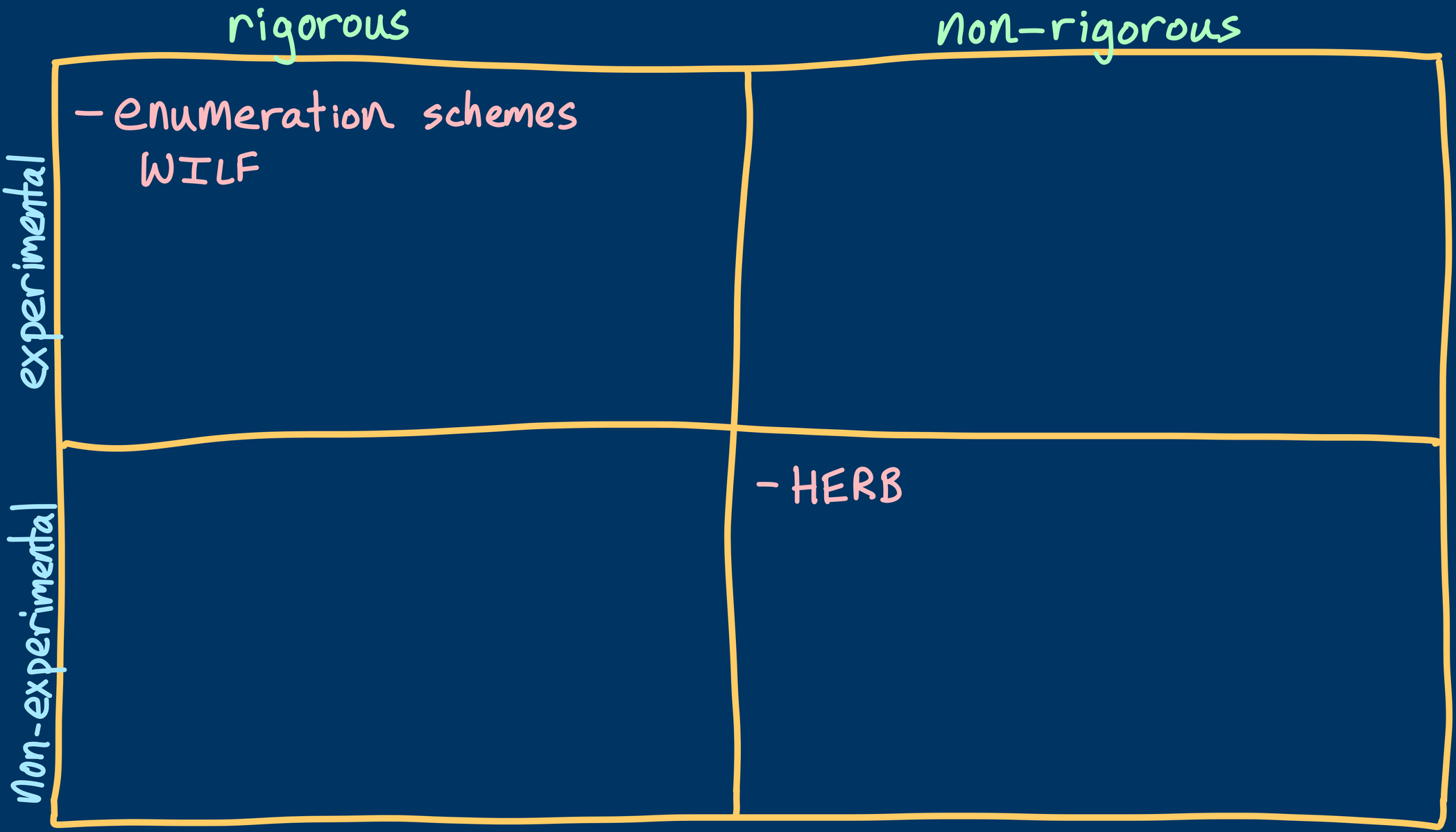
Received May 27, 1998

Experimental: when you "hit go", you don't know whether or not it will return an answer

Rigorous: if it does give an answer, it's guaranteed to be correct


rigorous
non-rigorous
experimental
non-experimental

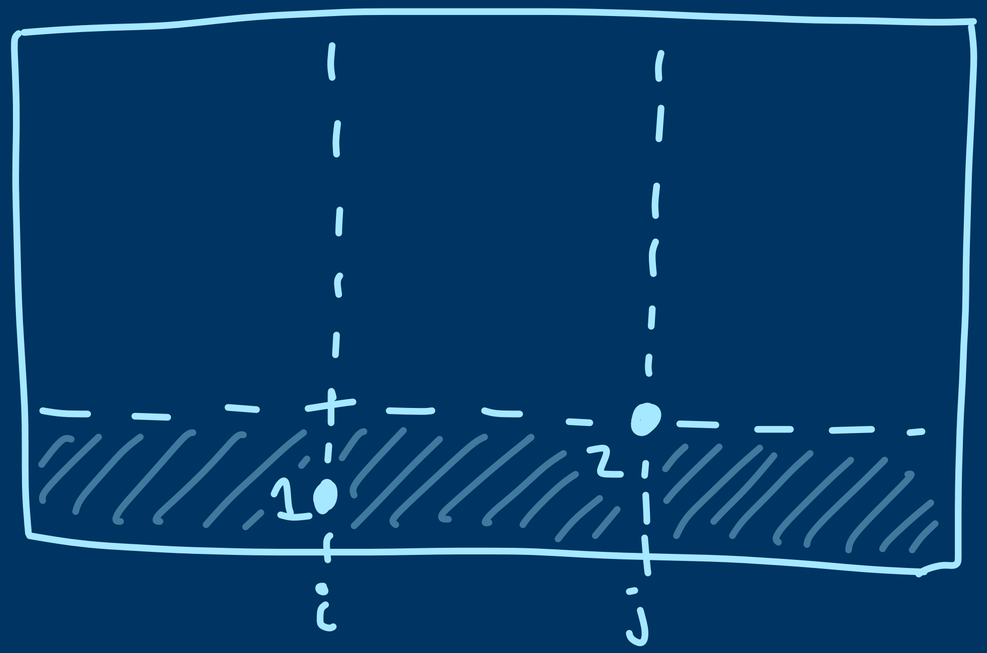

rigorous
non-rigorous
experimental
non-experimental
- enumeration schemes
WILF


rigorous
non-rigorous
experimental
non-experimental
- enumeration schemes
WILF
- HERB

Enumeration Schemes — WILFPLUS

In 2007, Vince Vatter made the method more powerful by increasing the number of situations in which a point can be declared reversibly deletable.

Av(1342,1432)



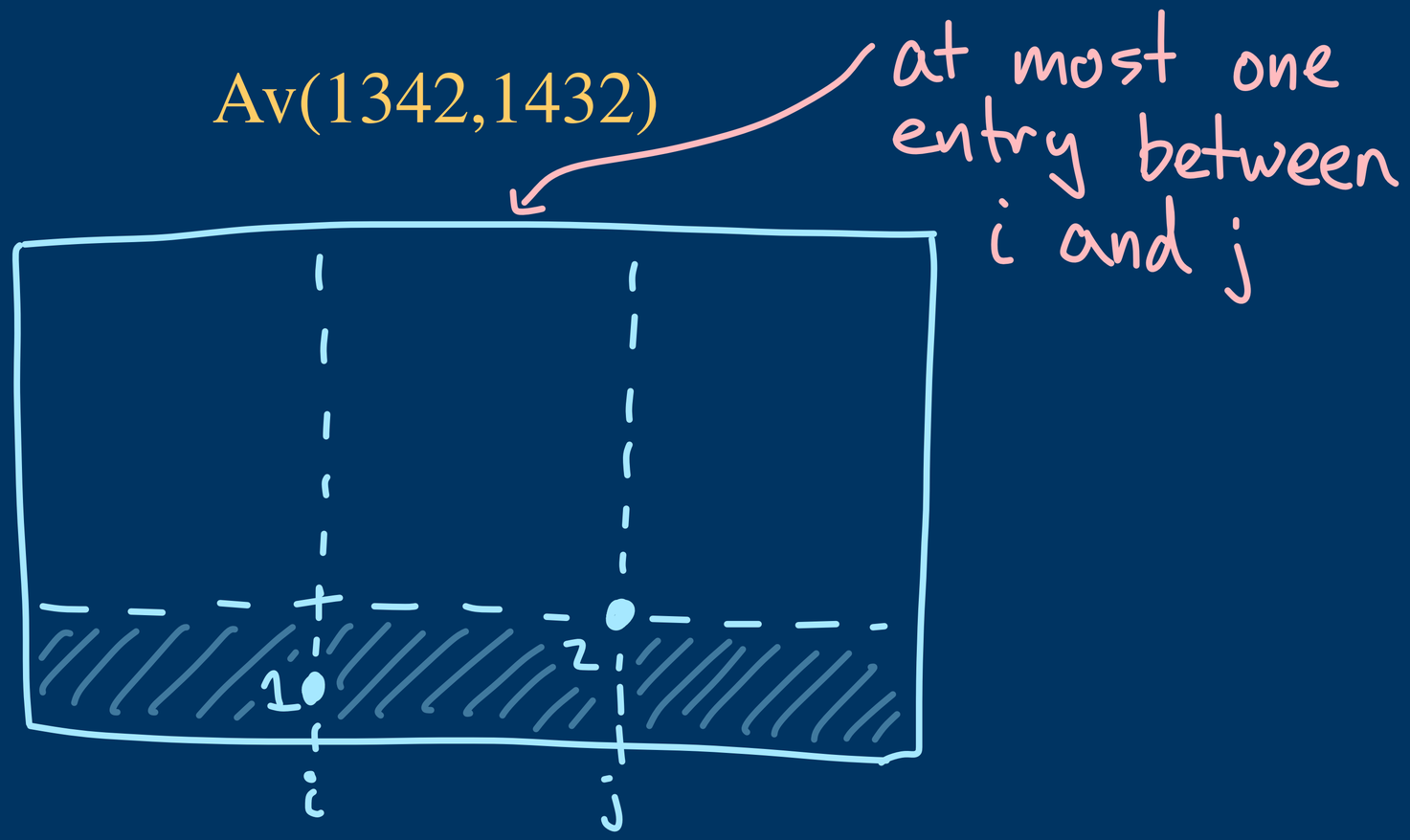
Enumeration Schemes for Restricted Permutations

VINCENT VATTER[1†]

[1]School of Mathematics and Statistics, University of St Andrews
St Andrews, Fife KY19 9SS, UK
(e-mail: vince@mcs.st-and.ac.uk
http://turnbull.mcs.st-and.ac.uk/~vince)

Enumeration Schemes — WILFPLUS

In 2007, Vince Vatter made the method more powerful by increasing the number of situations in which a point can be declared reversibly deletable.



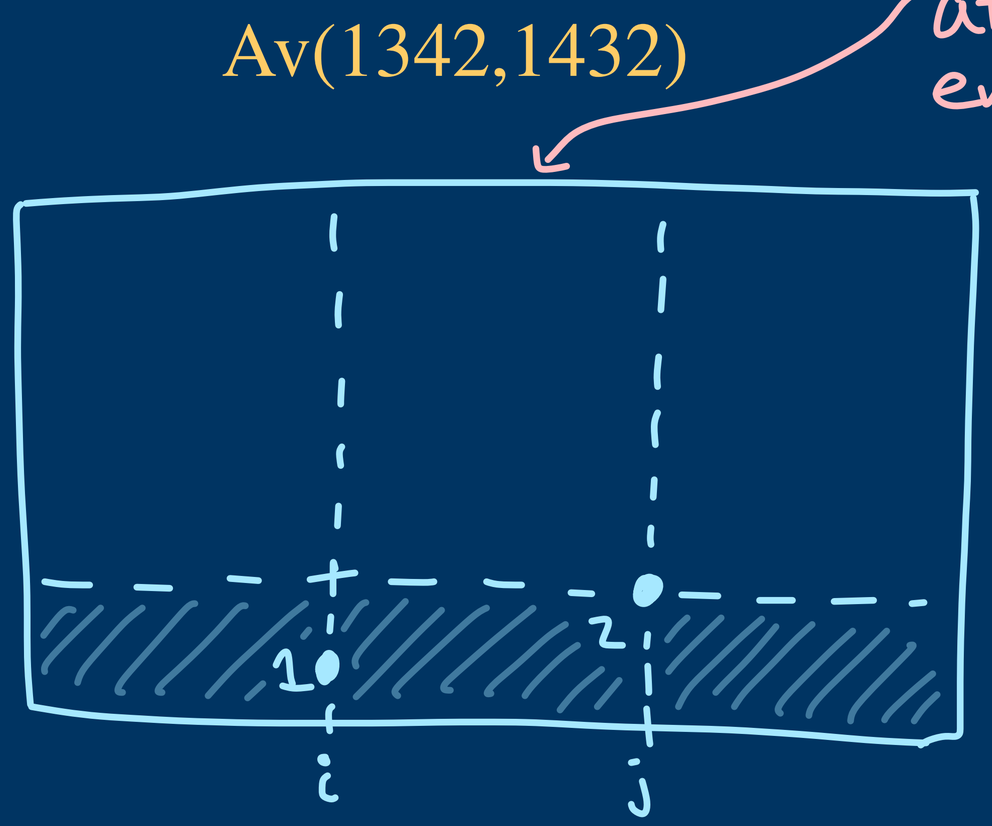
Enumeration Schemes for Restricted Permutations

VINCENT VATTER[1†]

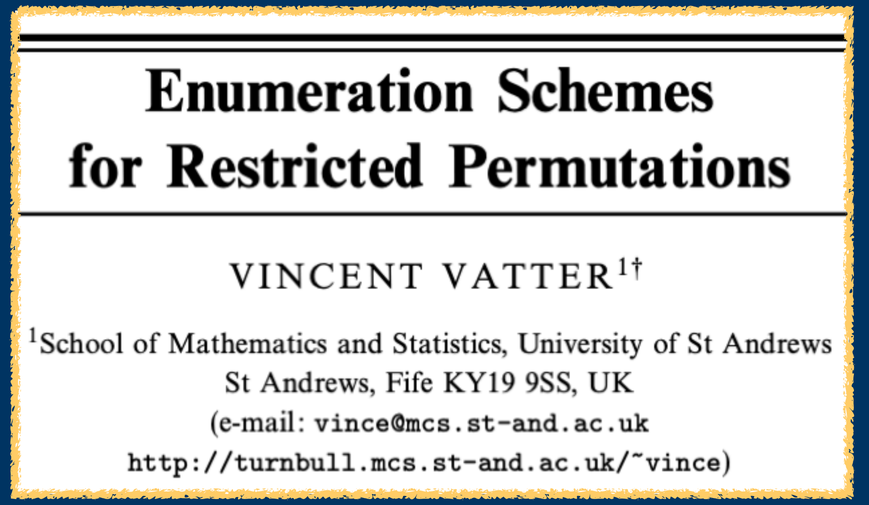
[1]School of Mathematics and Statistics, University of St Andrews
St Andrews, Fife KY19 9SS, UK
(e-mail: vince@mcs.st-and.ac.uk
http://turnbull.mcs.st-and.ac.uk/~vince)

Enumeration Schemes – WILFPLUS

In 2007, Vince Vatter made the method more powerful by increasing the number of situations in which a point can be declared reversibly deletable.



Knowing this: the entry $\pi(j)$ can be deleted.



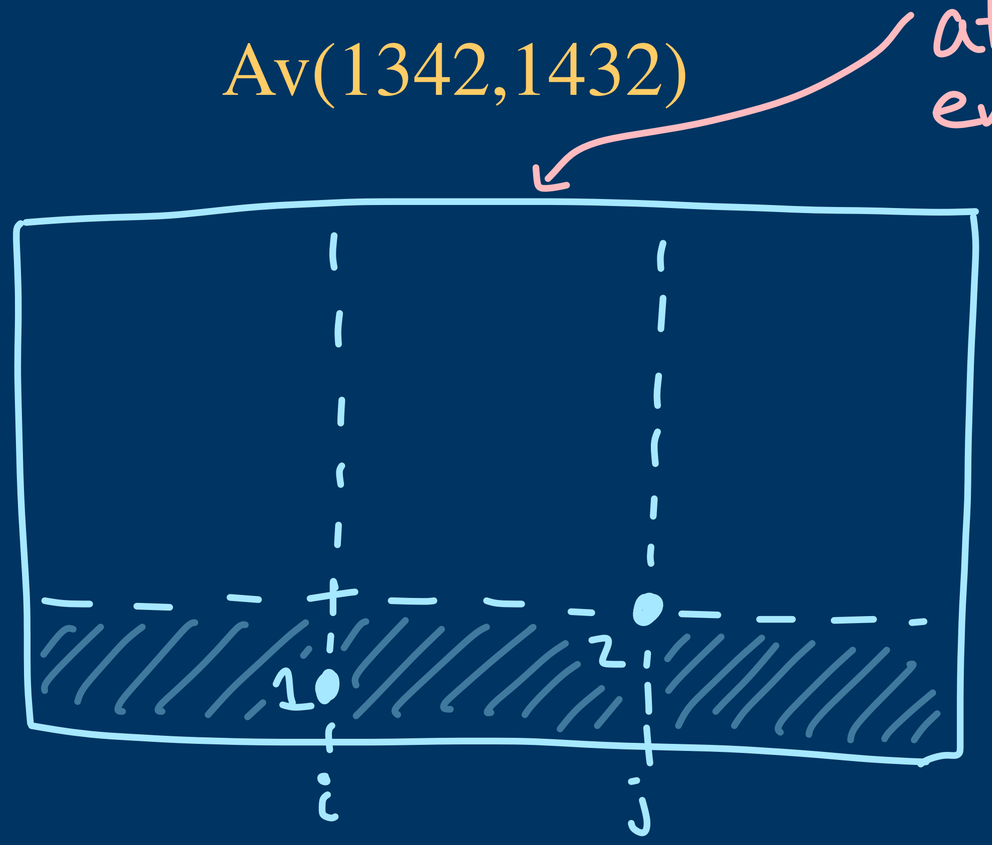
Enumeration Schemes for Restricted Permutations

VINCENT VATTER[1†]

[1]School of Mathematics and Statistics, University of St Andrews
St Andrews, Fife KY19 9SS, UK
(e-mail: vince@mcs.st-and.ac.uk
http://turnbull.mcs.st-and.ac.uk/~vince)

Enumeration Schemes – WILFPLUS

In 2007, Vince Vatter made the method more powerful by increasing the number of situations in which a point can be declared reversibly deletable.



Enumeration Schemes for Restricted Permutations

VINCENT VATTER[1†]

[1]School of Mathematics and Statistics, University of St Andrews
St Andrews, Fife KY19 9SS, UK
(e-mail: vince@mcs.st-and.ac.uk
http://turnbull.mcs.st-and.ac.uk/~vince)

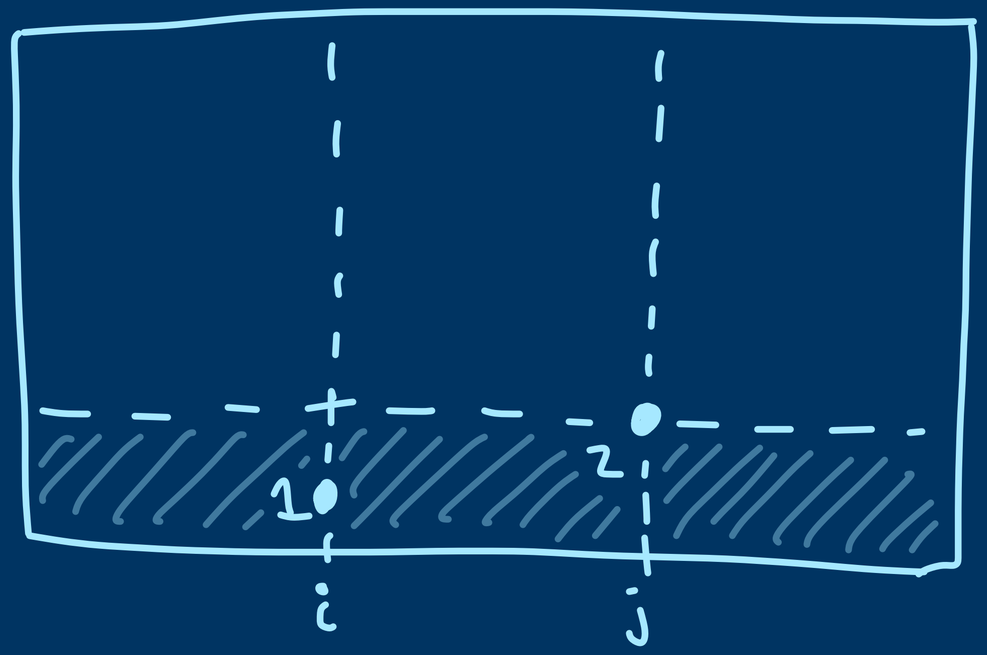
Knowing this: the entry $\pi(j)$ can be deleted.

Zeilberger's "logical reasoning" won't notice this.

Enumeration Schemes — WILFPLUS

In 2007, Vince Vatter made the method more powerful by increasing the number of situations in which a point can be declared reversibly deletable.

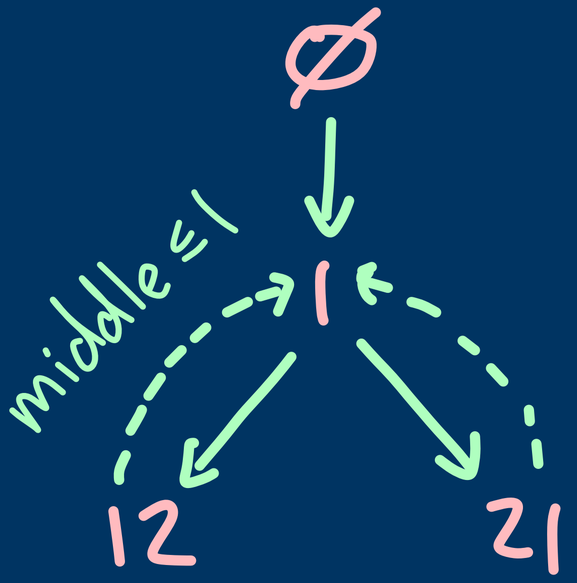
Av(1342,1432)



Enumeration Schemes for Restricted Permutations

VINCENT VATTER[1†]

[1]School of Mathematics and Statistics, University of St Andrews
St Andrews, Fife KY19 9SS, UK
(e-mail: vince@mcs.st-and.ac.uk
http://turnbull.mcs.st-and.ac.uk/~vince)



Enumeration Schemes — WILFPLUS

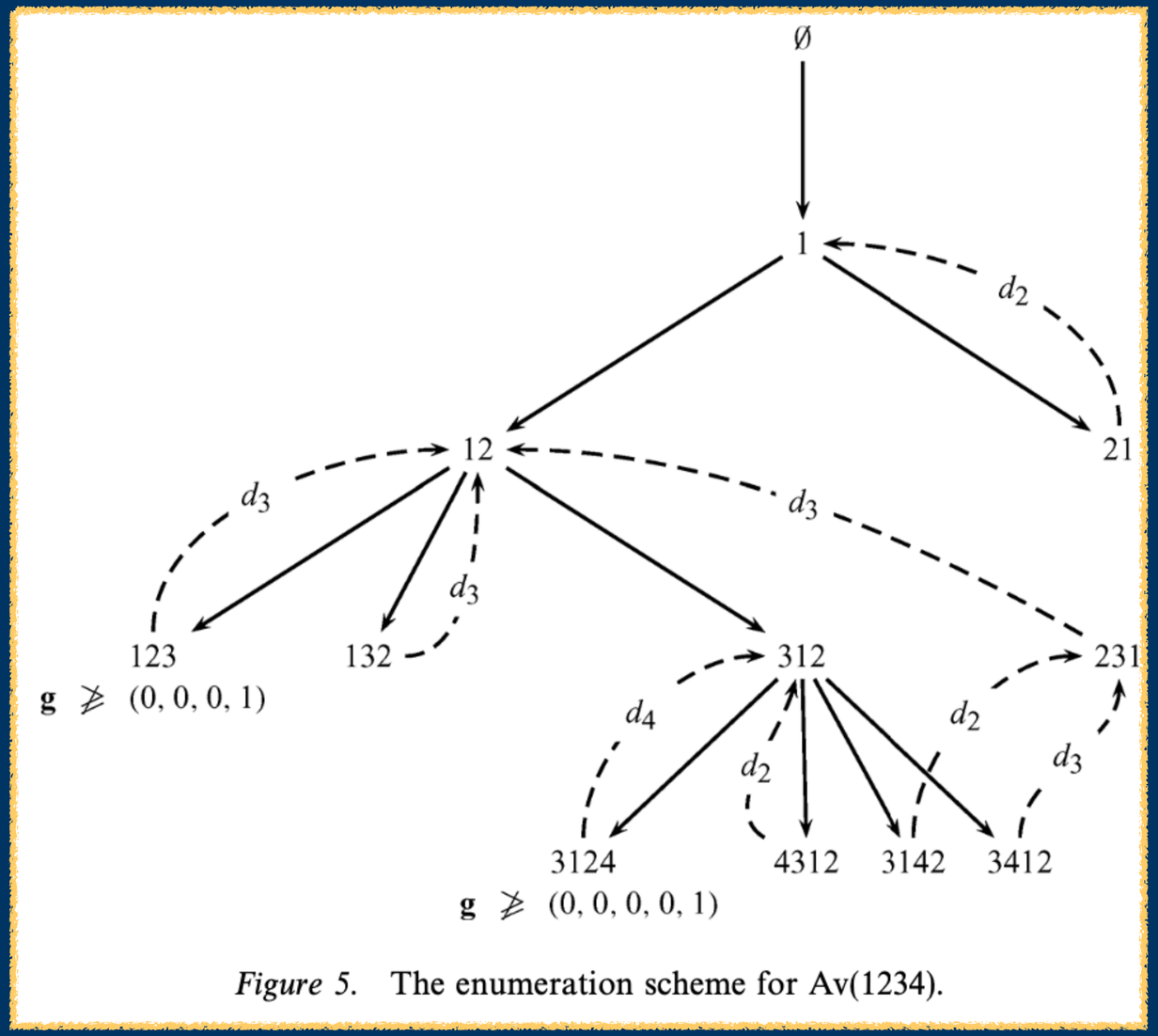


Figure 5. The enumeration scheme for Av(1234).

Enumeration Schemes for Restricted Permutations

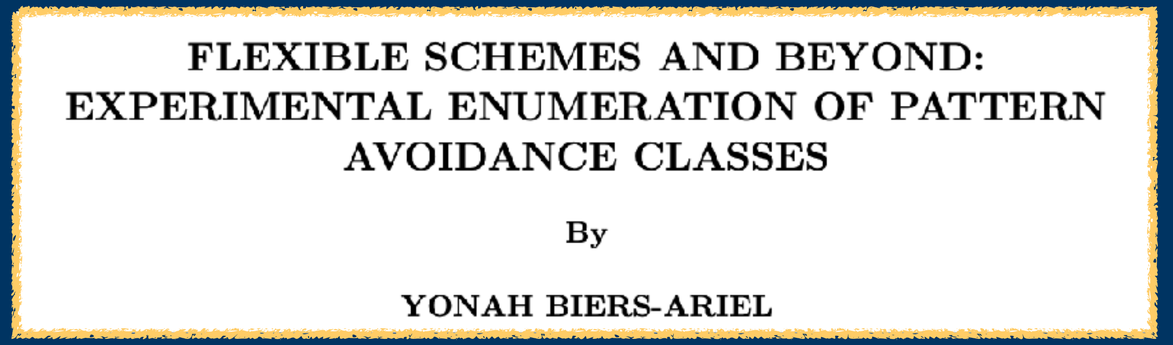
VINCENT VATTER[1†]

[1]School of Mathematics and Statistics, University of St Andrews
St Andrews, Fife KY19 9SS, UK
(e-mail: vince@mcs.st-and.ac.uk
http://turnbull.mcs.st-and.ac.uk/~vince)

Enumeration Schemes — Flexible Schemes

Z: When checking if a point is reversibly deletable, can take into account whether a gap between two entries must be empty.

can do only a few simple classes



FLEXIBLE SCHEMES AND BEYOND:
EXPERIMENTAL ENUMERATION OF PATTERN AVOIDANCE CLASSES

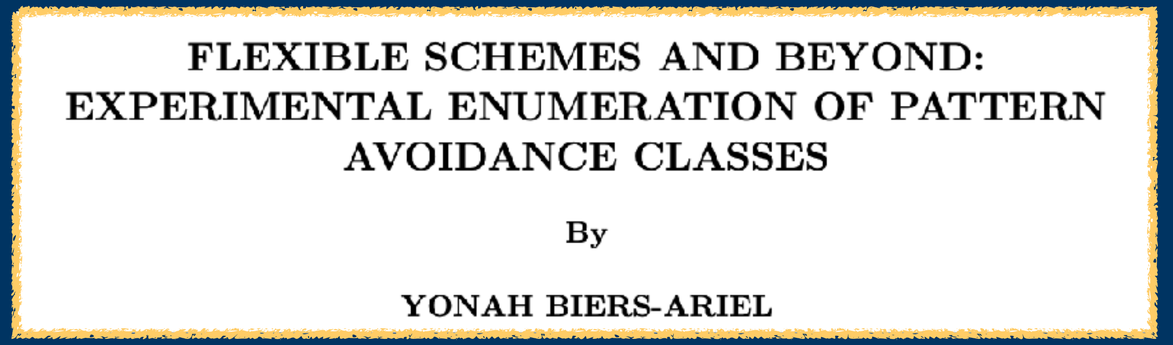
By

YONAH BIERS-ARIEL

Enumeration Schemes — Flexible Schemes

Z: When checking if a point is reversibly deletable, can take into account whether a gap between two entries must be empty.

can do only a few simple classes


FLEXIBLE SCHEMES AND BEYOND:
EXPERIMENTAL ENUMERATION OF PATTERN AVOIDANCE CLASSES

By

YONAH BIERS-ARIEL

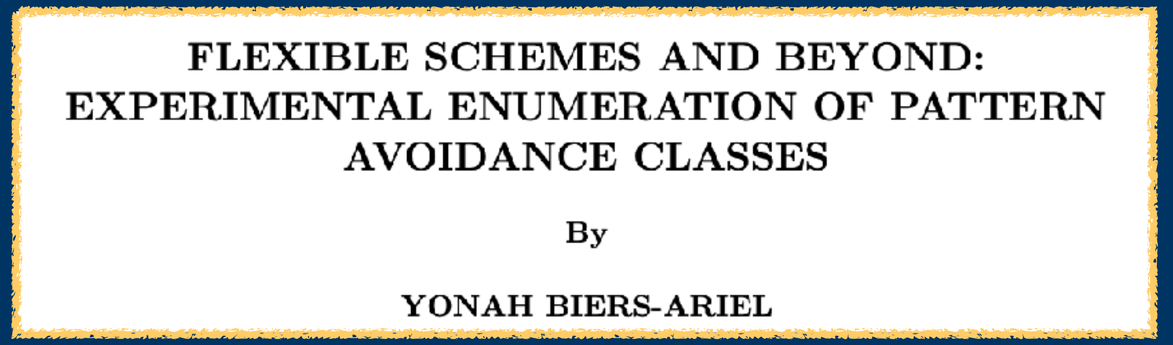
V: Can take into account when a gap is constrained to have a finite number of entries (and more complicated similar constraints)

can do more classes

Enumeration Schemes — Flexible Schemes

Z: When checking if a point is reversibly deletable, can take into account whether a gap between two entries must be empty.

can do only a few simple classes


FLEXIBLE SCHEMES AND BEYOND:
EXPERIMENTAL ENUMERATION OF PATTERN AVOIDANCE CLASSES

By

YONAH BIERS-ARIEL

V: Can take into account when a gap is constrained to have a finite number of entries (and more complicated similar constraints)

can do more classes

B-A: Sometimes if a gap is constrained to a finite number of possibilities, there could be one entry deletable for some of these possibilities, and a *different entry* deletable for the other possibilities.

can do even more classes

Enumeration Schemes — Flexible Schemes

Z: When checking if a point is reversibly deletable, can take into account whether a gap between two entries must be empty.

can do only a few simple classes

FLEXIBLE SCHEMES AND BEYOND: EXPERIMENTAL ENUMERATION OF PATTERN AVOIDANCE CLASSES

By

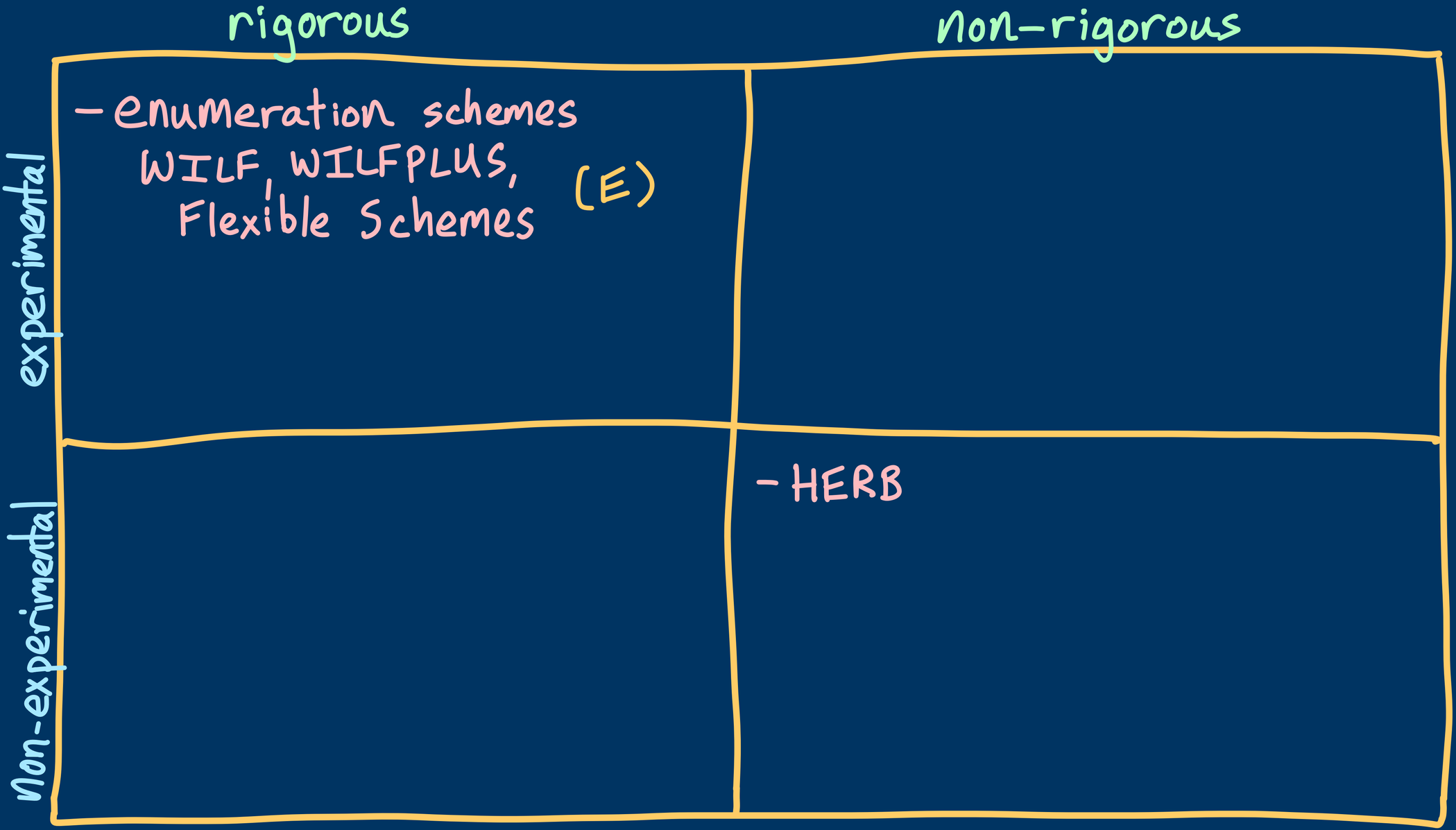
YONAH BIERS-ARIEL

Pat length[1]	Sym Classes[2]	Ins. Enc.	ES	FS	New with FS
[3]	2	0	2	2	0
[4]	7	0	2	2	0
[5]	23	0	2	2	0
[3], [3]	5	5	5	5	0
[4], [4]	56	13	33	44	9
[4], [5]	434	30	112	173	59

V: Ca ned to have a finite number of entri ts)

B-A: Sometimes if a gap is constrained to a finite number of possibilities, there could be one entry deletable for some of these possibilities, and a *different entry* deletable for the other possibilities.

can do even more classes


rigorous
non-rigorous
experimental
non-experimental
- enumeration schemes
WILF, WILFPLUS,
Flexible Schemes
(E)
- HERB

rigorous

non-rigorous

- enumeration schemes

Enumeration schemes for vincular patterns

Andrew M. Baxter [a,1], Lara K. Pudwell [b,*]

[a] Mathematics Department, Pennsylvania State University, State College, PA 08902, United States
[b] Department of Mathematics and Computer Science, Valparaiso University, Valparaiso, IN 46383, United States

exp

non-experimental

- HERB

rigorous

non-rigorous

- enumeration schemes

Enumeration schemes for vincular patterns

Andrew M. Baxter [a,1], Lara K. Pudwell [b,*]

[a] Mathematics Department, Pennsylvania State University, State College, PA 08902, United States
[b] Department of Mathematics and Computer Science, Valparaiso University, Valparaiso, IN 46383, United States

exp

non-experimental

Refining enumeration schemes to count according to permutation statistics

Andrew M. Baxter

Department of Mathematics
Pennsylvania State University
Pennsylvania , U.S.A.

baxter@math.psu.edu

rigorous

non-rigorous

- enumeration schemes

Enumeration schemes for vincular patterns

Andrew M. Baxter [a,1], Lara K. Pudwell [b,*]

[a] Mathematics Department, Pennsylvania State University, State College, PA 08902, United States
[b] Department of Mathematics and Computer Science, Valparaiso University, Valparaiso, IN 46383, United States

exp

non-experimental

Enumeration Schemes for Permutations Avoiding Barred Patterns

Lara Pudwell*

Department of Mathematics and Computer Science
Valparaiso University, Valparaiso, IN 46383

Lara.Pudwell@valpo.edu

Refining enumeration schemes to count according to permutation statistics

Andrew M. Baxter

Department of Mathematics
Pennsylvania State University
Pennsylvania , U.S.A.

baxter@math.psu.edu

Generating Trees

- A “generating tree” for a set of permutations is a way of rigorously representing its structure. It describes where new maximum entries can be inserted into permutations so that they remain in the set.

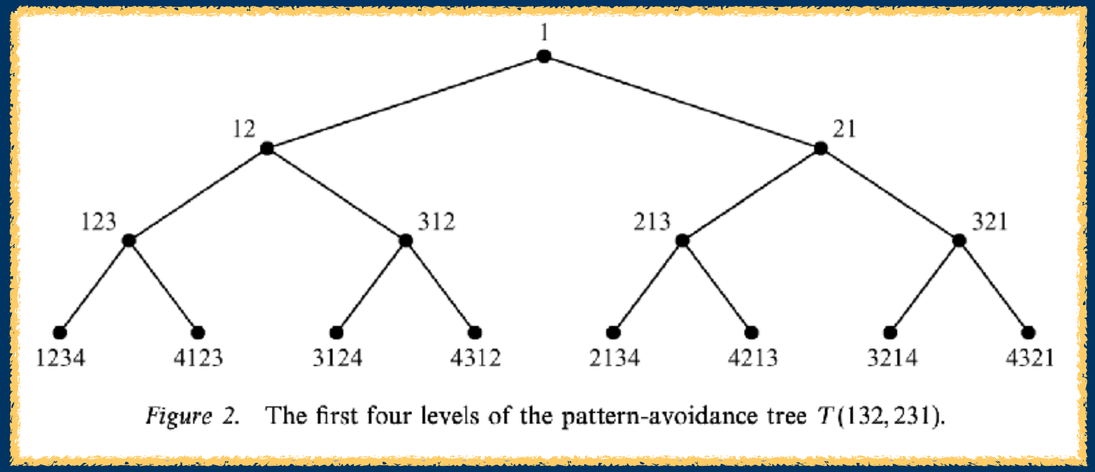


Figure 2. The first four levels of the pattern-avoidance tree $T(132, 231)$.

(Vatter 2007)

Generating Trees

- 1978: Chung, Graham, Hoggatt Jr., and Kleiman invented generating trees to enumerate the *Baxter permutations*.

Generating Trees

- 1978: Chung, Graham, Hoggatt Jr., and Kleiman invented generating trees to enumerate the *Baxter permutations*.

- 1995/1996: West uses generating trees to enumerate several permutation classes.

Generating Trees

- 1978: Chung, Graham, Hoggatt Jr., and Kleiman invented generating trees to enumerate the *Baxter permutations*.

- 1995/1996: West uses generating trees to enumerate several permutation classes.

- 2006: Vatter categorizes the permutation classes that have finitely labeled generating trees and writes the Maple package FINLABEL to enumerate them automatically.

Generating Trees

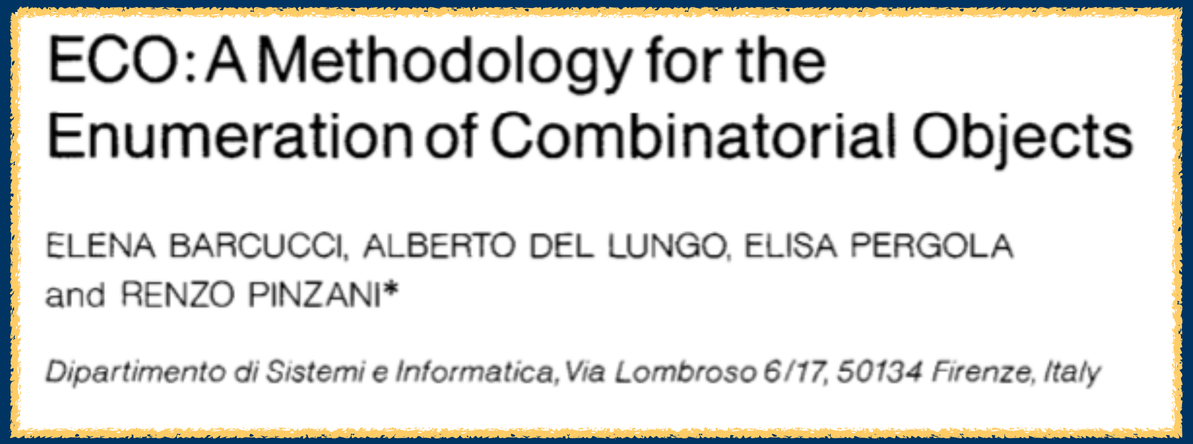
- 1998 — present: ECO Method

  Exports the idea of generating trees to other combinatorial objects and uses them to do many things: enumeration, generating functions, exhaustively generating all objects in a fast way, ...

Generating Trees

- 1998 — present: ECO Method

  Exports the idea of generating trees to other combinatorial objects and uses them to do many things: enumeration, generating functions, exhaustively generating all objects in a fast way, ...



ECO: A Methodology for the Enumeration of Combinatorial Objects

ELENA BARCUCCI, ALBERTO DEL LUNGO, ELISA PERGOLA and RENZO PINZANI*

Dipartimento di Sistemi e Informatica, Via Lombroso 6/17, 50134 Firenze, Italy

Generating Trees

- 199

Some applications arising from the interactions between the theory of Catalan-like numbers and the ECO method*

Luca Ferrari† Elisa Pergola‡ Renzo Pinzani‡
Simone Rinaldi†

trees to other combinatorial objects and uses
eration, generating functions, exhaustively
generating all objects in a fast way, ...

ECO: A Methodology for the Enumeration of Combinatorial Objects

ELENA BARCUCCI, ALBERTO DEL LUNGO, ELISA PERGOLA and RENZO PINZANI*

Dipartimento di Sistemi e Informatica, Via Lombroso 6/17, 50134 Firenze, Italy

Generating Trees

- 199

Some applications arising from the interactions between the theory of Catalan-like numbers and the ECO method*

Luca Ferrari† Elisa Pergola‡ Renzo Pinzani‡
Simone Rinaldi†

trees to

eration, g

ses

generating all objects in a fast way, ...

Integer Partitions in Discrete Dynamical Models and ECO Method*

Le Manh Ha[1] and Phan Thi Ha Duong[2]

[1] *College of Education, Hue University, 34 Le Loi, Hue, Vietnam*

[2] *Institute of Mathematics, 18 Hoang Quoc Viet Road, 10307 Hanoi, Vietnam*

ECO: A Methodology for the Enumeration of Combinatorial Objects

ELENA BARCUCCI, ALBERTO DEL LUNGO, ELISA PERGOLA and RENZO PINZANI*

Dipartimento di Sistemi e Informatica, Via Lombroso 6/17, 50134 Firenze, Italy

Generating Trees

- 199

Some applications arising from the interactions between the theory of Catalan-like numbers and the ECO method*

Luca Ferrari† Elisa Pergola‡ Renzo Pinzani‡
Simone Rinaldi†

trees to

eration, g

ses

generating all objects in a

Integer Partitions in Discrete Dynamical Models and ECO Method*

Le Manh Ha[1] and Phan Thi Ha Duong[2]

[1] College of Education, Hue University, 34 Le Loi, Hue, Vietnam

tnam

ECO: A M
Enumera

ELENA BARCUC
and RENZO PINZ

Dipartimento di Sistemi e Informatica, Via Lombroso 6/17, 50134 Firenze, Italy

ECO Method and the Exhaustive Generation of Convex Polyominoes

Alberto Del Lungo, Andrea Frosini, and Simone Rinaldi

Dipartimento di Scienze Matematiche ed Informatiche
Via del Capitano, 15, 53100, Siena, Italy
{dellungo,frosini,rinaldi}@unisi.it

Generating Trees

- 199

Some applications arising from the interactions between the theory of Catalan-like numbers and the ECO method*

Luca Ferrari† Elisa Pergola‡ Renzo Pinzani‡
Simone Rinaldi†

trees to

eration, g

ses

generating all objects in a

Integer Partitions in Discrete Dynamical Models and ECO Method*

Le Manh Ha[1] and Phan Thi Ha Duong[2]

[1] College of Education, Hue University, 34 Le Loi, Hue, Vietnam

tnam

ECO-generation for some restricted classes of compositions

Jean-Luc Baril, Phan-Thuan Do

ECO Method and the Exhaustive Generation of Convex Polyominoes

Alberto Del Lungo, Andrea Frosini, and Simone Rinaldi

Dipartimento di Scienze Matematiche ed Informatiche
Via del Capitano, 15, 53100, Siena, Italy
{dellungo,frosini,rinaldi}@unisi.it

and RENZO PINZ

Dipartimento di Sistemi e Informatica, Via Lombroso 6/17, 50134 Firenze, Italy

Generating Trees

- 199

Some applications arising from the interactions between the theory of Catalan-like numbers and the ECO method*

Luca Ferrari† Elisa Pergola‡ Renzo Pinzani‡
Simone Rinaldi†

trees to

ses

generating all objects in a

Integer Partitions in Discrete Dynamical Models and ECO Method*

n Thi Ha Duong[2]
ity, 34 Le Loi, Hue, Vietnam

tnam

Generating involutions, derangements, and relatives by ECO

Vincent Vajnovszki

LE2I – UMR CNRS, Université de Bourgogne, B.P. 47 870, 21078 DIJON-Cedex France.
Email: vvajnov@u-bourgogne.fr

ECO-generation for some restricted classes
compositions

Jean-Luc Baril, Phan-Thuan Do

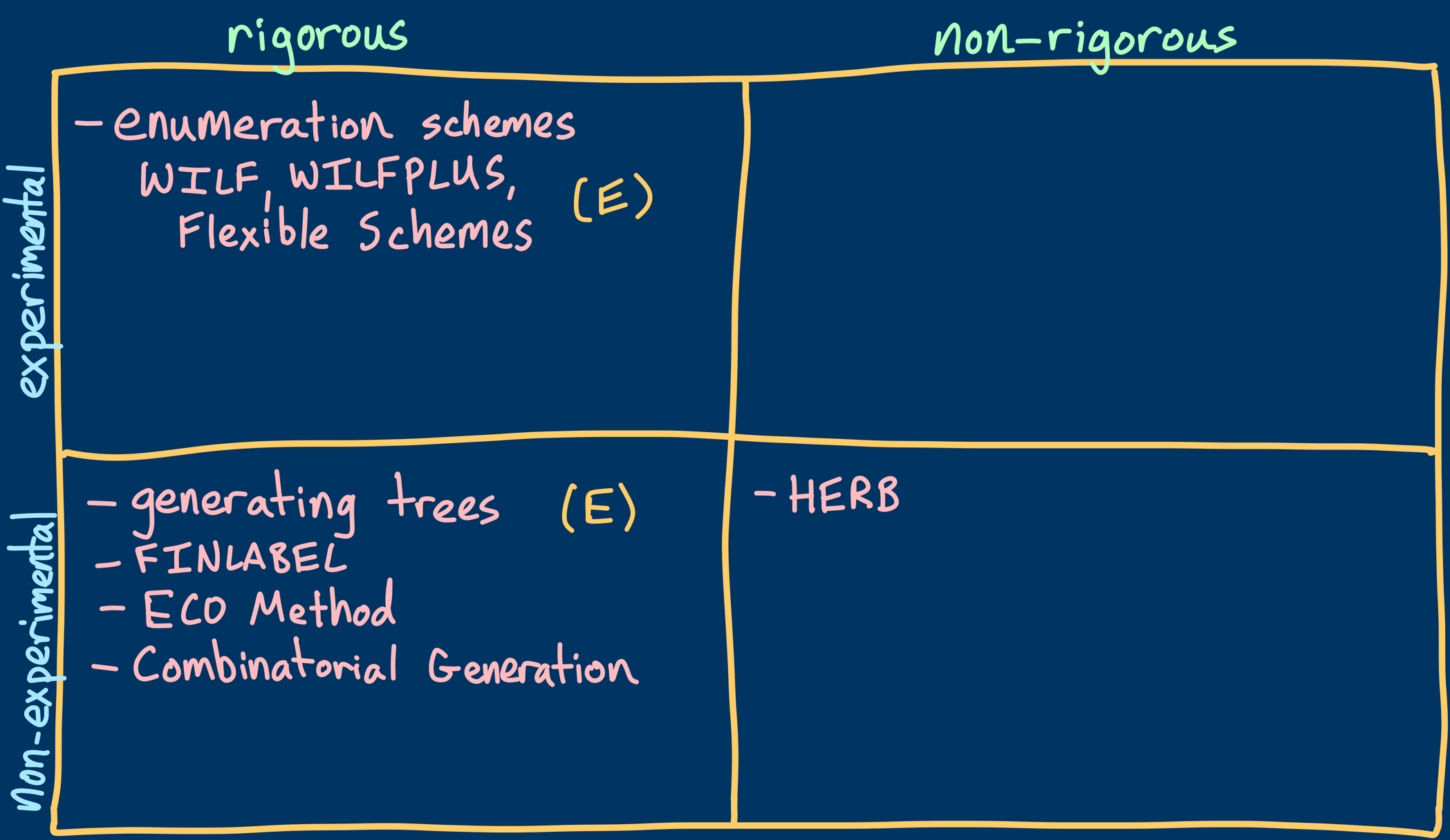
eneration of

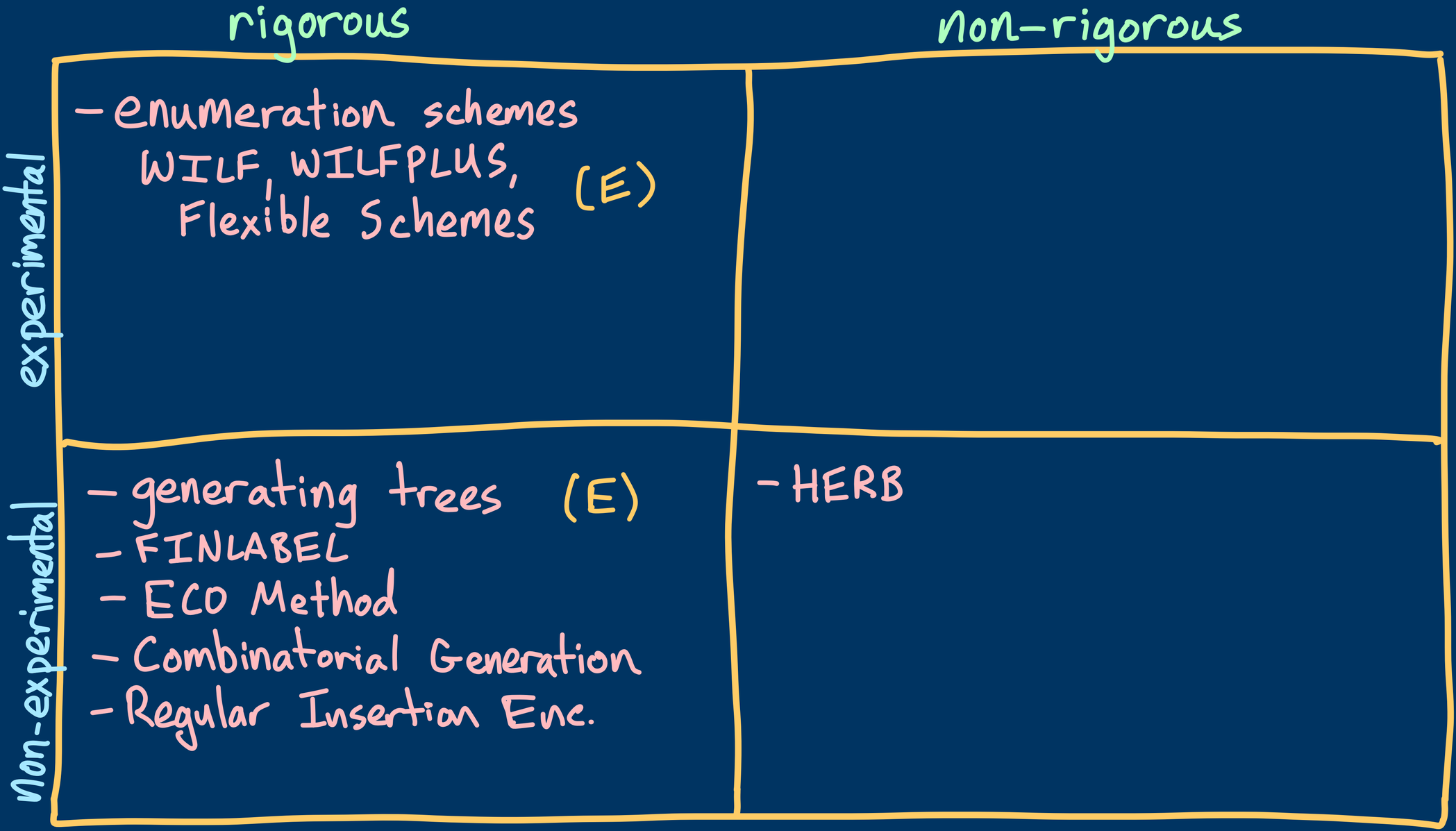
Alberto Del Lungo, Andrea Frosini, and Simone Rinaldi

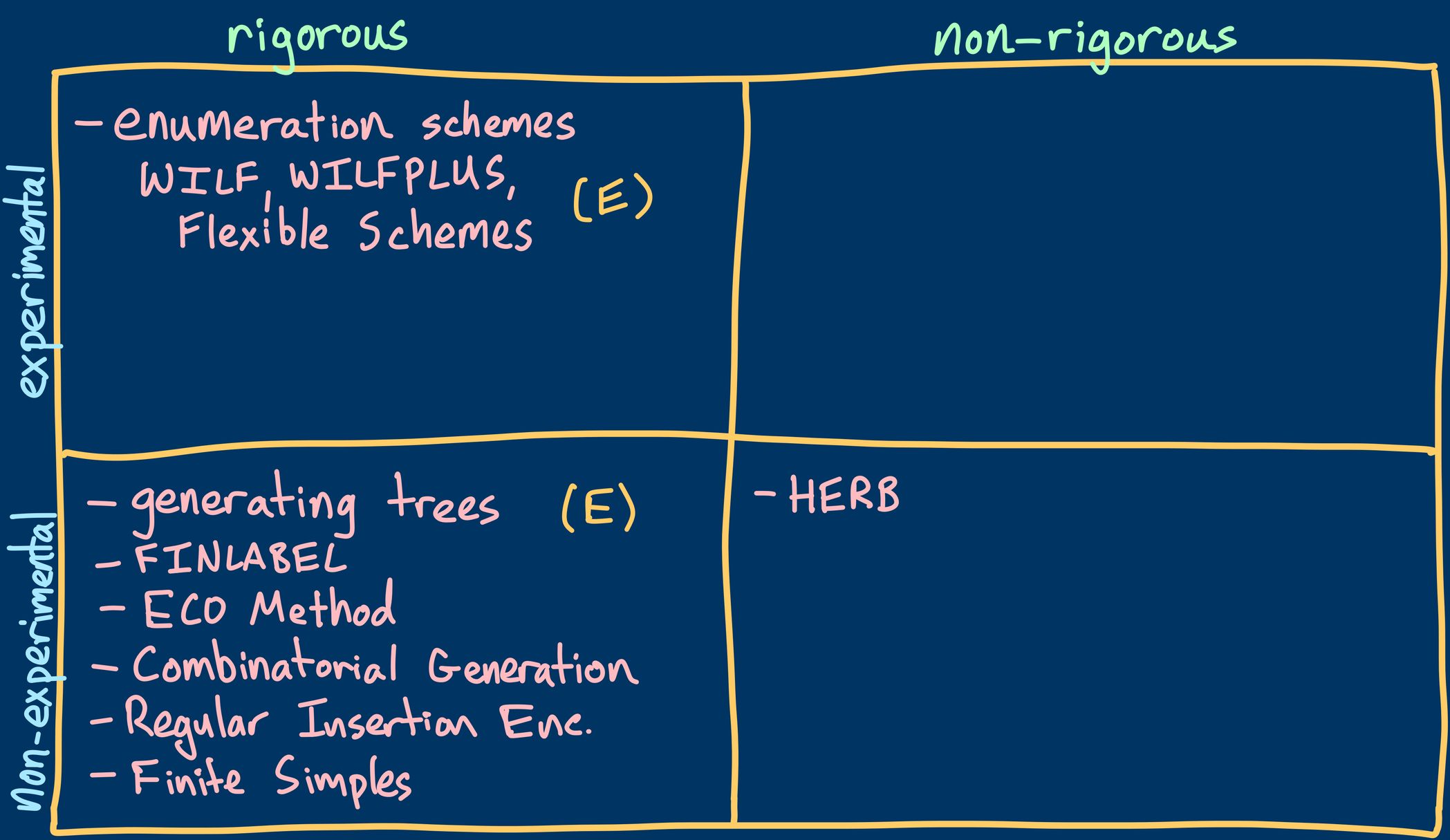
Dipartimento di Scienze Matematiche ed Informatiche
Via del Capitano, 15, 53100, Siena, Italy
{dellungo,frosini,rinaldi}@unisi.it

and RENZO PINZ

Dipartimento di Sistemi e Informatica, Via Lombroso 6/17, 50134 Firenze, Italy


rigorous
non-rigorous
experimental
non-experimental
- enumeration schemes
WILF, WILFPLUS,
Flexible Schemes (E)
- generating trees (E)
- FINLABEL
- ECO Method
- Combinatorial Generation
- HERB


rigorous
non-rigorous
experimental
non-experimental
- enumeration schemes
WILF, WILFPLUS,
Flexible Schemes (E)
- generating trees (E)
- FINLABEL
- ECO Method
- Combinatorial Generation
- Regular Insertion Enc.
- HERB


rigorous
non-rigorous
experimental
non-experimental
- enumeration schemes
WILF, WILFPLUS,
Flexible Schemes (E)
- generating trees (E)
- FINLABEL
- ECO Method
- Combinatorial Generation
- Regular Insertion Enc.
- Finite Simples
- HERB

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E)	
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB

Struct

Most of the methods described so far:

"expand a particular structure tree and hope it ends up being finite"

MATHEMATICS OF COMPUTATION
Volume 88, Number 318, July 2019, Pages 1967–1990
https://doi.org/10.1090/mcom/3386
Article electronically published on December 11, 2018

AUTOMATIC DISCOVERY OF STRUCTURAL RULES OF PERMUTATION CLASSES

CHRISTIAN BEAN, BJARKI GUDMUNDSSON, AND HENNING ULFARSSON

Struct is a software package that takes a permutation class as input and searches for a *set cover* that decomposes it into simpler disjoint parts.

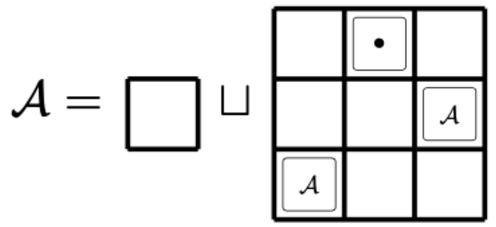


FIGURE 1. The structure of Av(231)

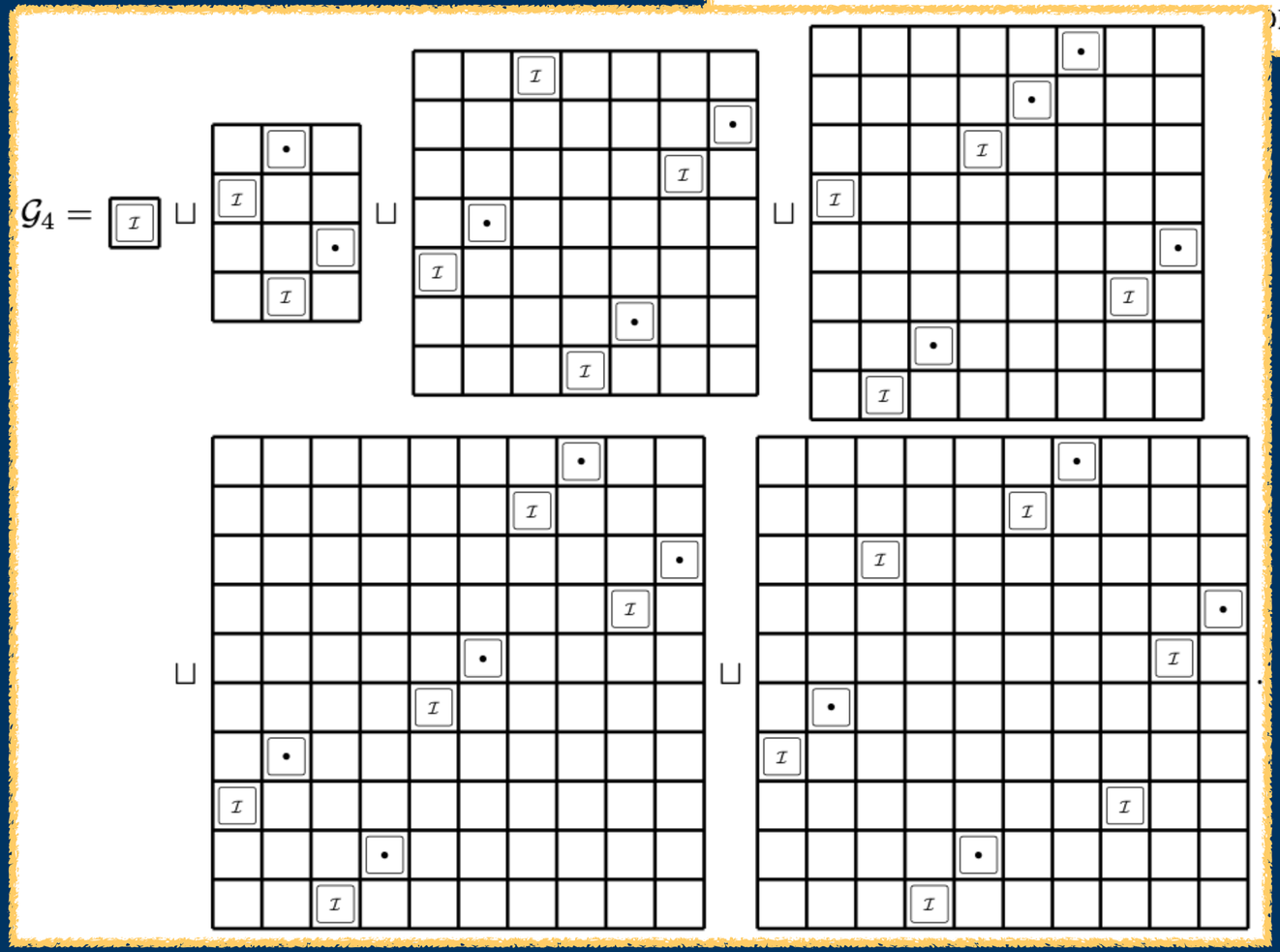
Struct

$\mathcal{G}_4 = \mathrm{Av}(321,1324)$

MATHEMATICS OF COMPUTATION
Volume 88, Number 318, July 2019, Pages 1967–1990
https://doi.org/10.1090/mcom/3386
Article electronically published on December 11, 2018

AUTOMATIC DISCOVERY OF STRUCTURAL RULES OF PERMUTATION CLASSES

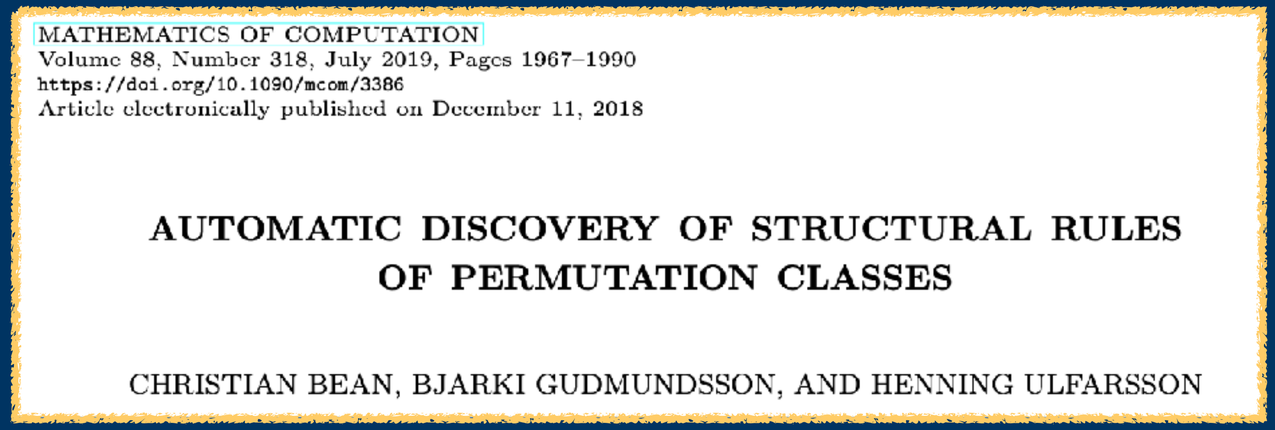
ÓMUNDSSON, AND HENNING ULFARSSON



Struct

Method:

- Construct a big list of grids that make subsets of the input class.



MATHEMATICS OF COMPUTATION
Volume 88, Number 318, July 2019, Pages 1967–1990
https://doi.org/10.1090/mcom/3386
Article electronically published on December 11, 2018

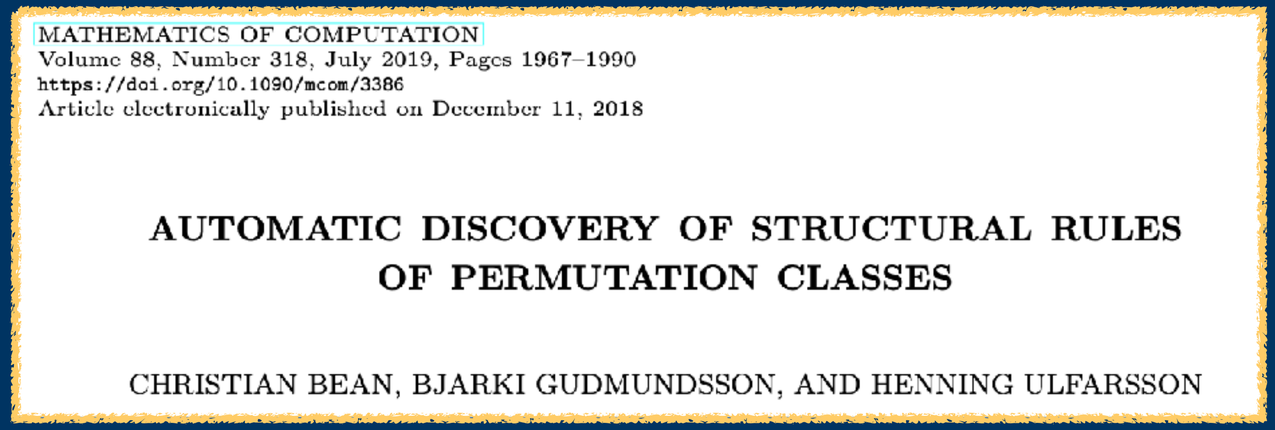
AUTOMATIC DISCOVERY OF STRUCTURAL RULES OF PERMUTATION CLASSES

CHRISTIAN BEAN, BJARKI GUDMUNDSSON, AND HENNING ULFARSSON

Struct

Method:

- Construct a big list of grids that make subsets of the input class.


MATHEMATICS OF COMPUTATION
Volume 88, Number 318, July 2019, Pages 1967–1990
https://doi.org/10.1090/mcom/3386
Article electronically published on December 11, 2018

AUTOMATIC DISCOVERY OF STRUCTURAL RULES OF PERMUTATION CLASSES

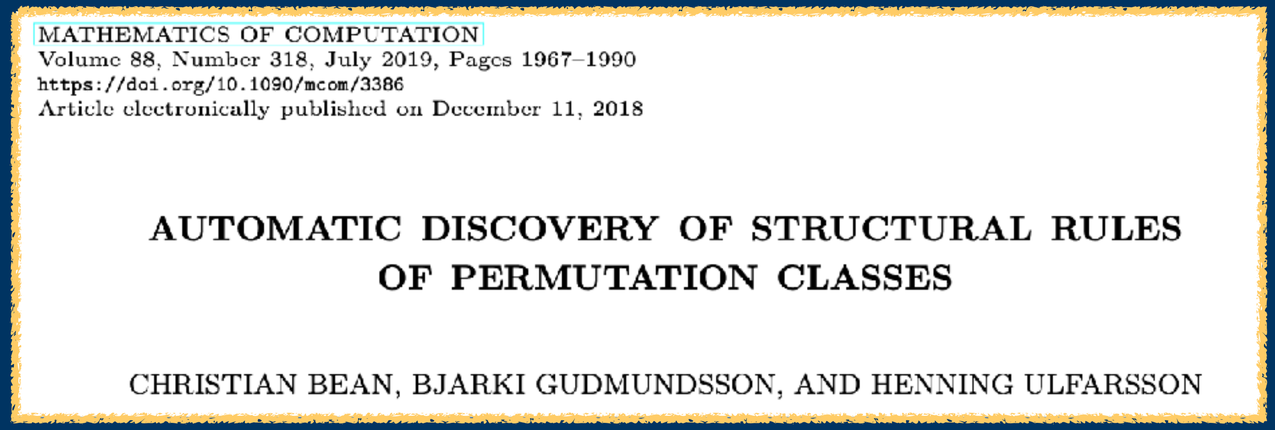
CHRISTIAN BEAN, BJARKI GUDMUNDSSON, AND HENNING ULFARSSON

- Set up an integer linear programming problem to pick a subset of grids that forms a set cover (each permutation in the class gives one constraint).

Struct

Method:

- Construct a big list of grids that make subsets of the input class.


MATHEMATICS OF COMPUTATION
Volume 88, Number 318, July 2019, Pages 1967–1990
https://doi.org/10.1090/mcom/3386
Article electronically published on December 11, 2018

AUTOMATIC DISCOVERY OF STRUCTURAL RULES OF PERMUTATION CLASSES

CHRISTIAN BEAN, BJARKI GUDMUNDSSON, AND HENNING ULFARSSON

- Set up an integer linear programming problem to pick a subset of grids that forms a set cover (each permutation in the class gives one constraint).
- Feed it into an ILP solver like Gurobi and wait patiently for a solution.

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E)	- Struct - BiSC
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB

Combinatorial Exploration

At the end of the Struct paper, the authors discuss some classes that Struct can’t do along with a possible future approach.

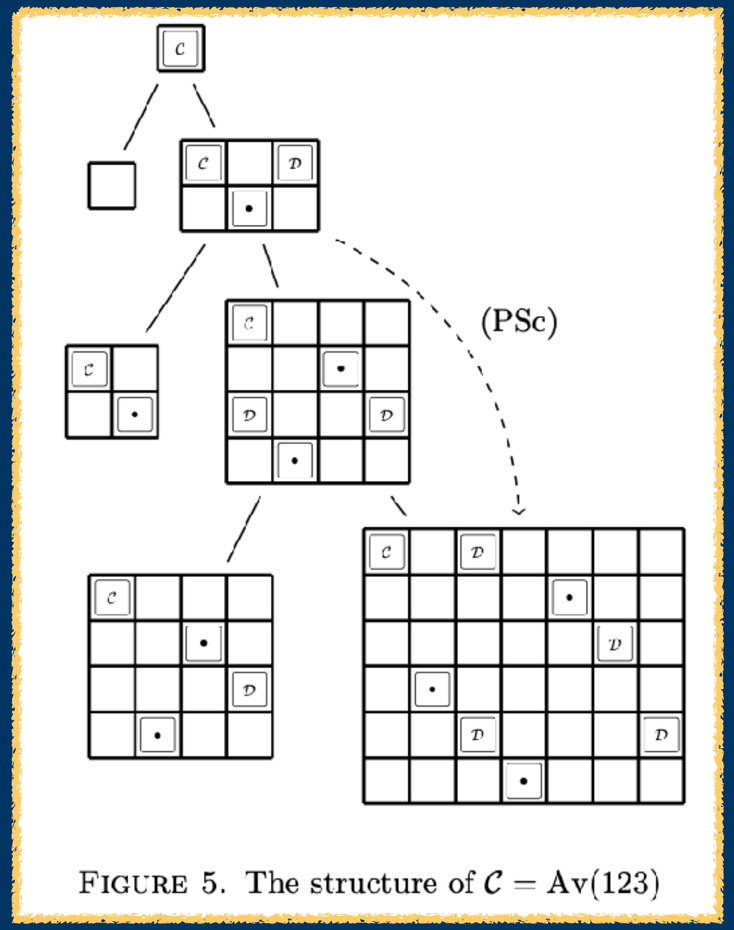


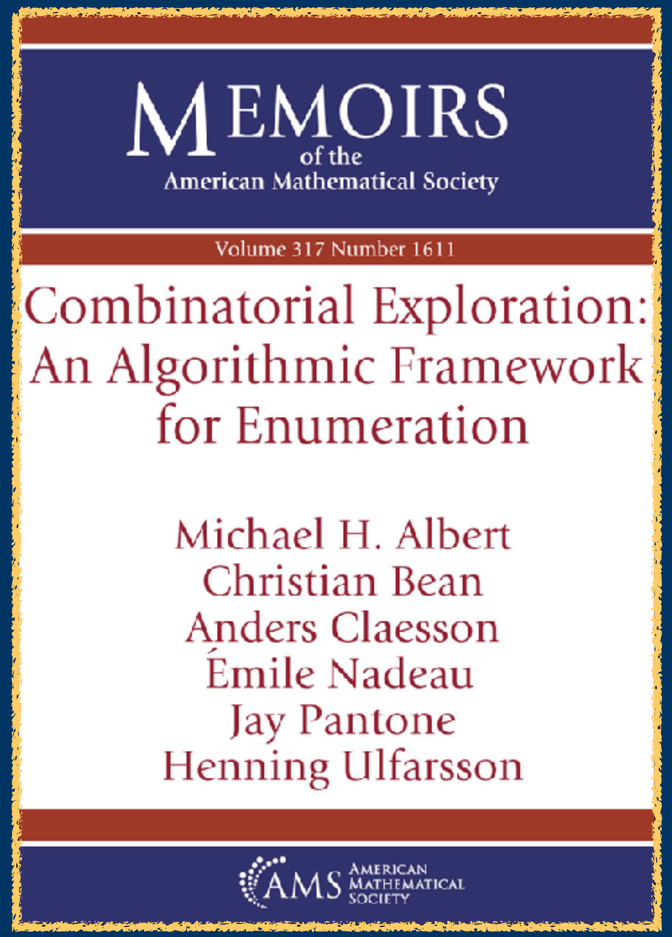
FIGURE 5. The structure of $\mathcal{C} = \mathrm{Av}(123)$

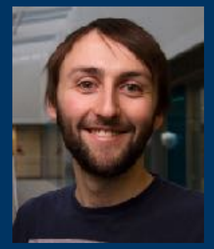
“proof tree”

Combinatorial Exploration

I started talking with Henning Ulfarsson and Christian Bean at PP 2016.

Many years later…





Combinatorial Exploration

Key insights:

1. Instead of expanding one particular structure tree and hoping it ends up being finite: produce a bunch of independent “rules” that relate a parent set to child sets, and hope that some subset of these rules can be assembled into a tree

Combinatorial Exploration

Key insights:

1. Instead of expanding one particular structure tree and hoping it ends up being finite: produce a bunch of independent “rules” that relate a parent set to child sets, and hope that some subset of these rules can be assembled into a tree
2. We need a much more efficient way to represent sets of permutations.

Combinatorial Exploration

Key insights:

1. Instead of expanding one particular structure tree and hoping it ends up being finite: produce a bunch of independent “rules” that relate a parent set to child sets, and hope that some subset of these rules can be assembled into a tree
2. We need a much more efficient way to represent sets of permutations.
3. If (1) and (2) are done correctly, then the result can still be fully rigorous.

C

$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation

factor

$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor

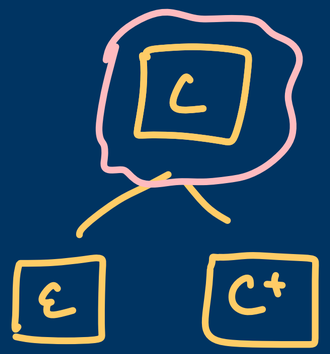
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



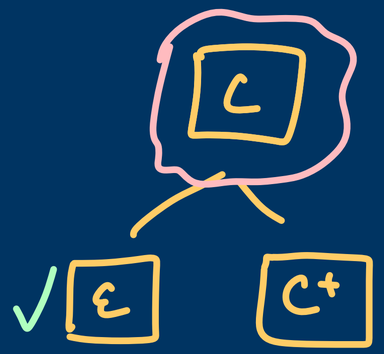
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



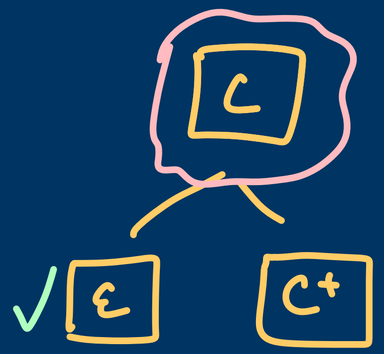
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



$C = Av(132)$

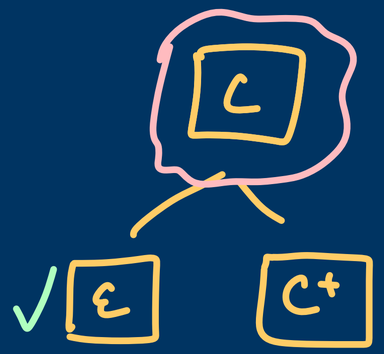
C^+ = nonempty perms

empty or not

point placement

row/col separation

factor



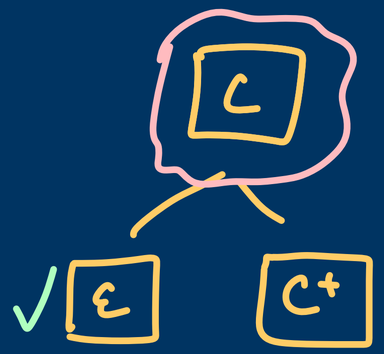
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



$C = Av(132)$

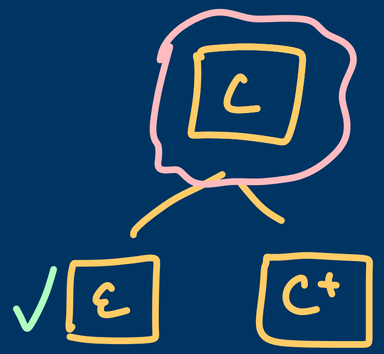
C^+ = nonempty perms

empty or not

point placement

row/col separation

factor



$C = Av(132)$

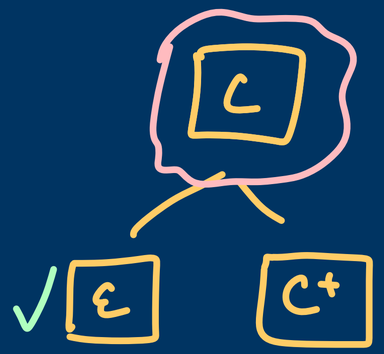
C^+ = nonempty perms

empty or not

point placement

row/col separation

factor



$C = Av(132)$

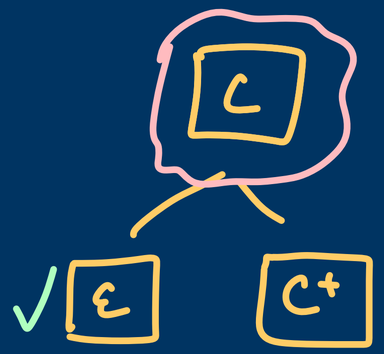
C^+ = nonempty perms

empty or not

point placement

row/col separation

factor



$C = Av(132)$

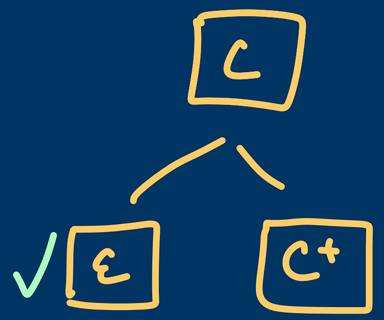
C^+ = nonempty perms

empty or not

point placement

row/col separation

factor



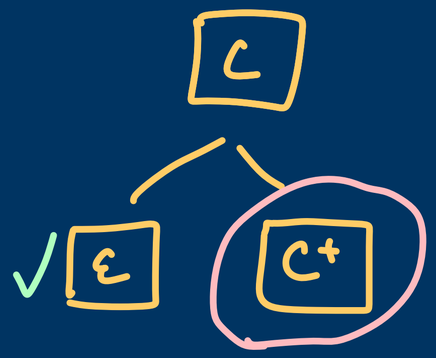
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



$C = Av(132)$

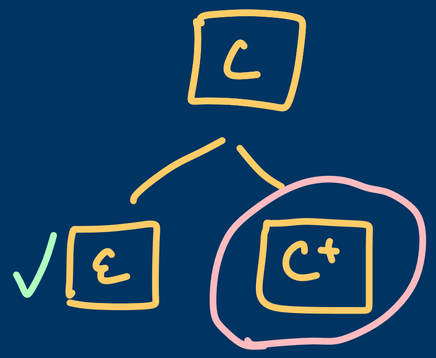
C^+ = nonempty perms

empty or not

point placement

row/col separation

factor



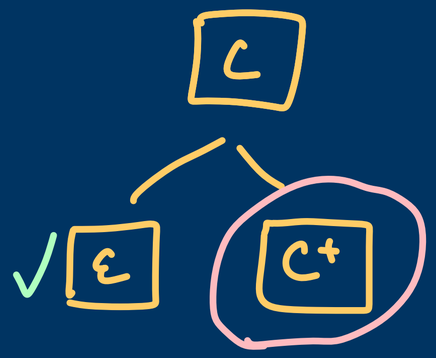
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



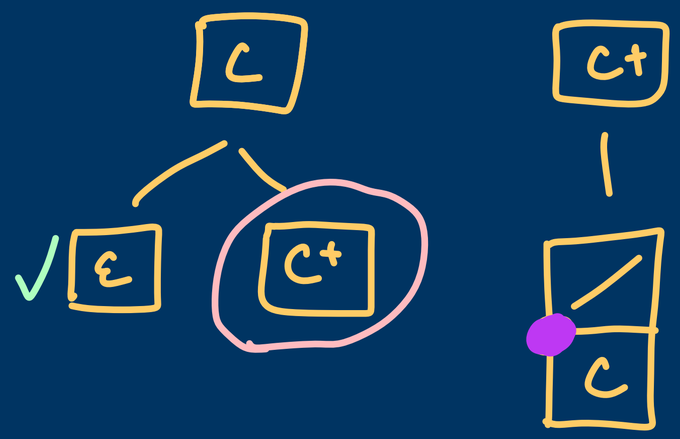
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



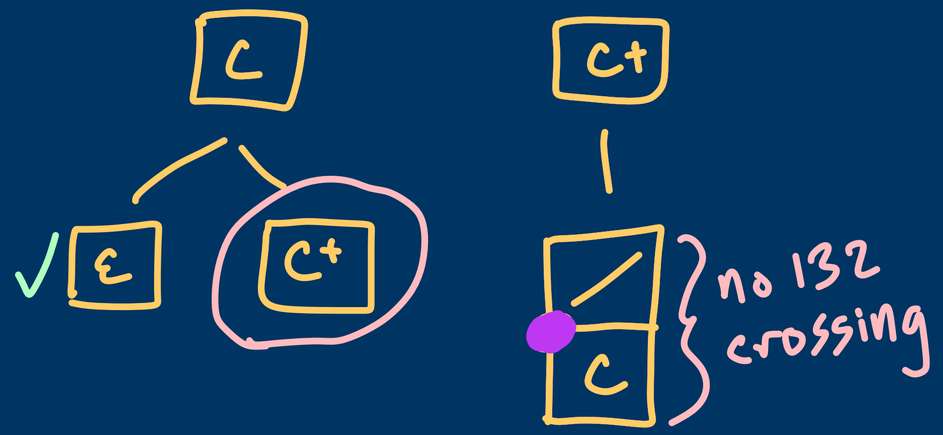
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



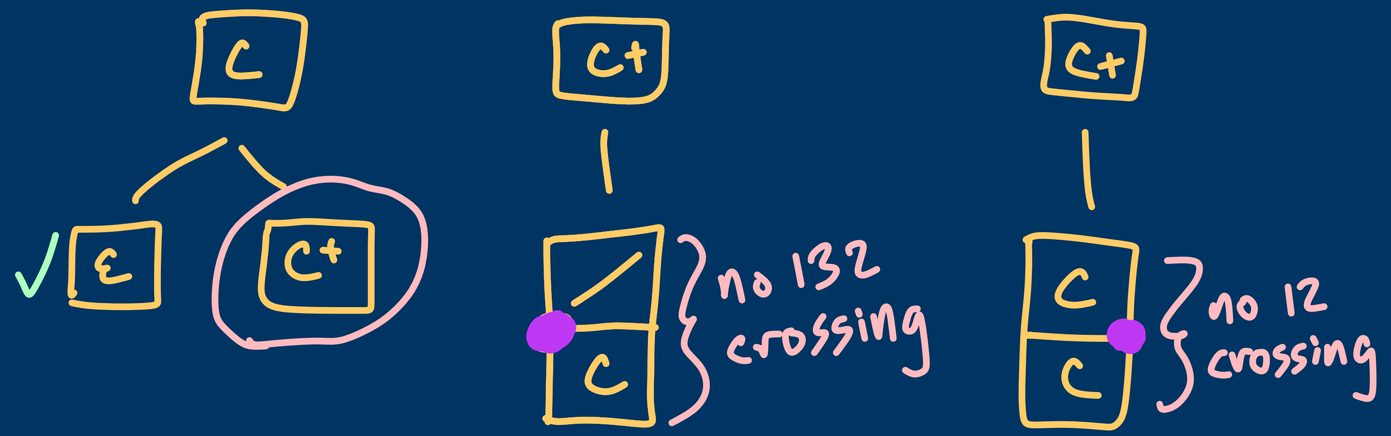
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



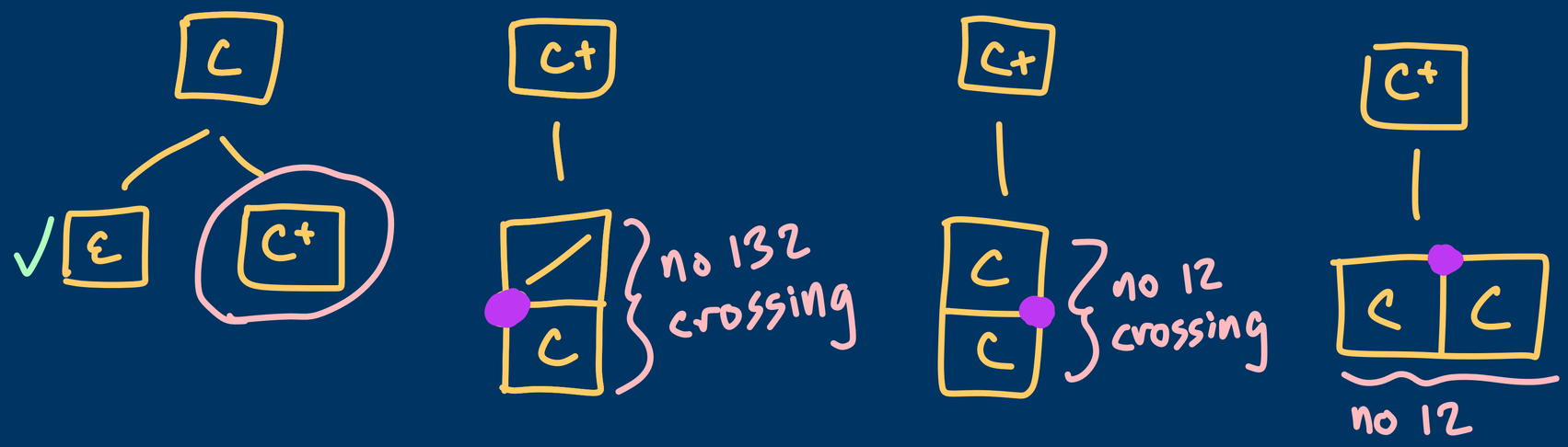
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



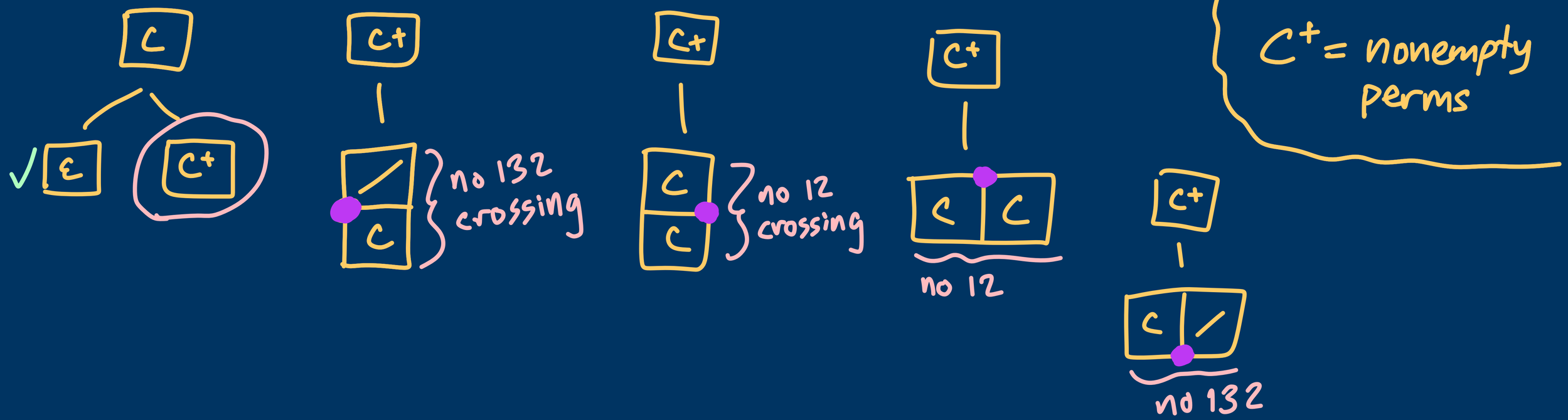
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

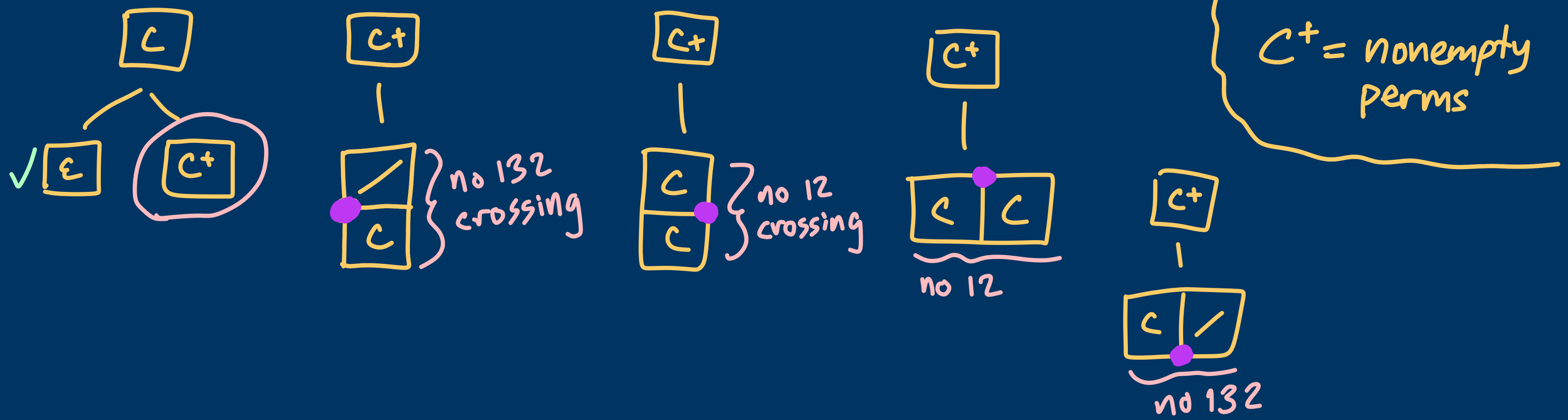
row/col separation factor



empty or not

point placement

row/col separation factor

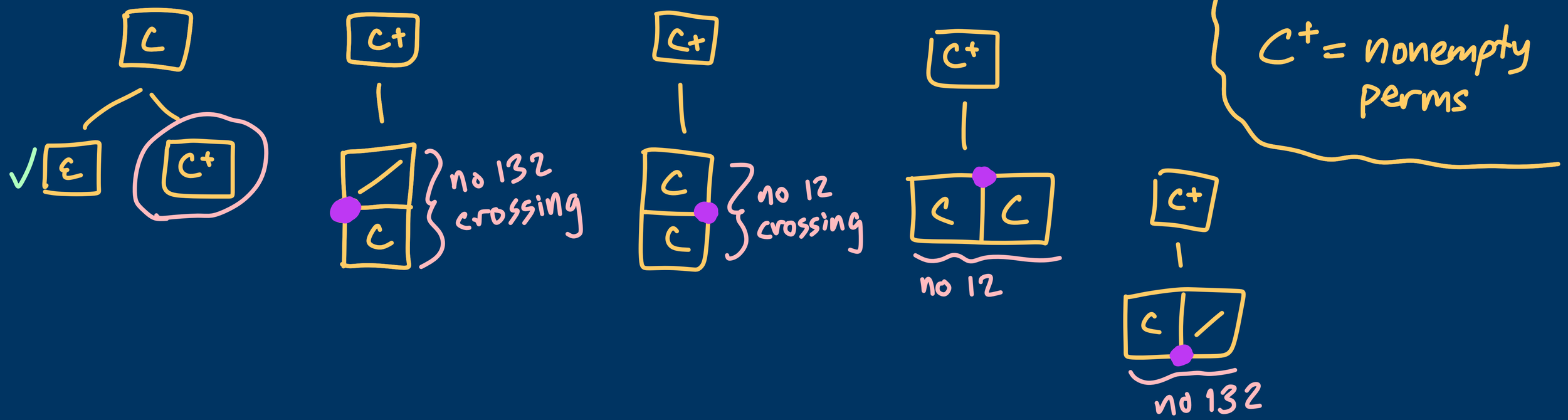


empty or not

point placement

row/col separation

factor

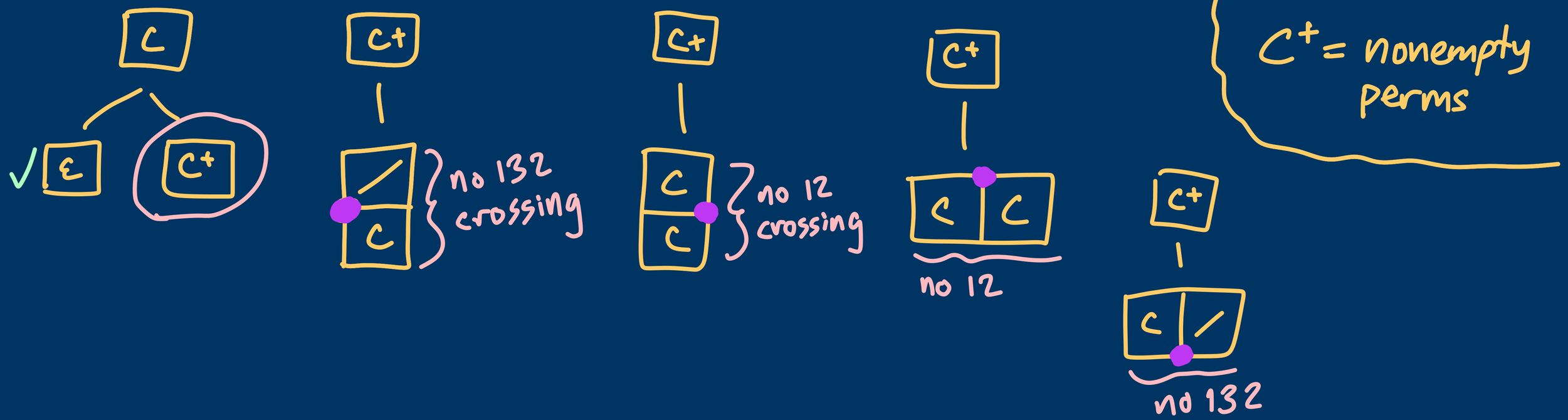


empty or not

point placement

row/col separation

factor

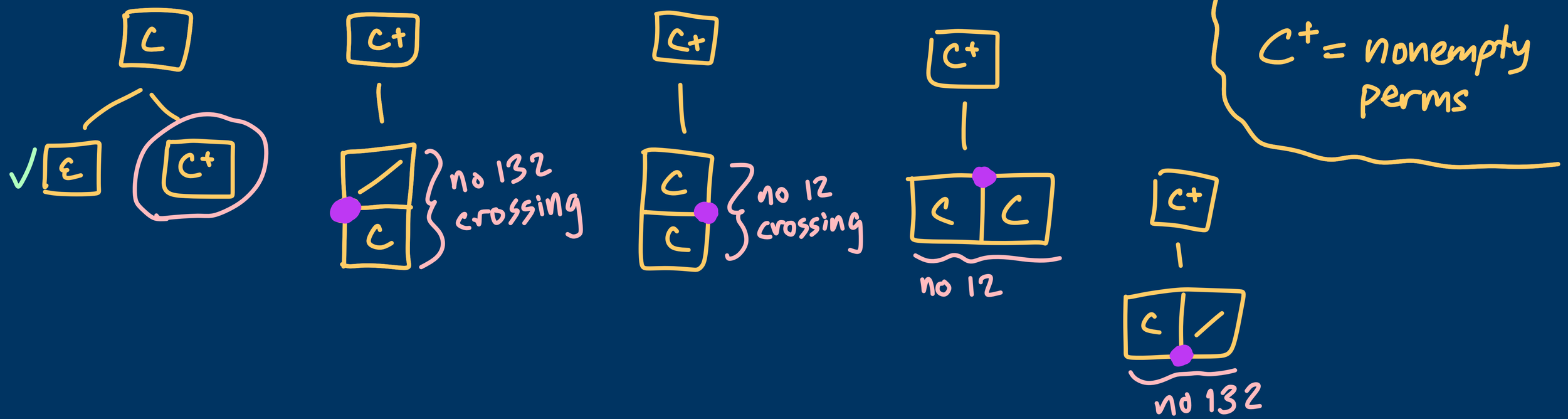


empty or not

point placement

row/col separation

factor

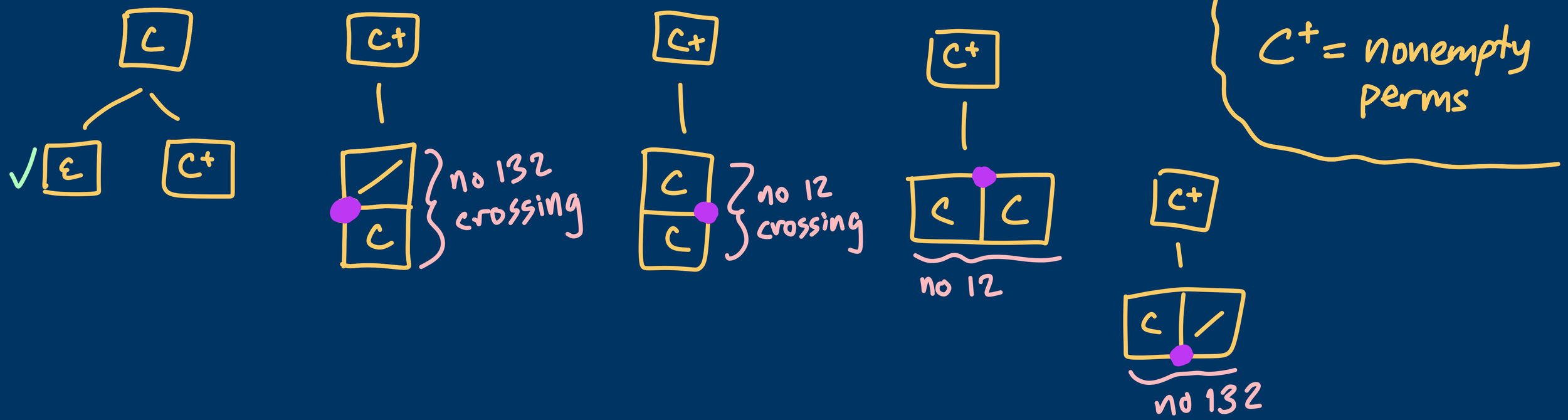


empty or not

point placement

row/col separation

factor

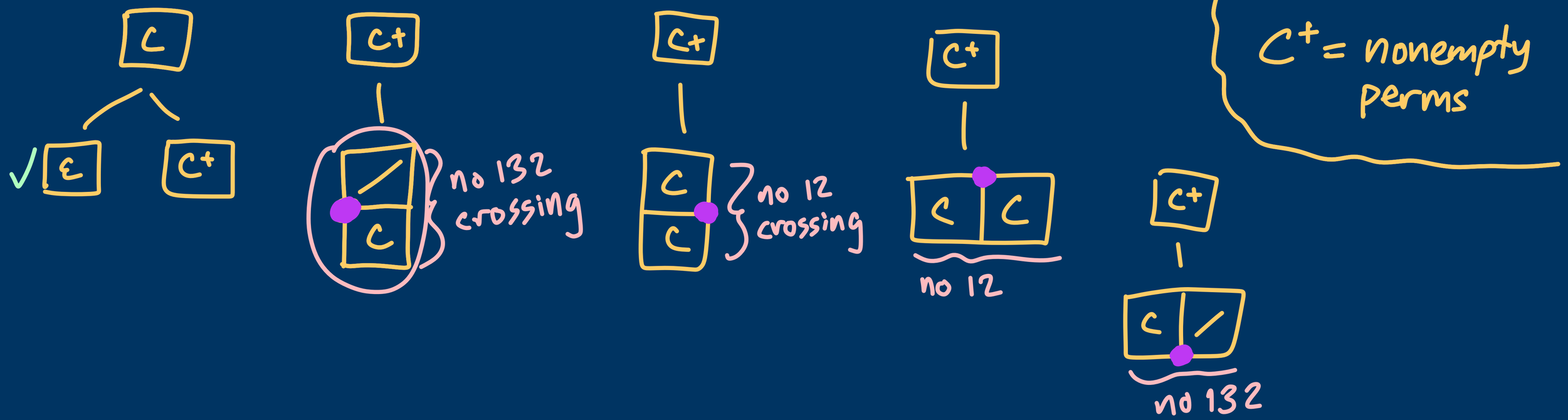


empty or not

point placement

row/col separation

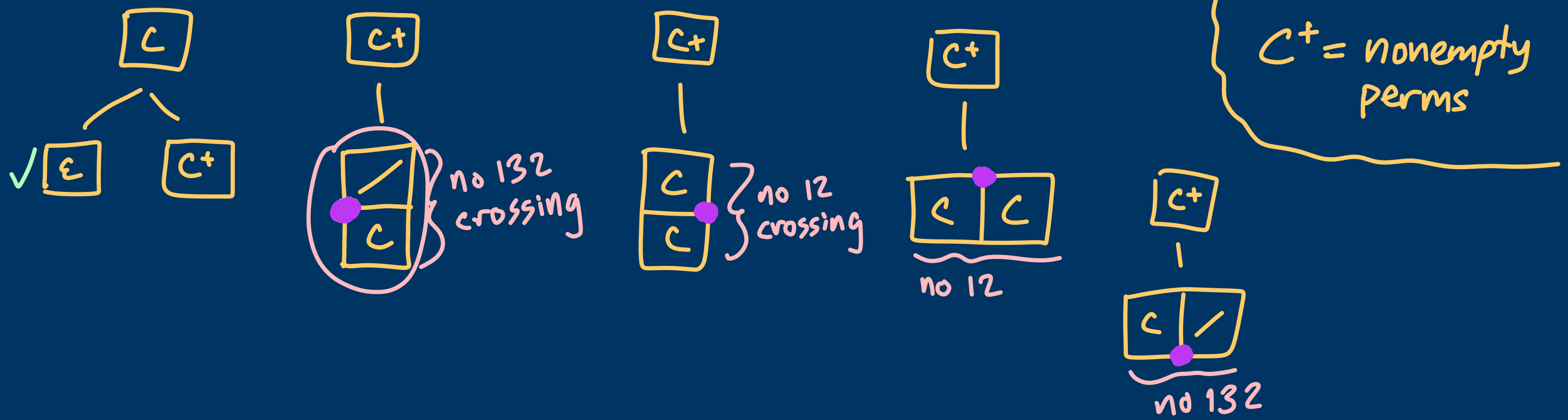
factor



empty or not

point placement

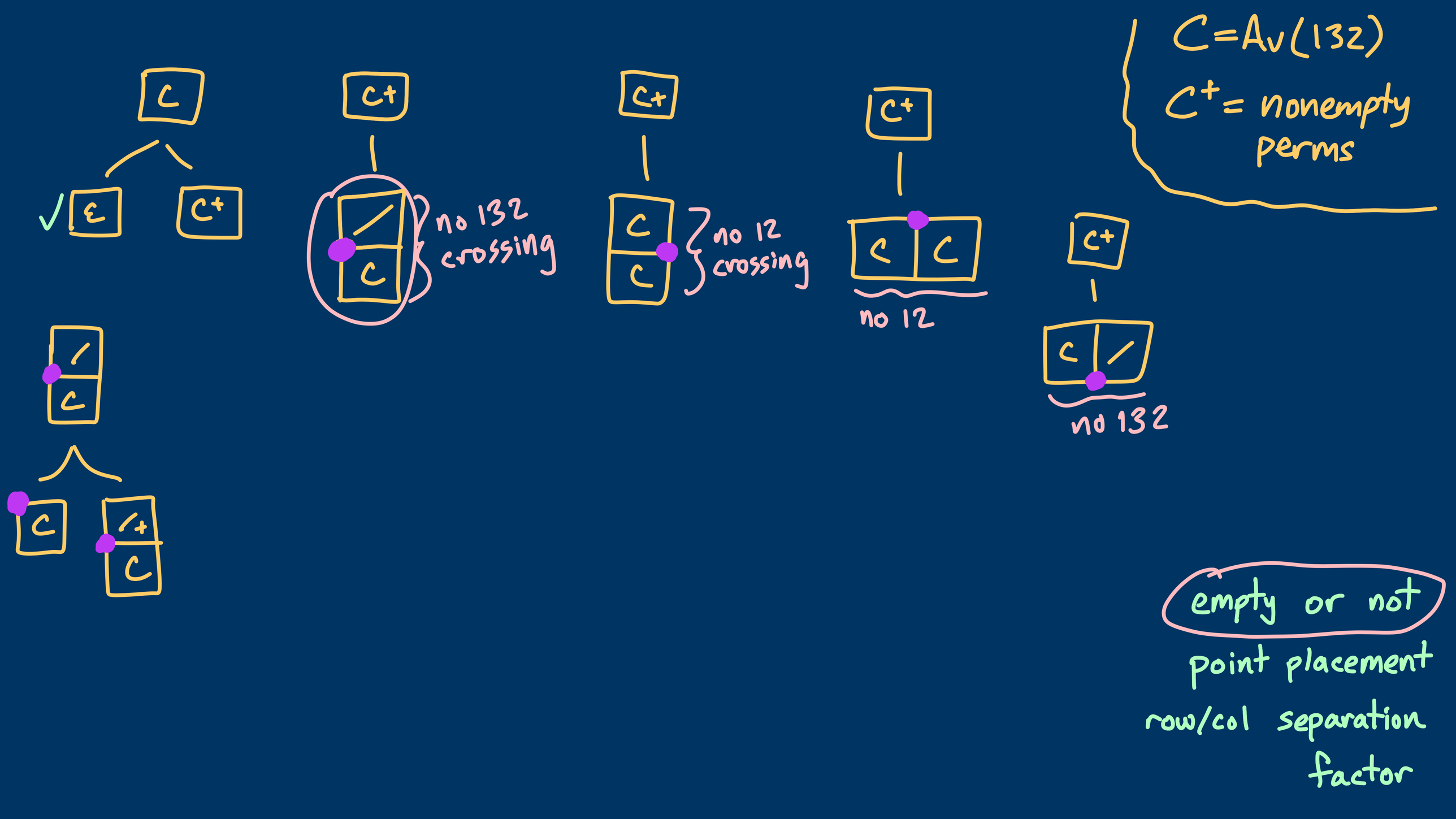
row/col separation factor

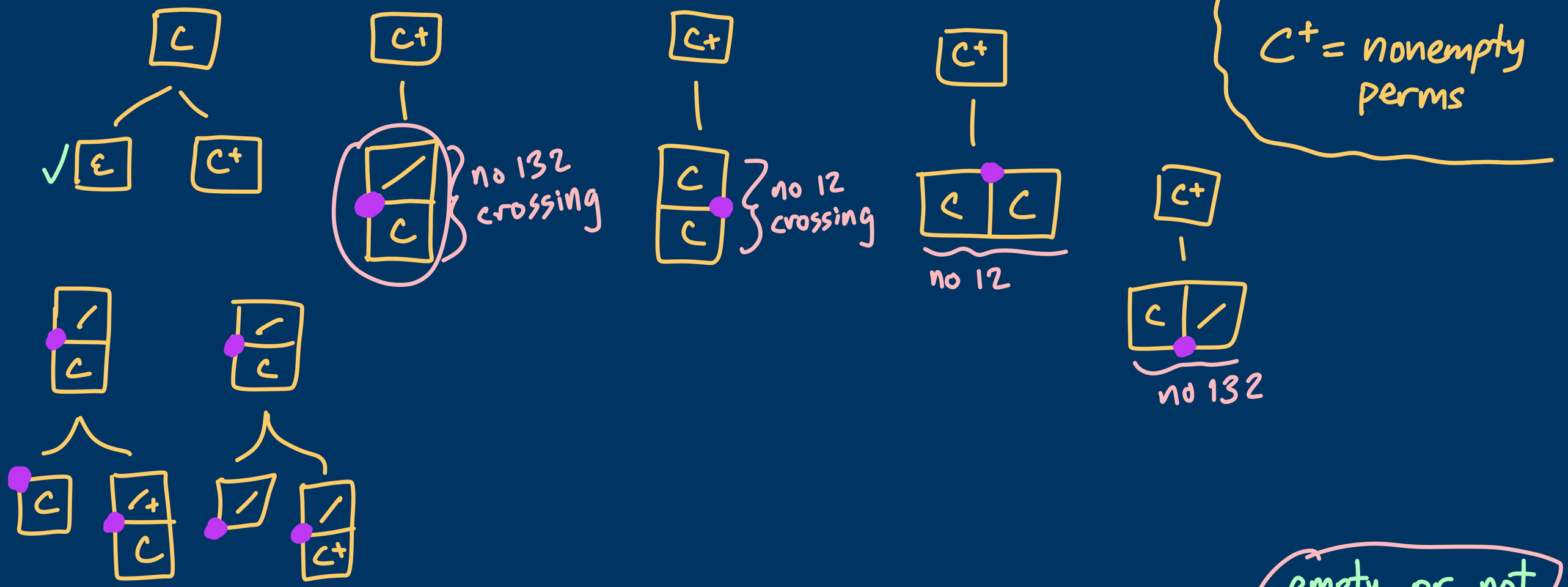


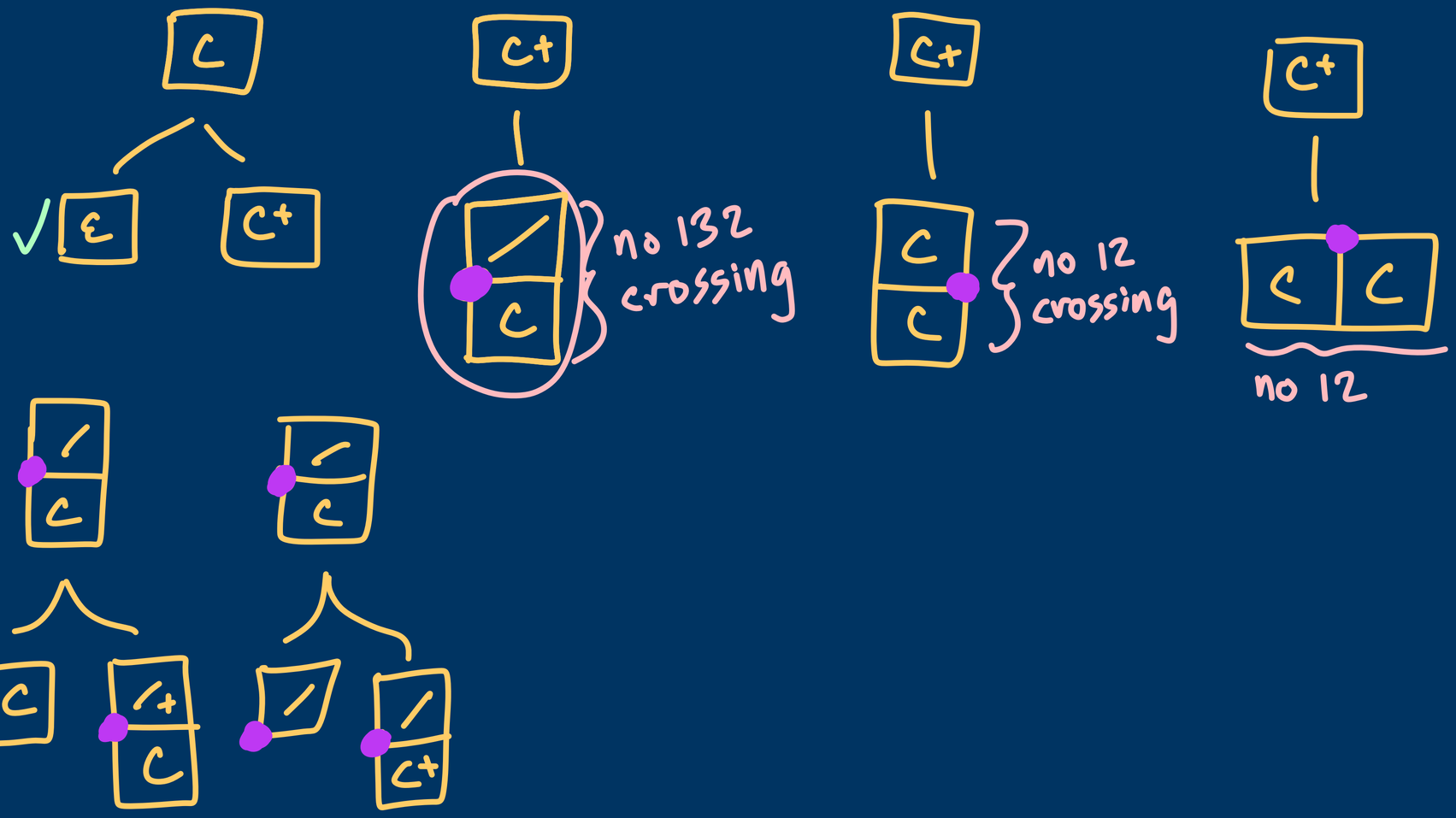
empty or not

point placement

row/col separation factor


C
√ε
C+
C+
no 132 crossing
C+
no 12 crossing
C+
no 12
C+
no 132
C=Av(132)
C+= nonempty perms
empty or not
point placement
row/col separation
factor


C
√ ε
C+
C+
C
C
no 132 crossing
C+
C
C
no 12 crossing
C+
C C
no 12
C+
C
no 132
C=Av(132)
C+ = nonempty perms
empty or not
point placement
row/col separation
factor



$C = Av(132)$

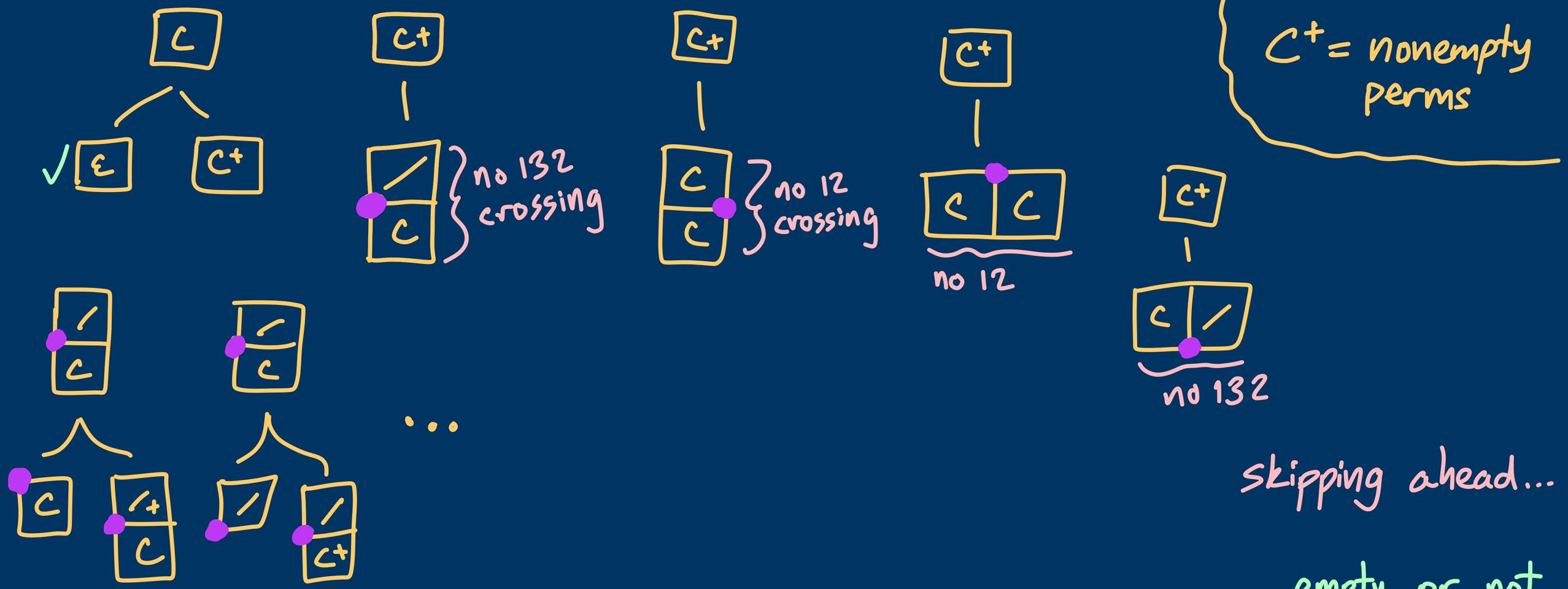
C^+ = nonempty perms

skipping ahead...

empty or not

point placement

row/col separation factor



$C = Av(132)$

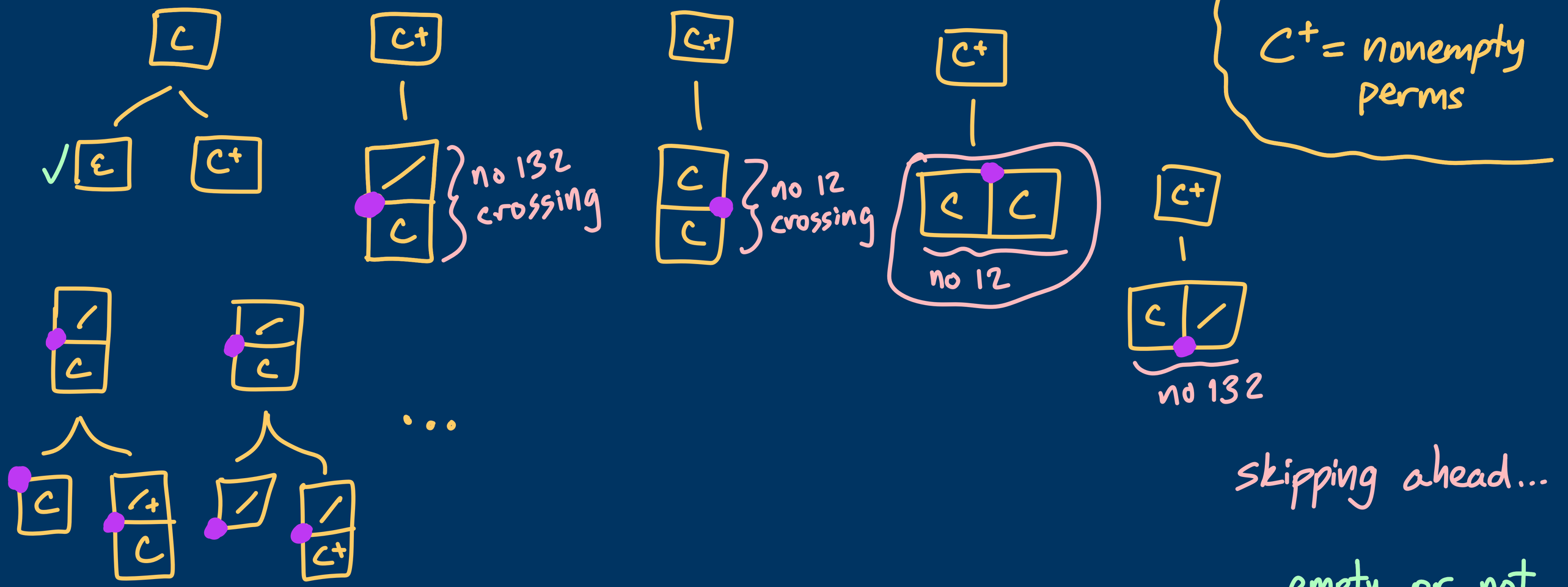
C^+ = nonempty perms

skipping ahead...

empty or not

point placement

row/col separation factor



$C = Av(132)$

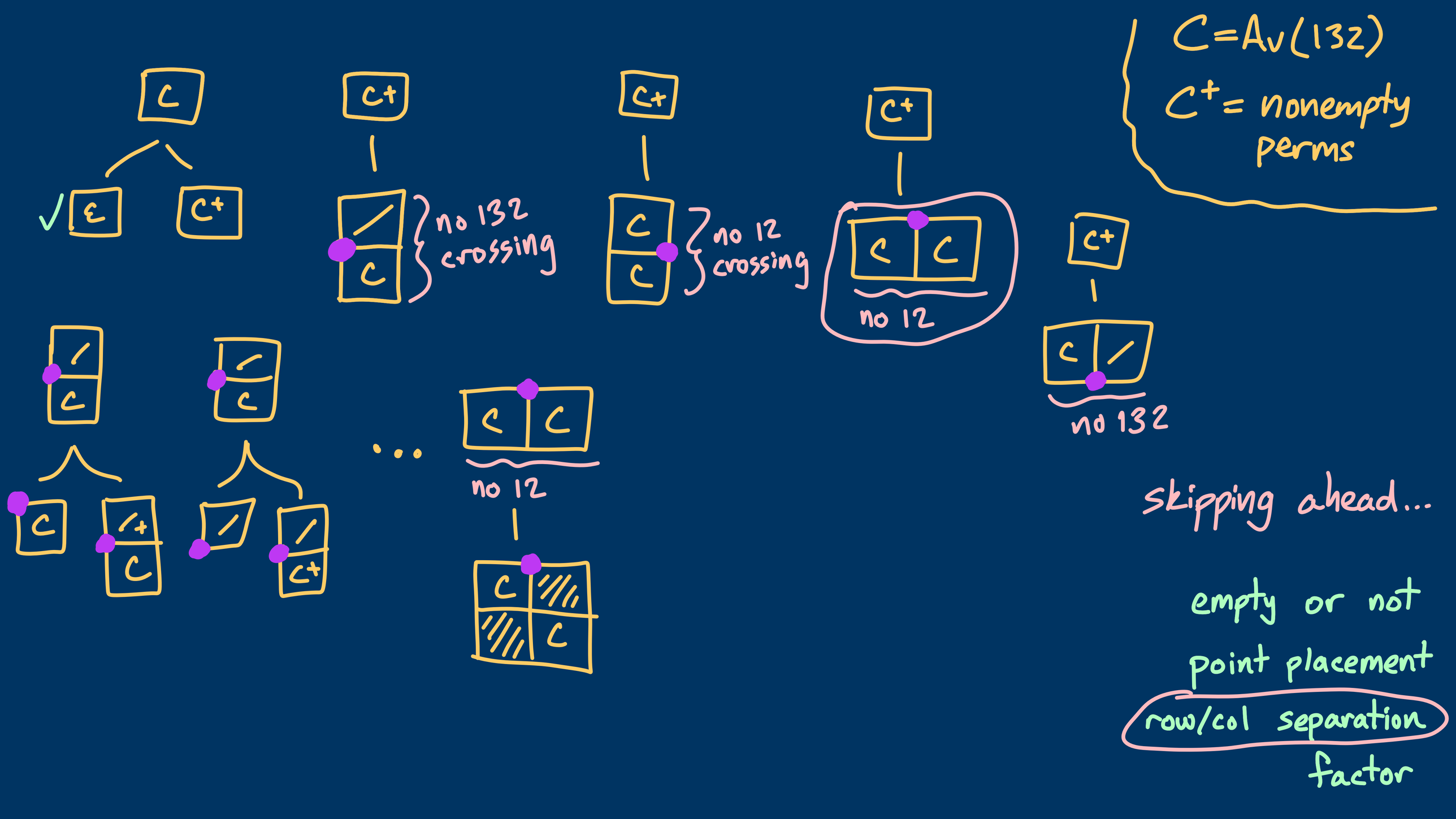
C^+ = nonempty perms

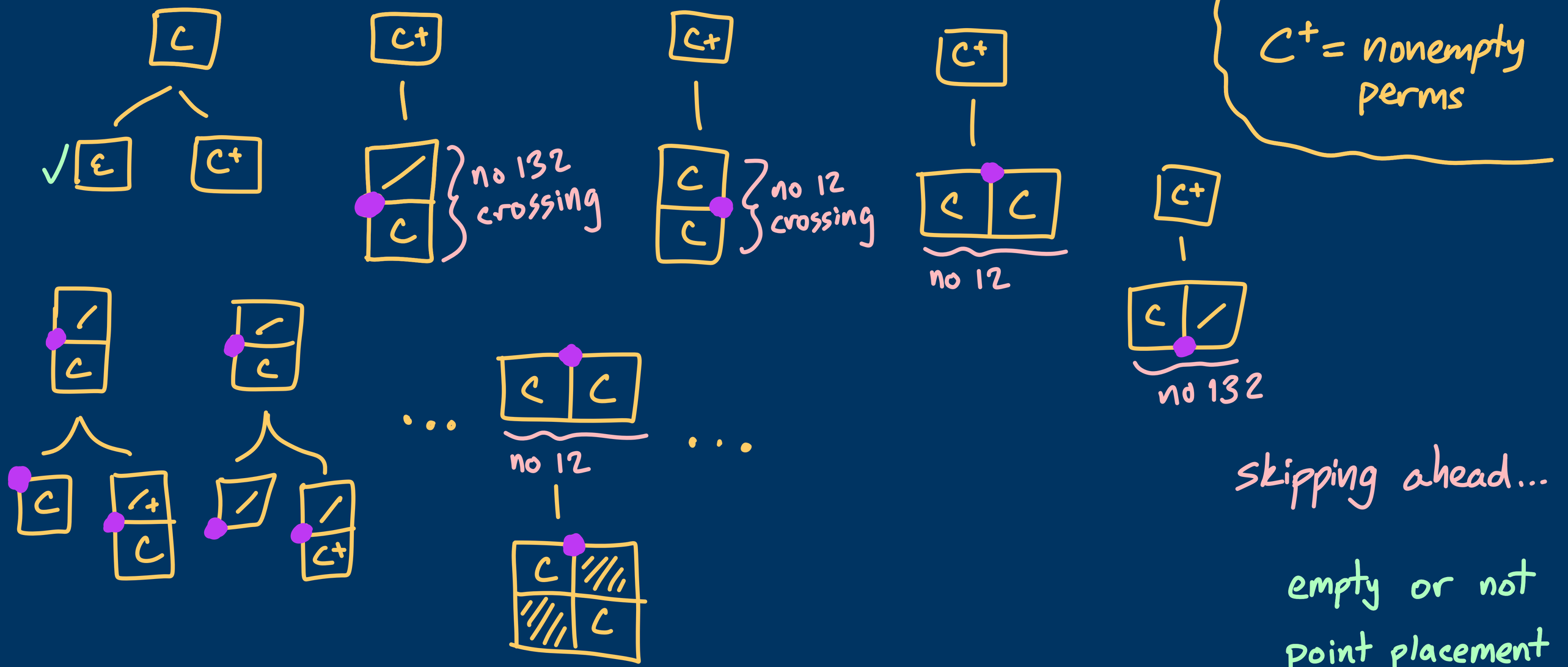
skipping ahead...

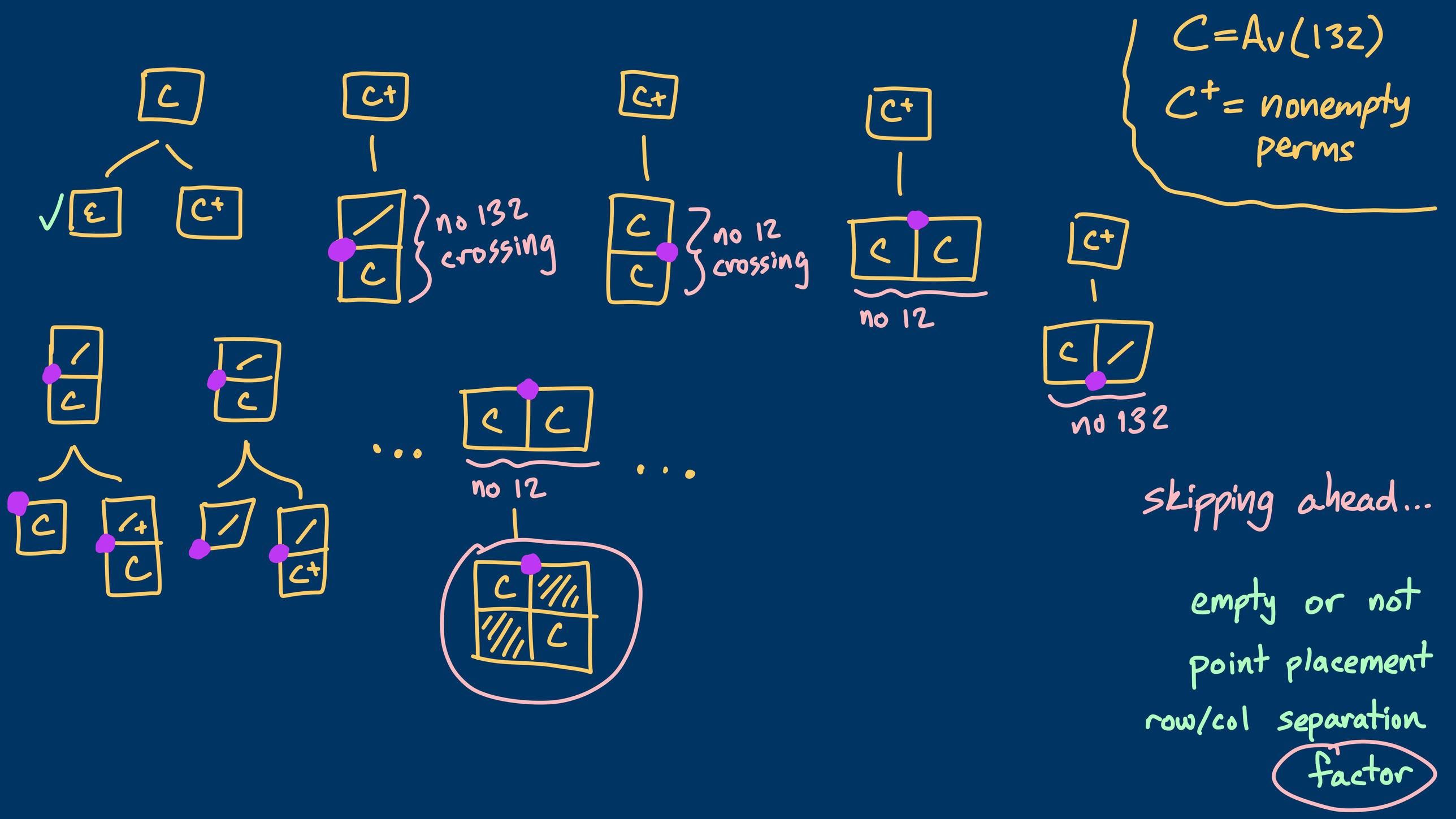
empty or not

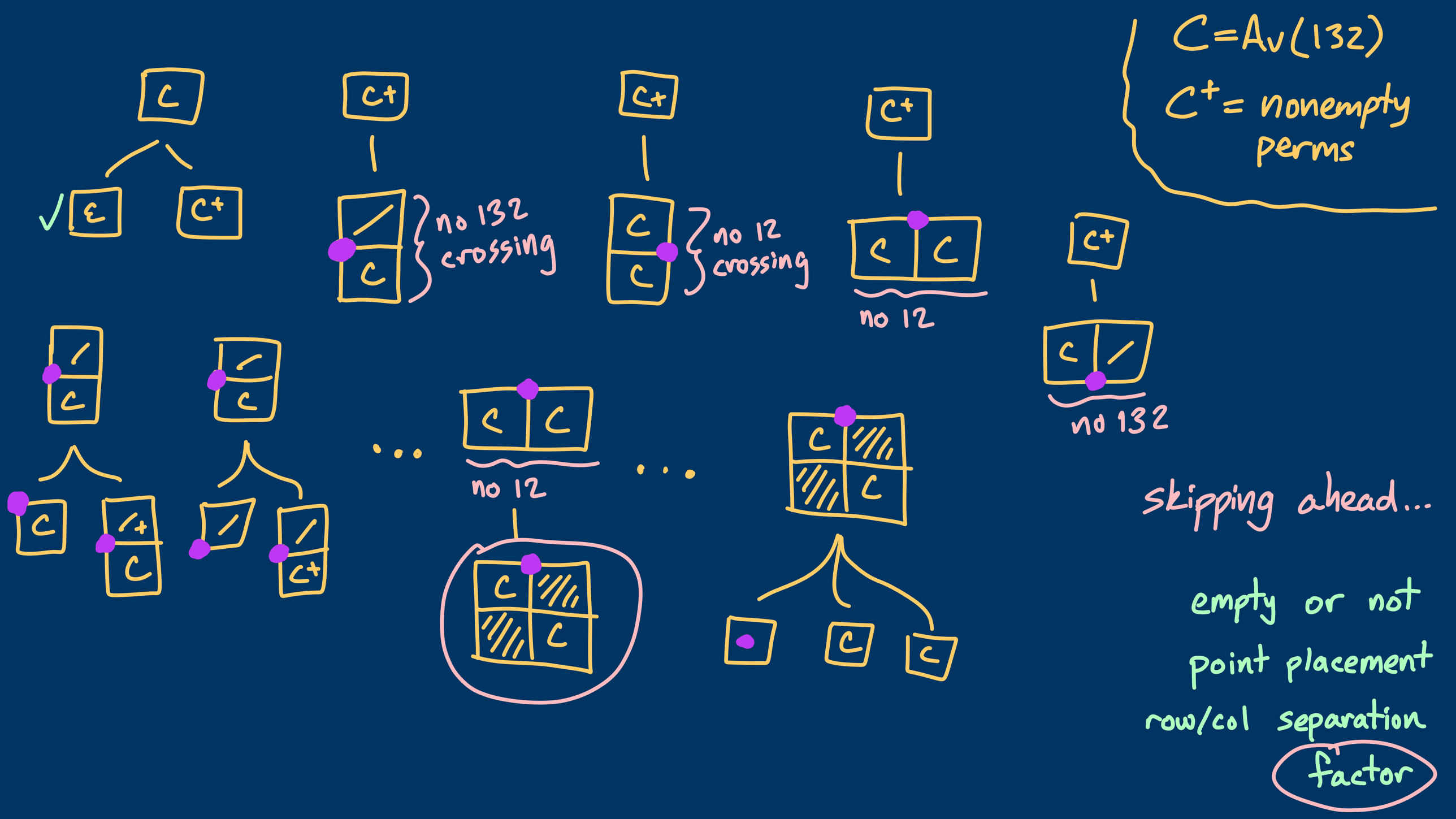
point placement

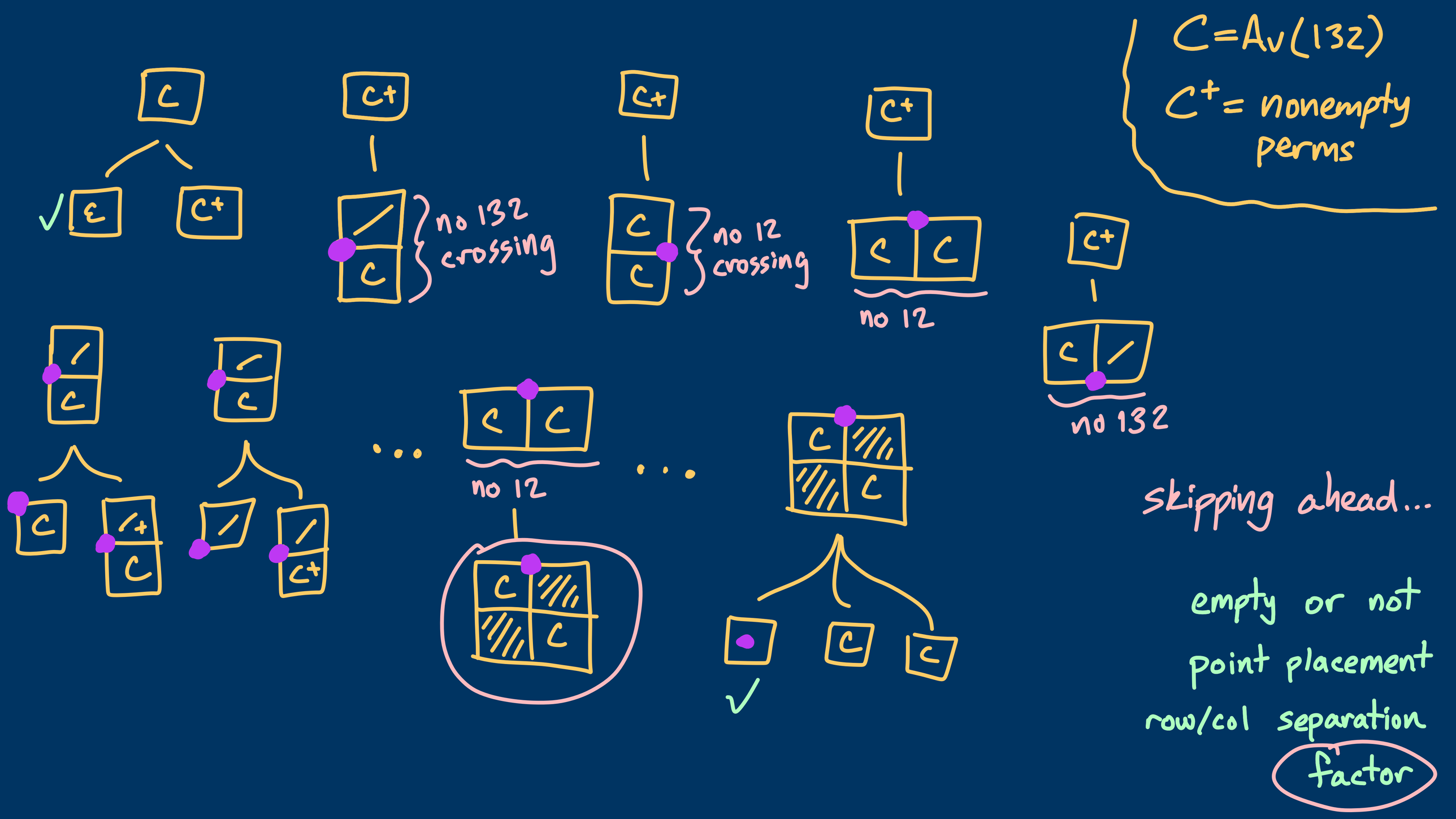
row/col separation factor

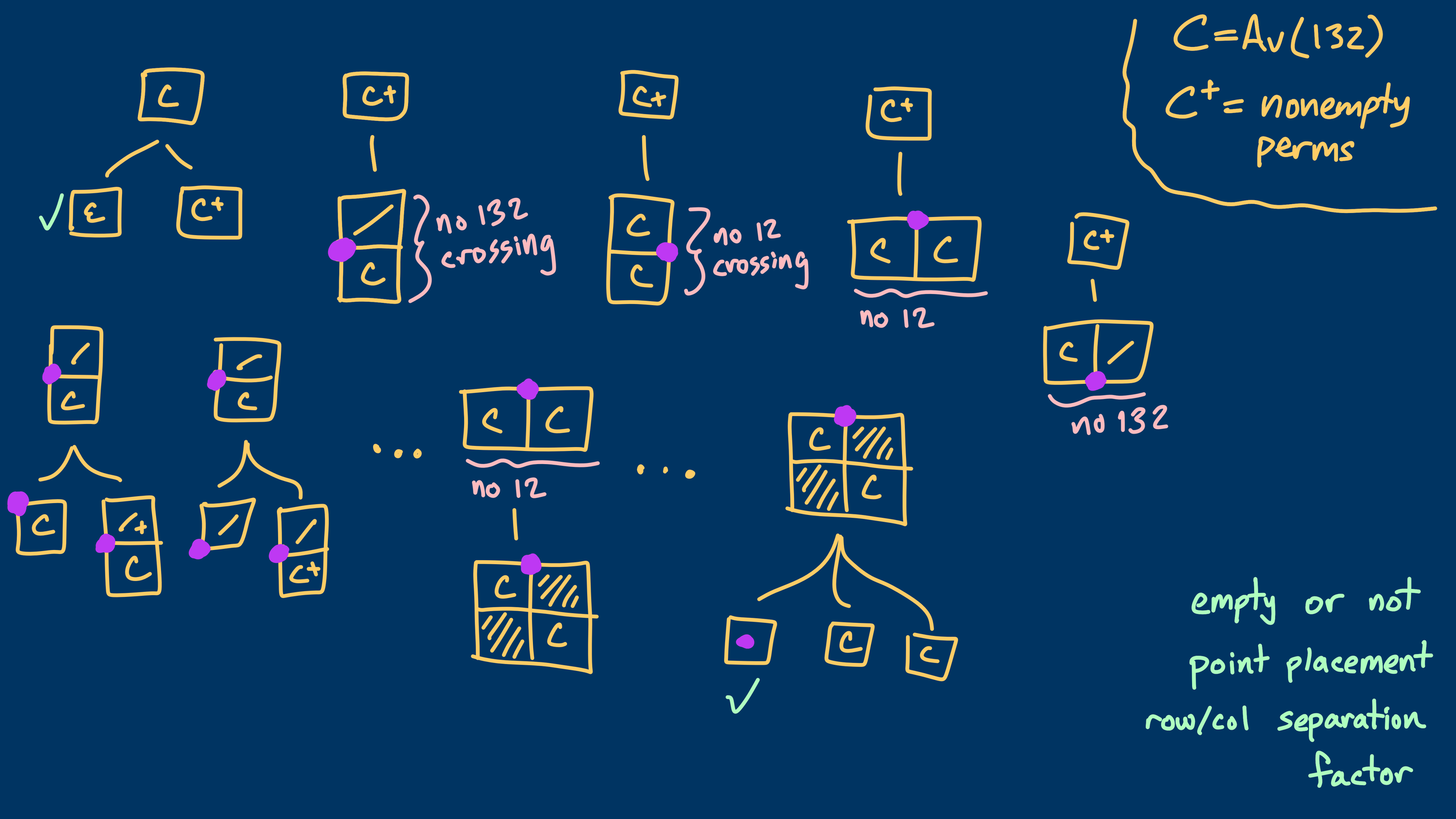

C
C+
ε
no 132 crossing
no 12 crossing
no 12
no 132
...
C=Av(132)
C+= nonempty perms
skipping ahead...
empty or not
point placement
row/col separation
factor

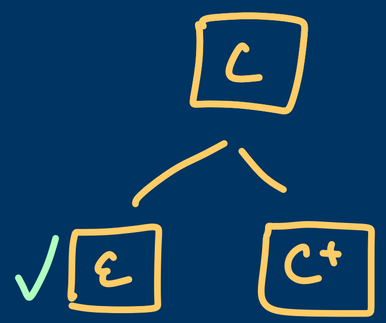

C
C+
ε
no 132 crossing
no 12 crossing
no 12
no 132
C=Av(132)
C+= nonempty perms
skipping ahead...
empty or not
point placement
row/col separation
factor

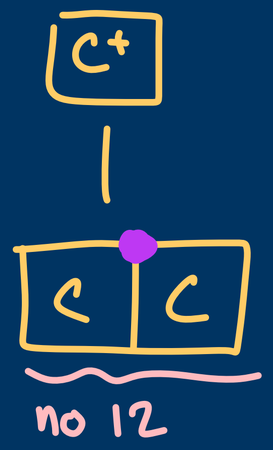

C
C+
ε
no 132 crossing
no 12 crossing
no 12
no 132
C=Av(132)
C+ = nonempty perms
skipping ahead...
empty or not
point placement
row/col separation
factor


C
C+
ε
no 132 crossing
no 12 crossing
no 12
no 132
C=Av(132)
C+= nonempty perms
skipping ahead...
empty or not
point placement
row/col separation
factor


C
C+
ε
no 132 crossing
no 12 crossing
no 12
no 132
C=Av(132)
C+= nonempty perms
skipping ahead...
empty or not
point placement
row/col separation
factor

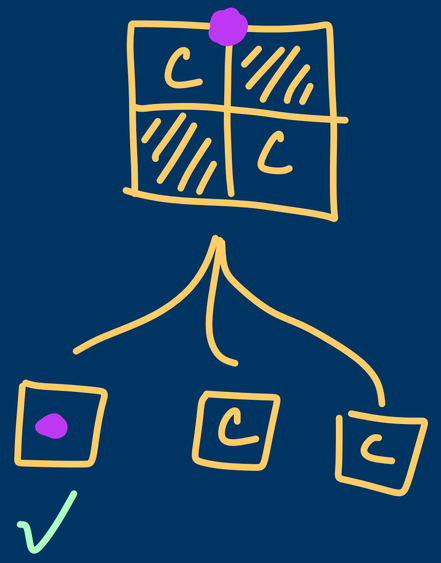

C=Av(132)
C+= nonempty perms
C
C+
ε
no 132 crossing
no 12 crossing
no 12
no 132
empty or not
point placement
row/col separation
factor





$C = Av(132)$

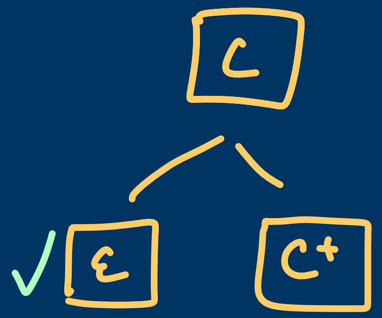
C^+ = nonempty perms

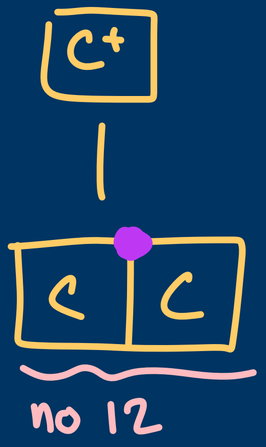


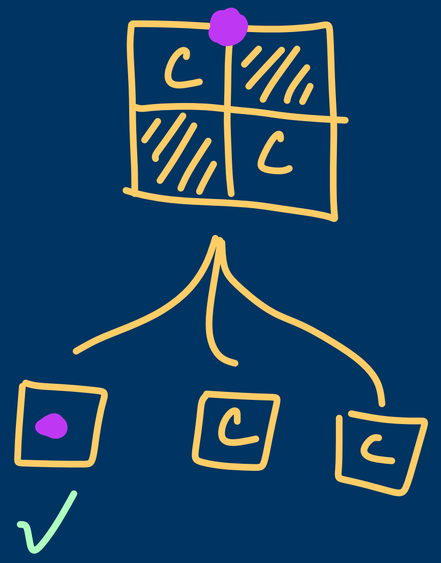
empty or not

point placement

row/col separation factor







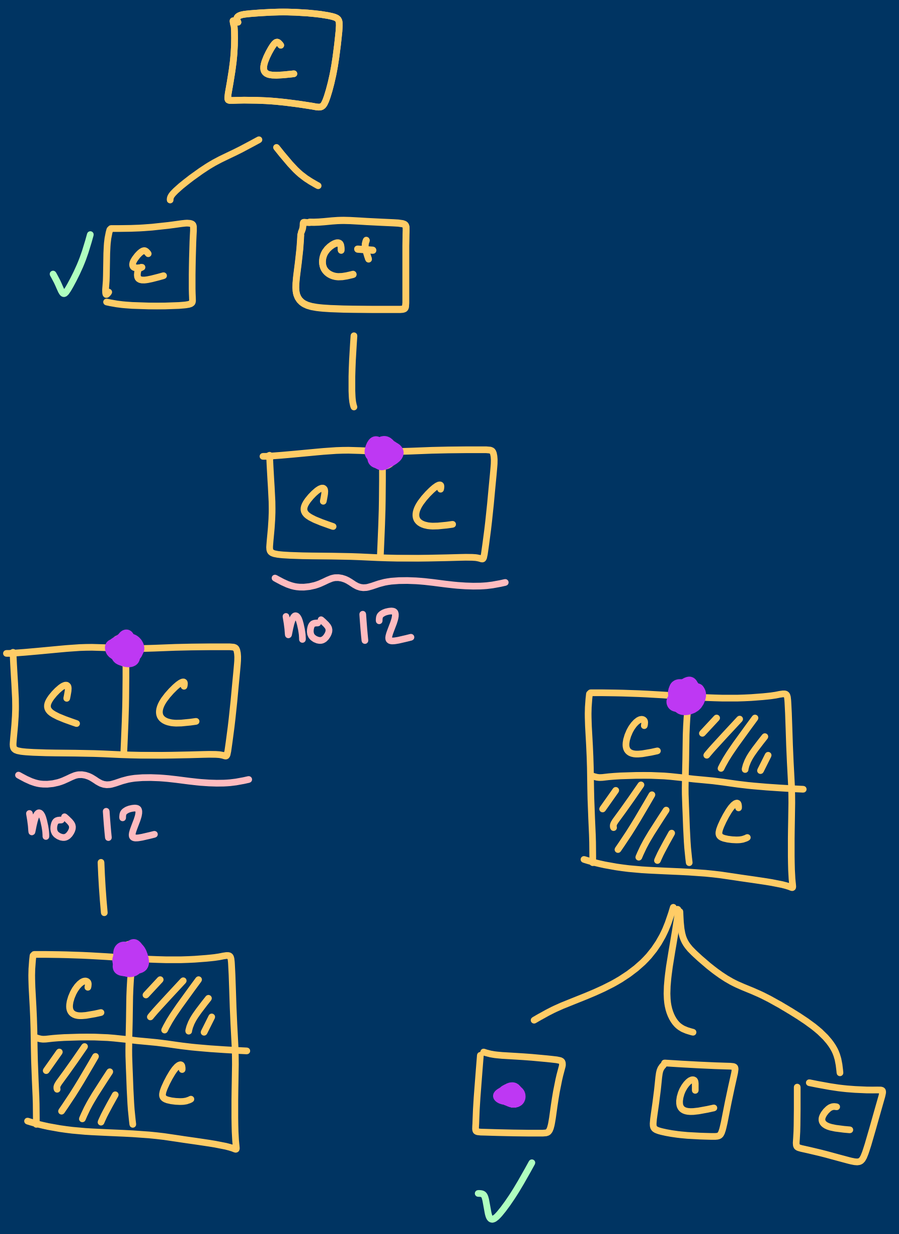
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



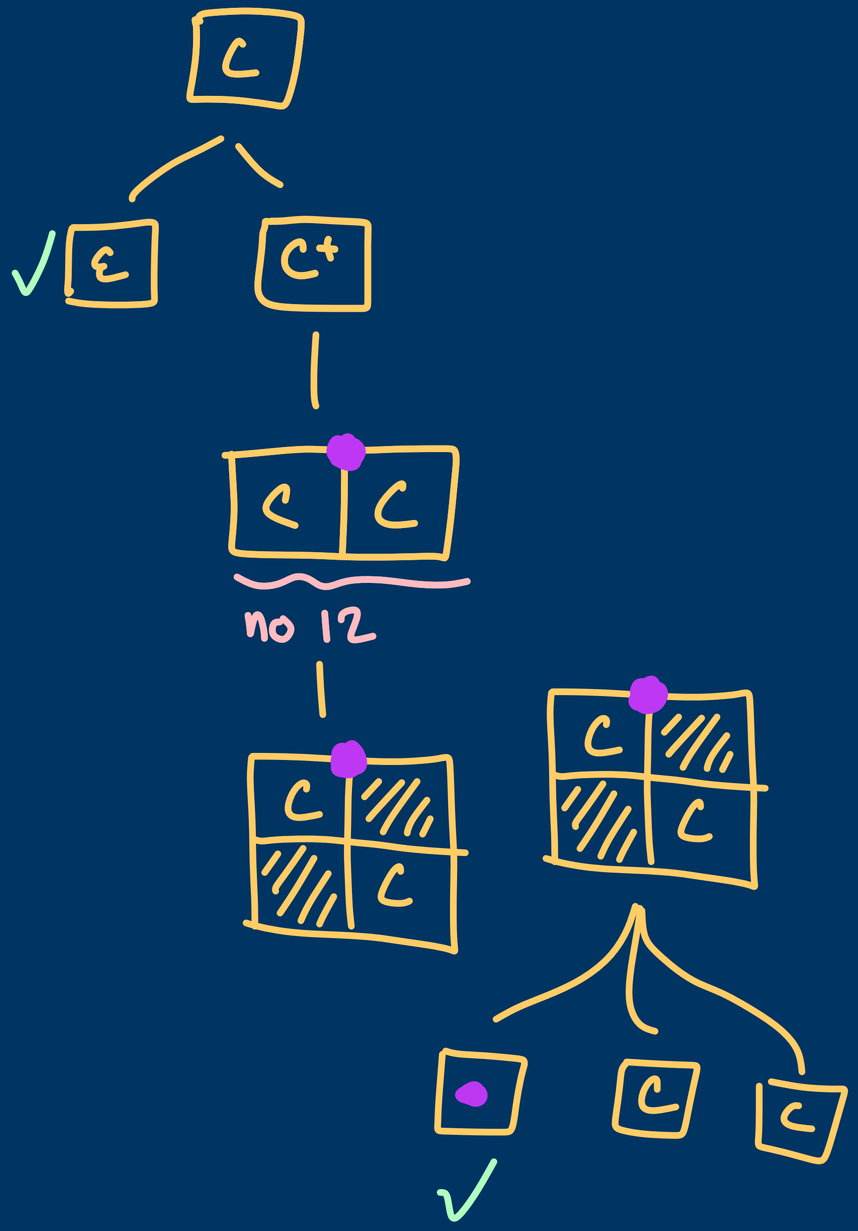
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



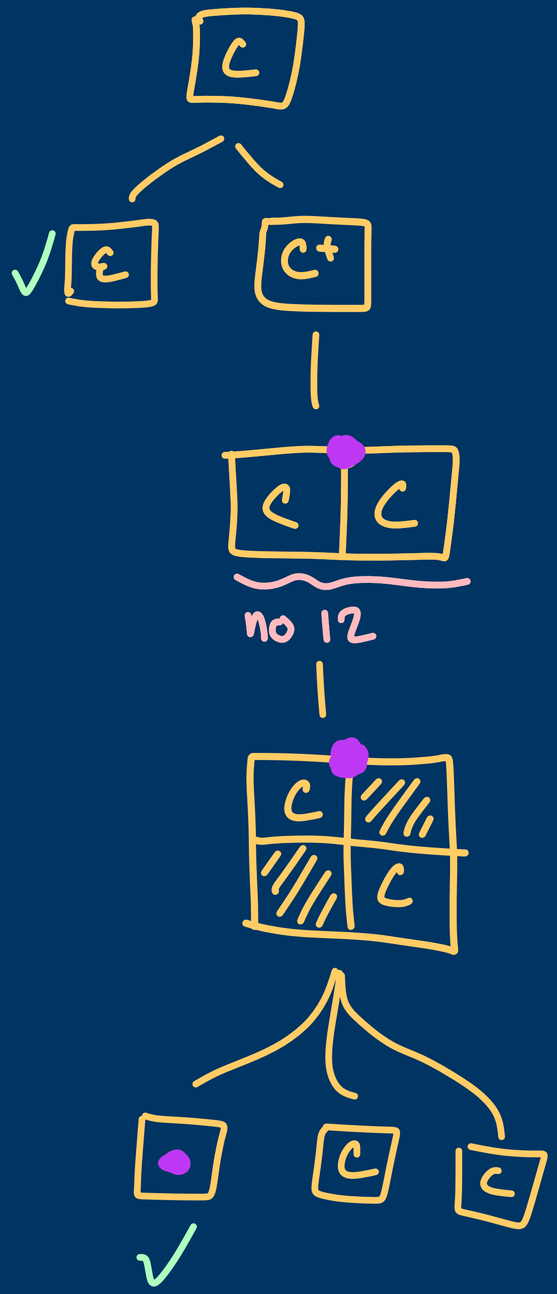
$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor



$C = Av(132)$

C^+ = nonempty perms

empty or not

point placement

row/col separation factor

Combinatorial Exploration

General outline:

- Teach the computer a set of strategies.

Combinatorial Exploration

General outline:

- Teach the computer a set of strategies.
- Apply them to the set of permutations you want to enumerate.

Combinatorial Exploration

General outline:

- Teach the computer a set of strategies.
- Apply them to the set of permutations you want to enumerate.
  and then to the children they produced

Combinatorial Exploration

General outline:

- Teach the computer a set of strategies.
- Apply them to the set of permutations you want to enumerate.
  and then to the children they produced
    and then to the children they produced

Combinatorial Exploration

General outline:

- Teach the computer a set of strategies.
- Apply them to the set of permutations you want to enumerate.
  and then to the children they produced
  and then to the children they produced
  and then to the children they produced
  …

Combinatorial Exploration

General outline:

- Teach the computer a set of strategies.
- Apply them to the set of permutations you want to enumerate.
  and then to the children they produced
  and then to the children they produced
  and then to the children they produced
  …
- Each time you apply a strategy to a set, you make a puzzle piece. Search the pile of puzzle pieces for a subset that makes a combinatorial specification. If you find one, you win!
  polynomial-time counting algorithm, system of equations for the GF,
  uniform random sampling routine, exhaustive generation (but slow)

Combinatorial Exploration

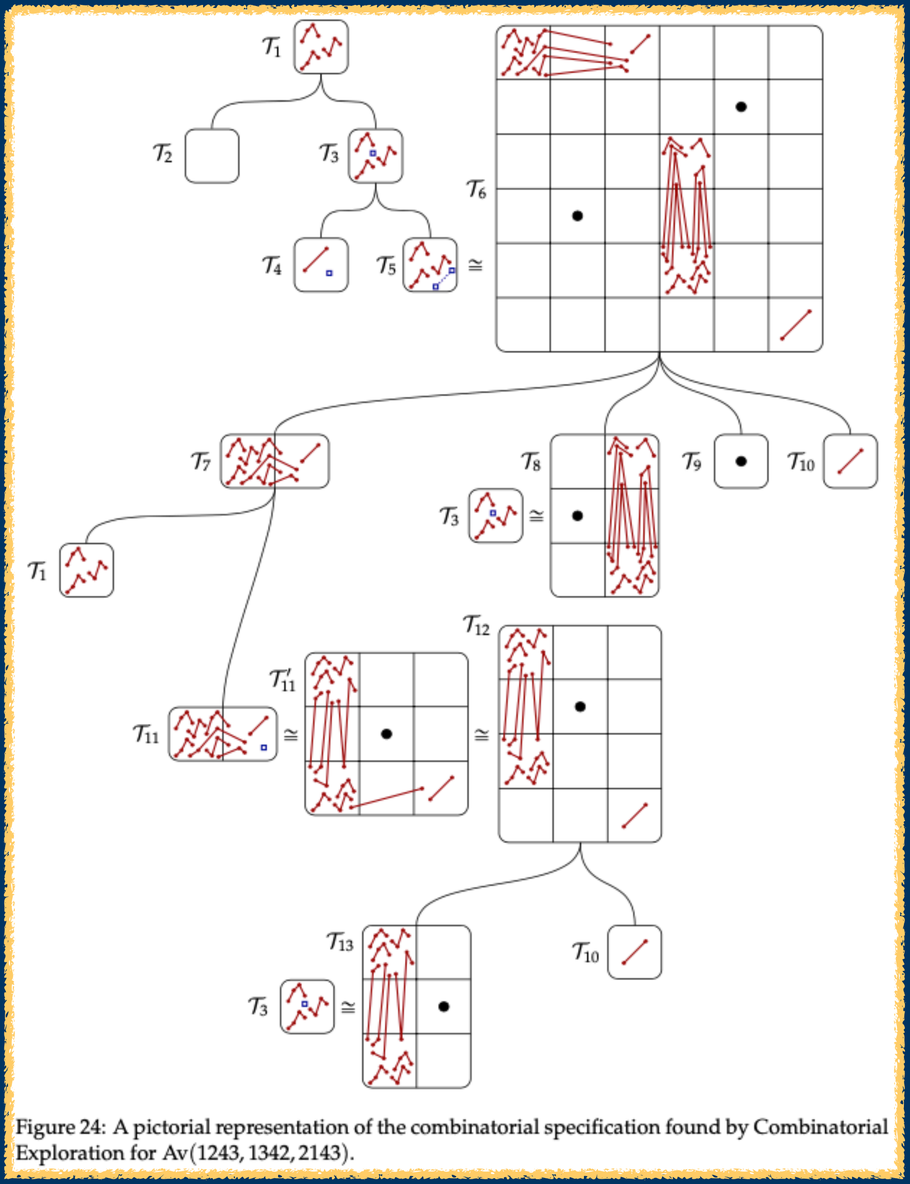


Figure 24: A pictorial representation of the combinatorial specification found by Combinatorial Exploration for Av(1243, 1342, 2143).

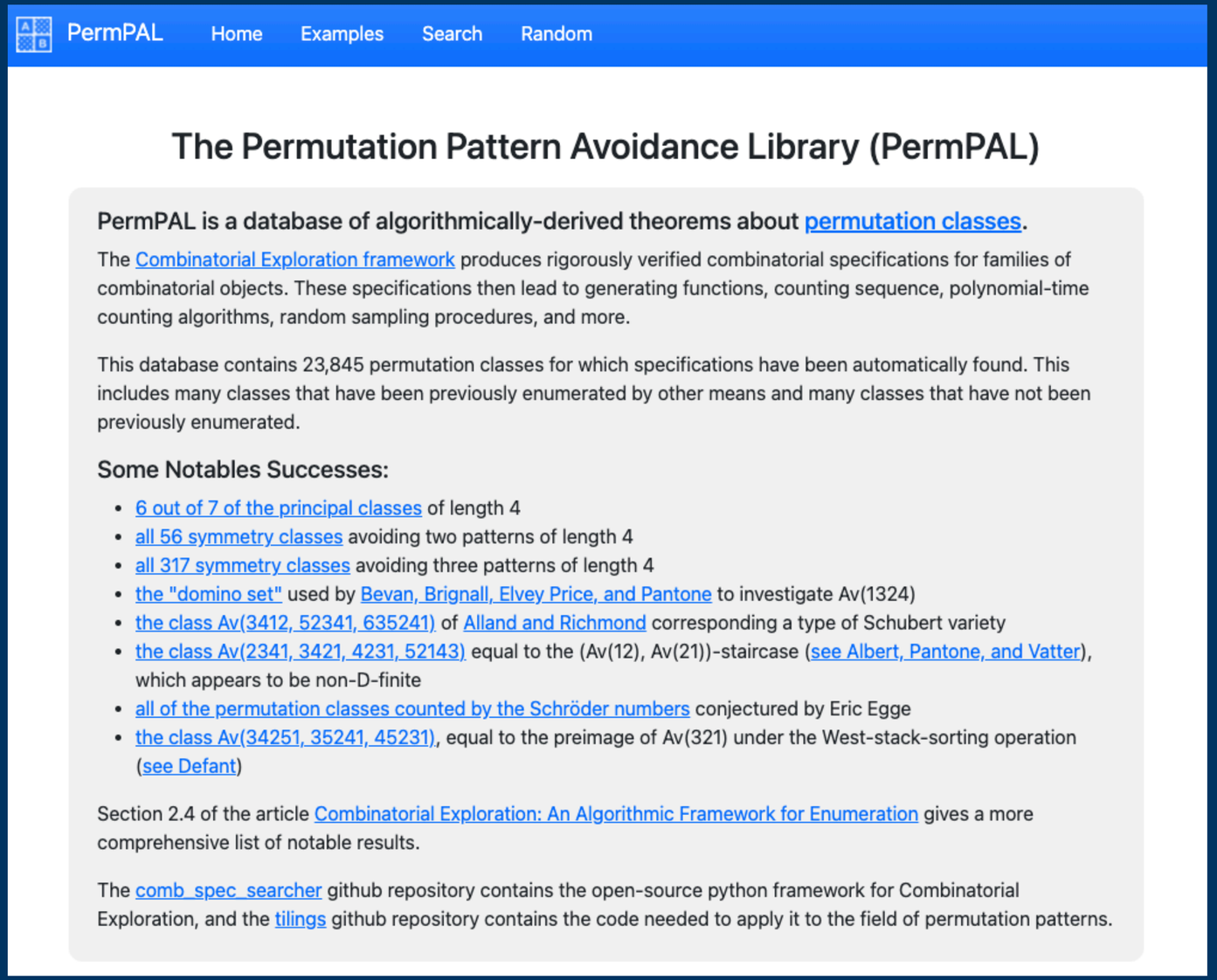
Av(1243, 1342, 2143)

The algorithm generates about 5,400 rules before it finds this subset of 10 rules that makes a rigorous specification.

55 seconds

Combinatorial Exploration

https://permpal.com



Combinatorial Exploration

We can find combinatorial specifications for:

- 6 out of 7 of the classes avoiding 1 pattern of length 4
  First direct enumerations of $\mathrm{Av}(1342)$ and $\mathrm{Av}(2413)$

- All 56 classes avoiding 2 patterns of length 4
  3 are conjectured to be non-D-finite
  can derive the algebraic GF for the other 53

- All 317 classes avoiding 3 patterns of length 4

- All classes avoiding 4 or more patterns of length 4

Combinatorial Exploration

We can find combinatorial specifications for:

- 1324-avoiding domino permutations
- Preimage of $\mathrm{Av}(321)$ under West-stack-sorting
  $\mathrm{Av}(34251, 35241, 45231)$
- LCI Schubert Varieties
  $\mathrm{Av}(52341, 53241, 52431, 35142, 42513, 351624)$
- “Box classes” like $\mathrm{Av}(1\square 2\square 3)$ and $\mathrm{Av}(1\square\square 32)$
- “POP classes”
- Permutations corresponding to Schubert varieties with a complete parabolic bundle structure
  $\mathrm{Av}(3412, 52341, 635241)$

Av(2143, 3412)

View Raw Data

Generating Function

$$\frac{3x-1}{\sqrt{-4x+1}\,(2x-1)}$$

Copy to clipboard: latex Maple sympy Search on PermPAL

Counting Sequence

1, 1, 2, 6, 22, 86, 340, 1340, 5254, 20518, 79932, 311028, 1209916, 4707964, 18330728, ...

Copy 101 terms to clipboard Search on OEIS Search on PermPAL

Recurrence

$$a(0) = 1$$
$$a(1) = 1$$
$$a(2) = 2$$
$$a(n+3) = \frac{12\,(1+2n)a(n)}{n+3} - \frac{2\,(16+13n)a(n+1)}{n+3} + \frac{(19+9n)a(n+2)}{n+3},$$

Copy to clipboard: latex Maple

Implicit Equation for the Generating Function

$$(4x-1)(2x-1)^2F(x)^2 + (3x-1)^2 = 0$$

Copy to clipboard: latex Maple Search on PermPAL

Heatmap

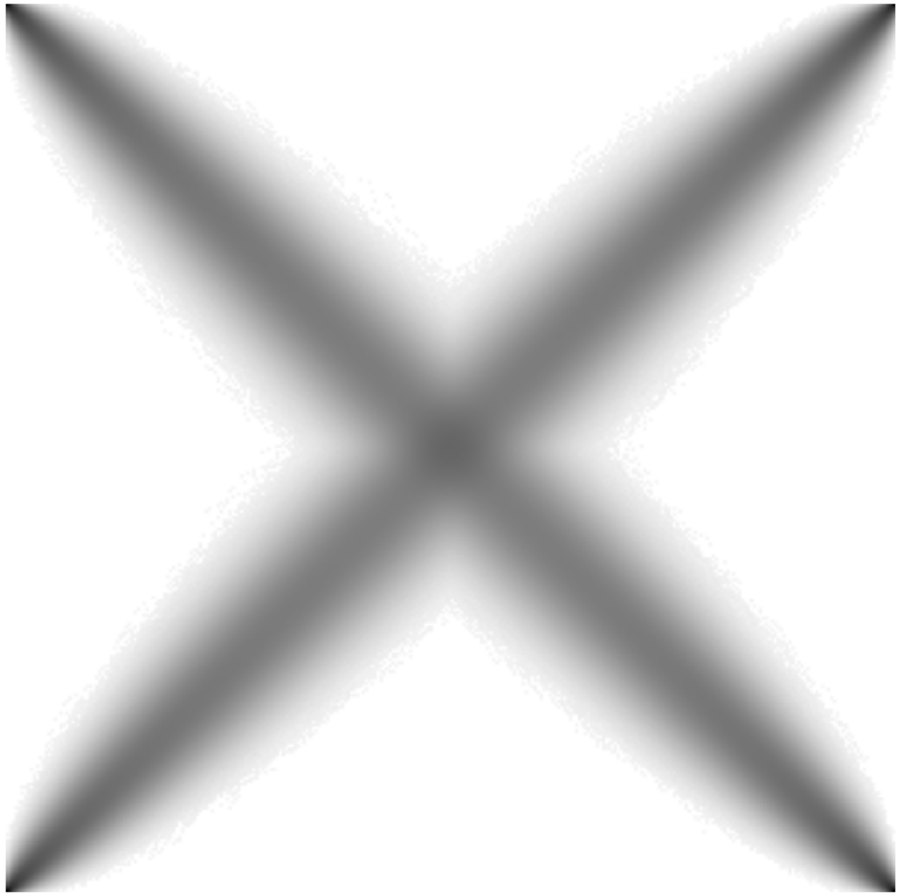
To create this heatmap, we sampled 1,000,000 permutations of length 300 uniformly at random. The color of the point (i, j) represents how many permutations have value j at index i (darker = more).

$$a(n+3) = \frac{12(1+2n)a(n)}{n+3} - \frac{2(16+13n)a(n+1)}{n+3} + \frac{(19+9n)a(n+2)}{n+3},$$

Copy to clipboard: latex Maple

Heatmap

To create this heatmap, we sampled 1,000,000 permutations of length 300 uniformly at random. The color of the point (i, j) represents how many permutations have value j at index i (darker = more).



Specification 1 | Specification 2 | Specification 3 | Specification 4 | Specification 5

This specification was found using the strategy pack "Row And Col Placements Tracked Fusion Isolated" and has 29 rules.

Found on April 21, 2021.

Specification 1 | Specification 2 | Specification 3 | Specification 4 | Specification 5

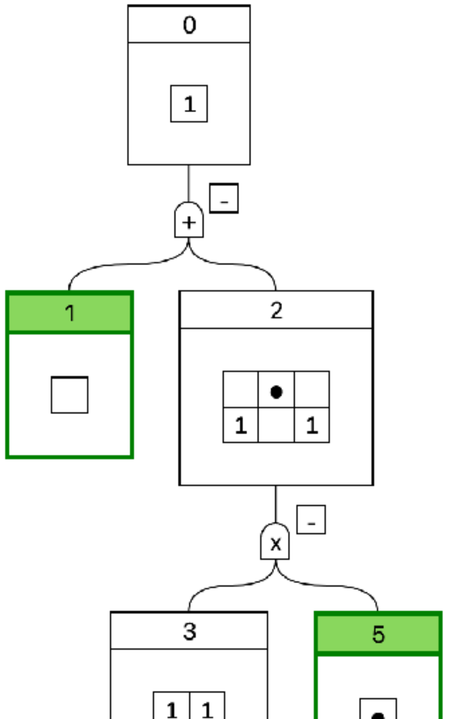
This specification was found using the strategy pack "Row And Col Placements Tracked Fusion Isolated" and has 29 rules.

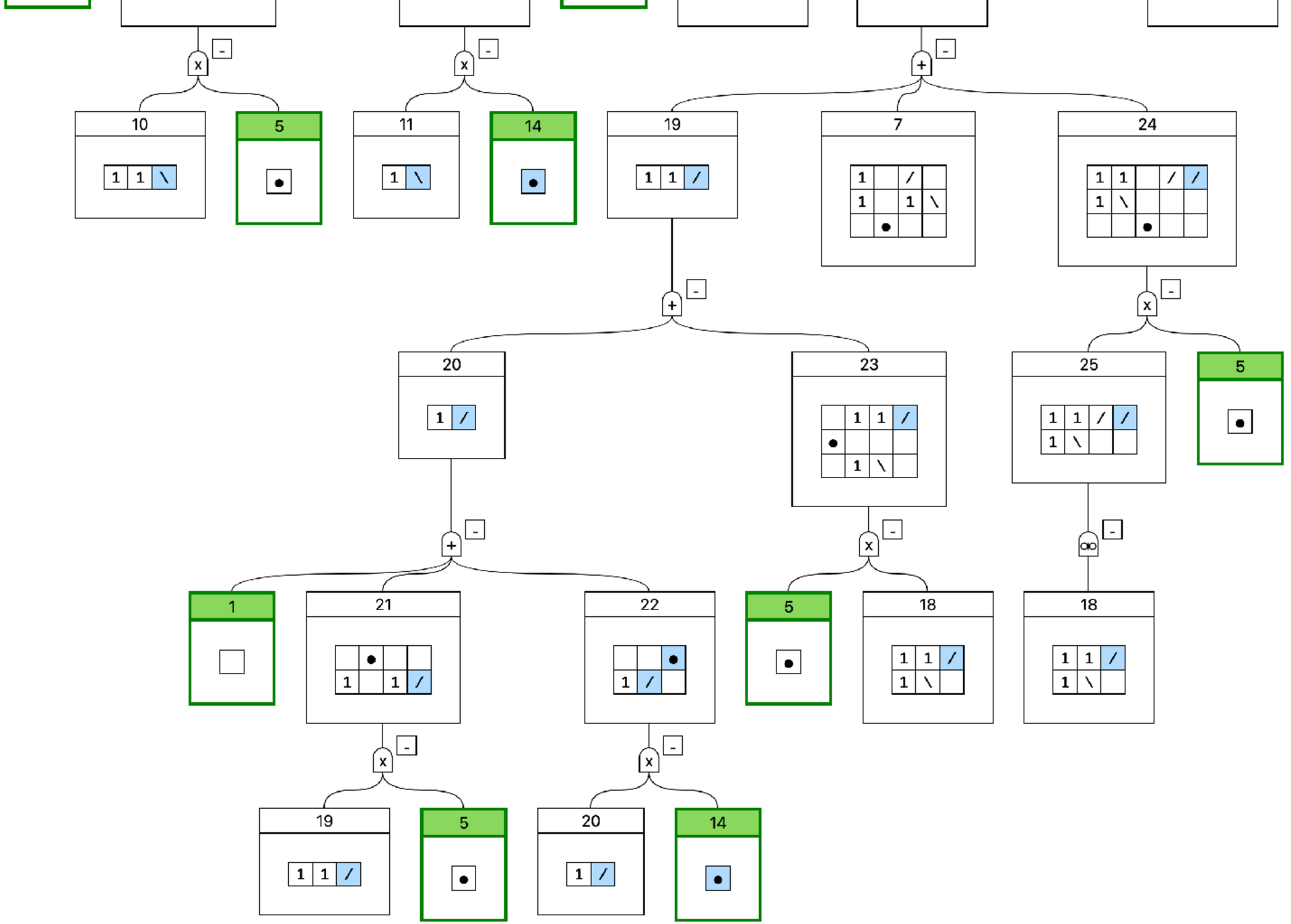
Found on April 21, 2021.
Finding the specification took 653 seconds.

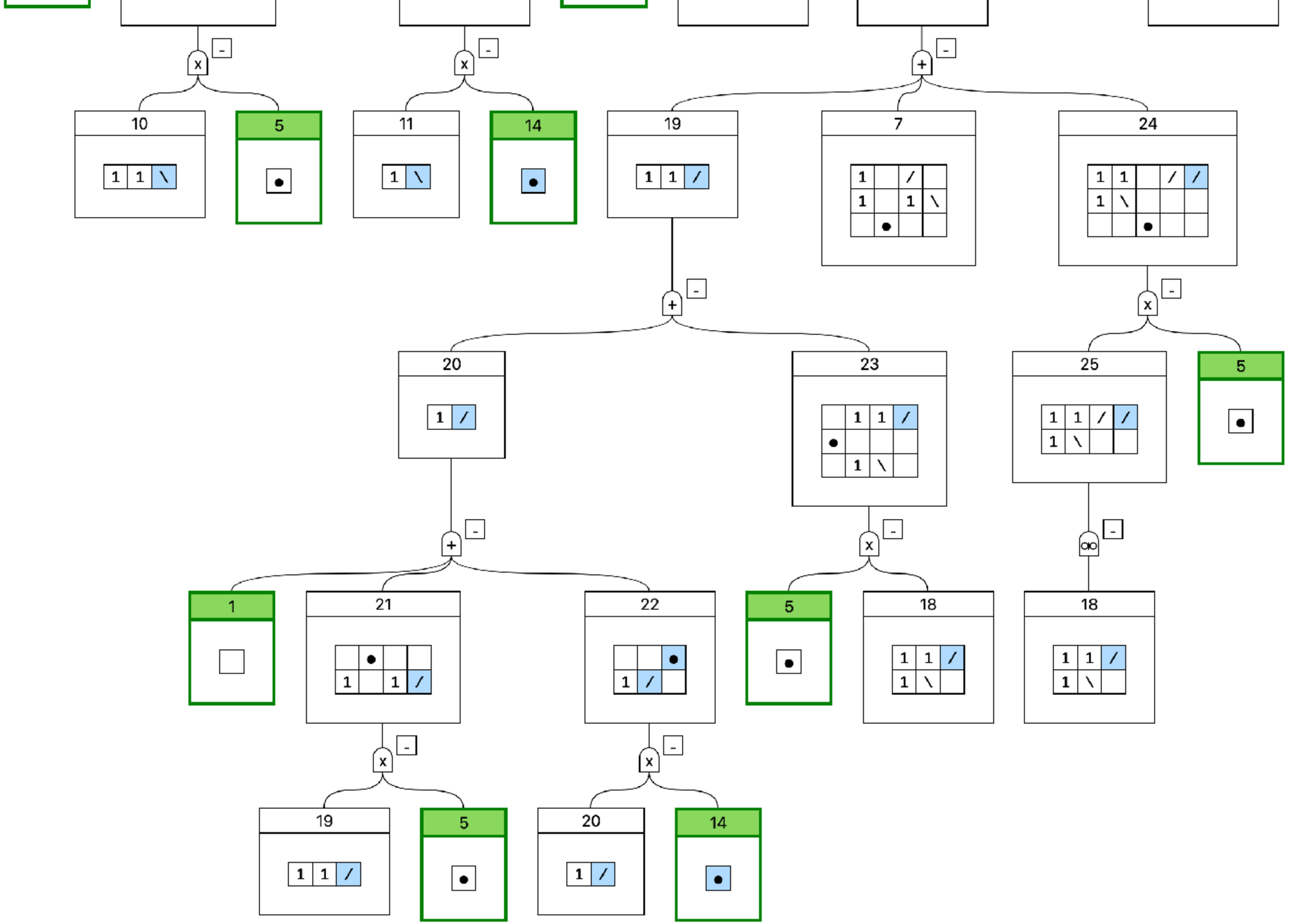
Proof Tree

Copy to clipboard: specification json | pack json

View tree on standalone page.




10
5
11
14
19
7
24
20
23
25
5
1
21
22
5
18
18
19
5
20
14

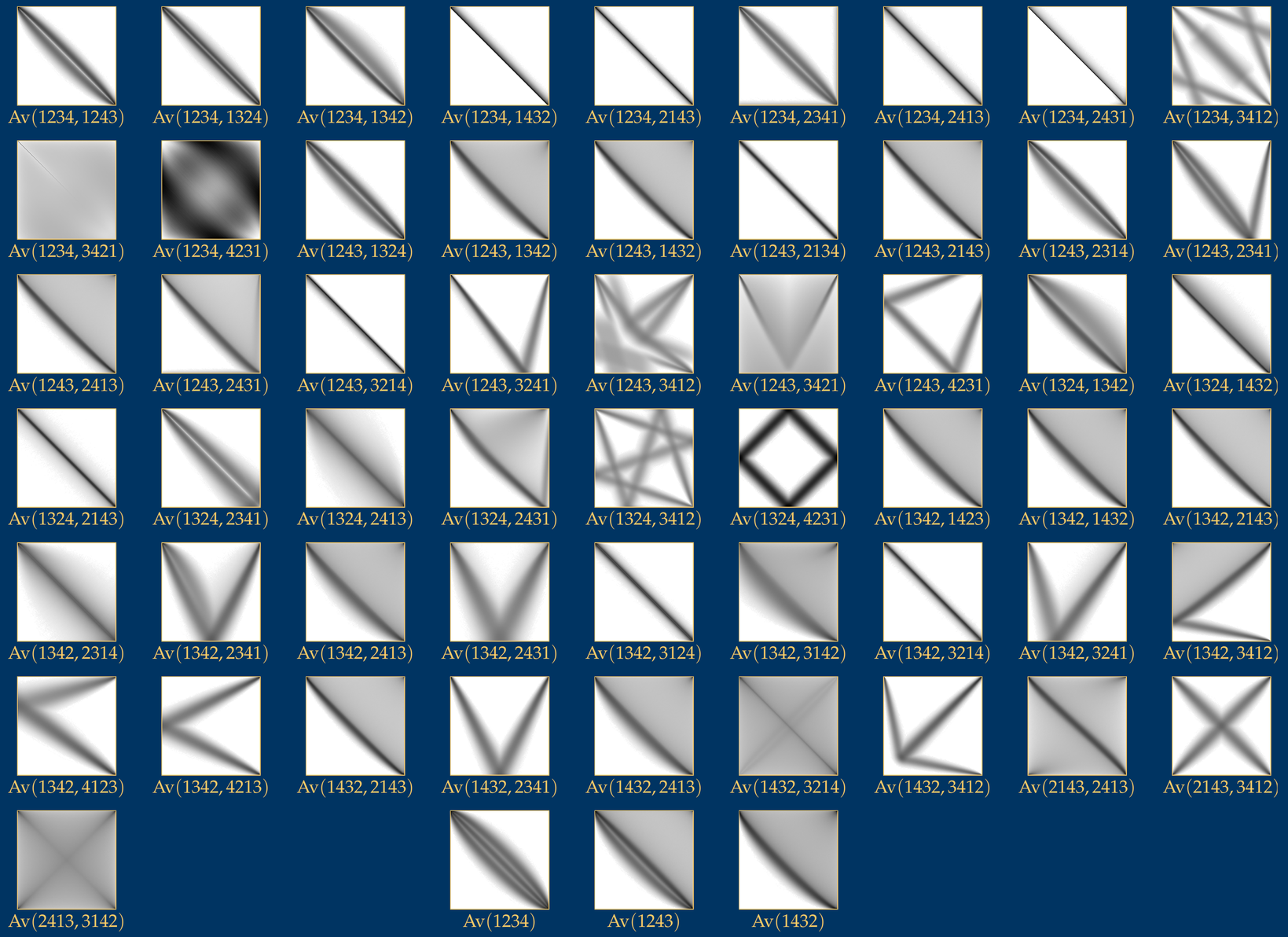

10
5
11
14
19
7
24
20
23
25
5
1
21
22
5
18
18
19
5
20
14

System of Equations

Copy 29 equations to clipboard: latex | Maple | sympy

$$
\begin{aligned}
F_0(x) &= F_1(x) + F_2(x) \\
F_1(x) &= 1 \\
F_2(x) &= F_3(x)F_5(x) \\
F_3(x) &= F_0(x) + F_4(x) \\
F_4(x) &= F_5(x)F_6(x) \\
F_5(x) &= x \\
F_6(x) &= F_{28}(x) + F_3(x) + F_7(x) \\
F_7(x) &= F_5(x)F_8(x) \\
F_8(x) &= F_9(x, 1) \\
F_9(x, y) &= F_{10}(x, y) + F_{16}(x) + F_{26}(x, y) \\
F_{10}(x, y) &= F_{11}(x, y) + F_{15}(x, y) \\
F_{11}(x, y) &= F_1(x) + F_{12}(x, y) + F_{13}(x, y) \\
F_{12}(x, y) &= F_{10}(x, y)F_5(x) \\
F_{13}(x, y) &= F_{11}(x, y)F_{14}(x, y) \\
F_{14}(x, y) &= yx \\
F_{15}(x, y) &= F_5(x)F_9(x, y) \\
F_{16}(x) &= F_{17}(x)F_5(x) \\
F_{17}(x) &= F_{18}(x, 1) \\
F_{18}(x, y) &= F_{19}(x, y) + F_{24}(x, y) + F_7(x) \\
F_{19}(x, y) &= F_{20}(x, y) + F_{23}(x, y) \\
F_{20}(x, y) &= F_1(x) + F_{21}(x, y) + F_{22}(x, y) \\
F_{21}(x, y) &= F_{19}(x, y)F_5(x) \\
F_{22}(x, y) &= F_{14}(x, y)F_{20}(x, y) \\
F_{23}(x, y) &= F_{18}(x, y)F_5(x) \\
F_{24}(x, y) &= F_{25}(x, y)F_5(x) \\
F_{25}(x, y) &= -\frac{-yF_{18}(x, y) + F_{18}(x, 1)}{-1 + y} \\
F_{26}(x, y) &= F_{27}(x, y)F_5(x) \\
F_{27}(x, y) &= -\frac{-yF_9(x, y) + F_9(x, 1)}{-1 + y} \\
F_{28}(x) &= F_{17}(x)F_5(x)
\end{aligned}
$$

Combinatorial Exploration



Combinatorial Exploration

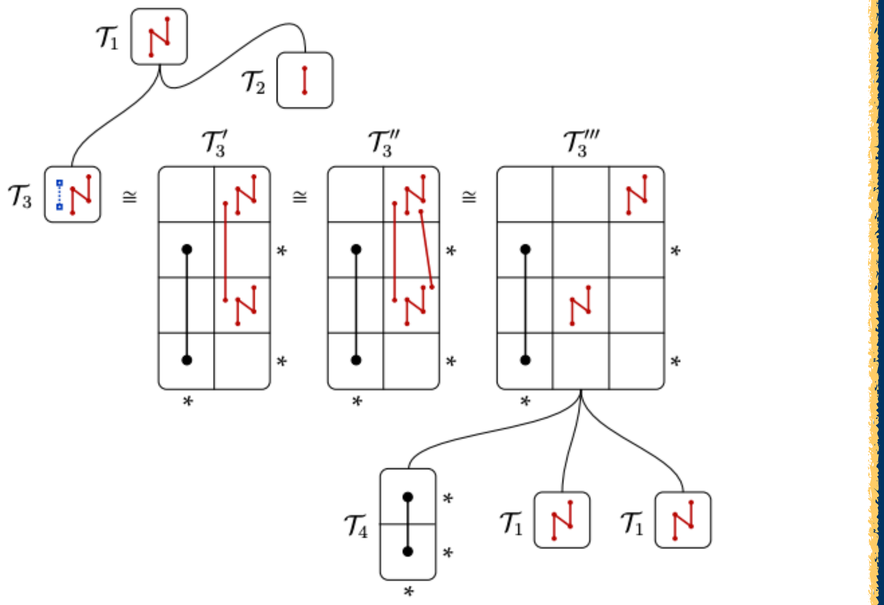


Figure 2.4: A visual representation of the proof tree for noncrossing diagrams $\mathcal{D}(\frown\frown)$.

Enumerative perspectives on chord diagrams

by

Lukas Nabergall

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Combinatorics and Optimization

Combinatorial Exploration

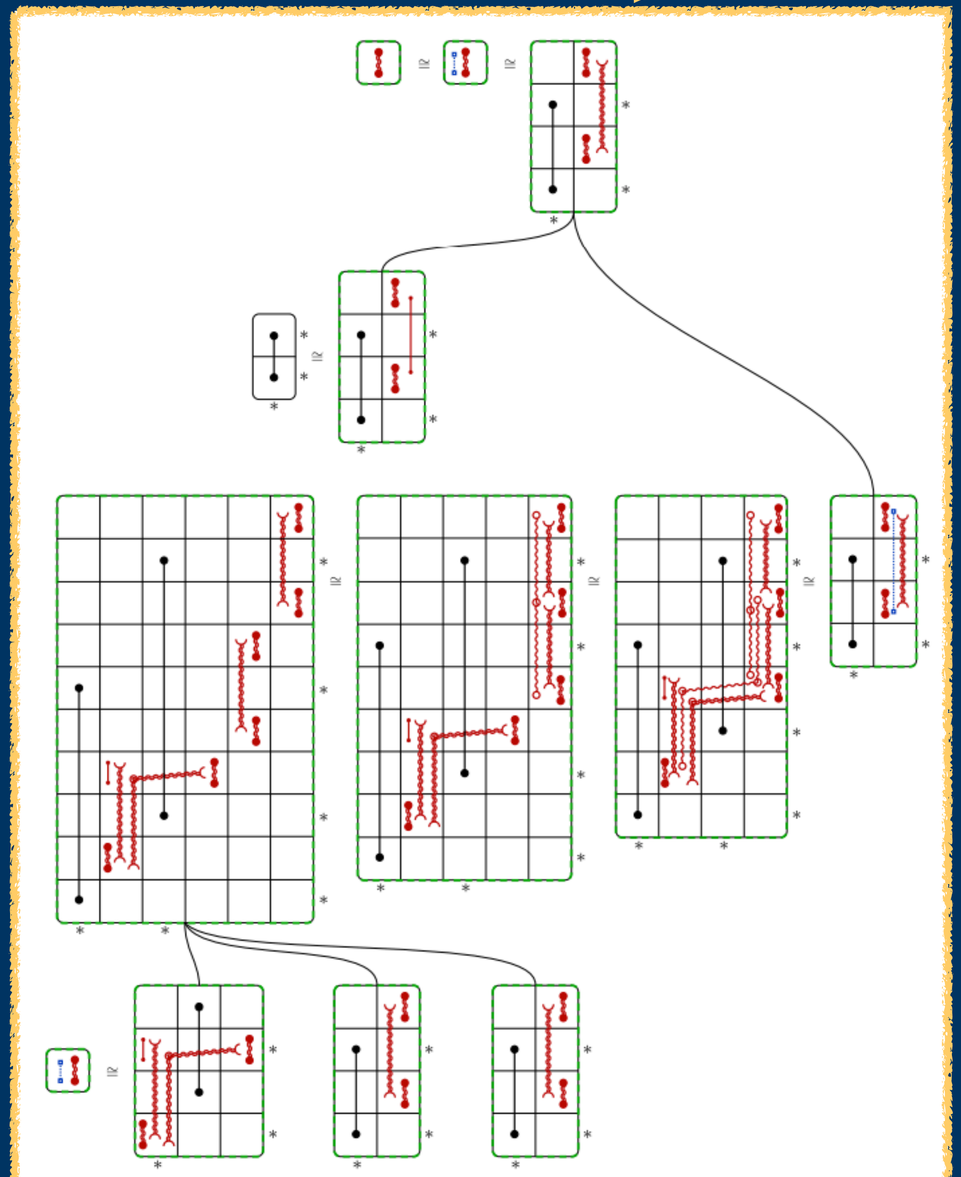


Figure 3.2: A proof tree for the class $\mathcal{C}(T_{\geqslant 3}, B_{\geqslant 3})$ of tree diagrams.

Enumerative perspectives on chord diagrams

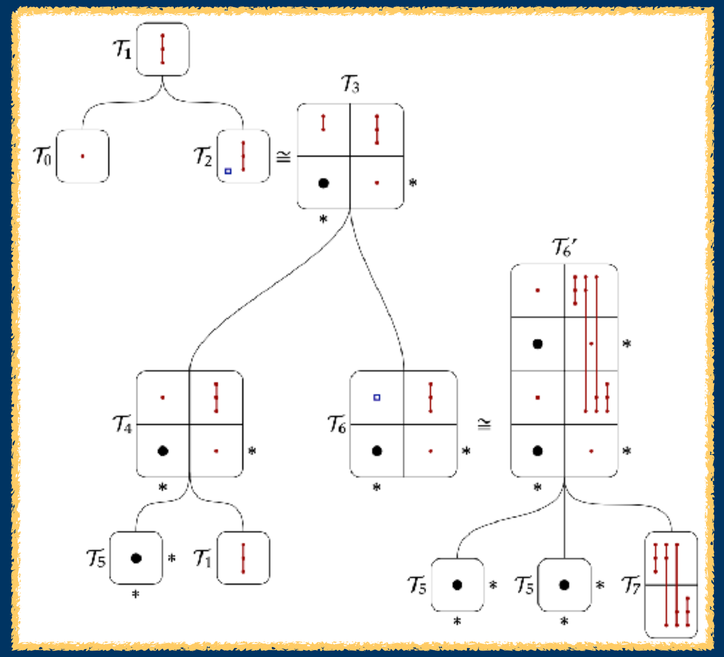
by

Lukas Nabergall

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Combinatorics and Optimization

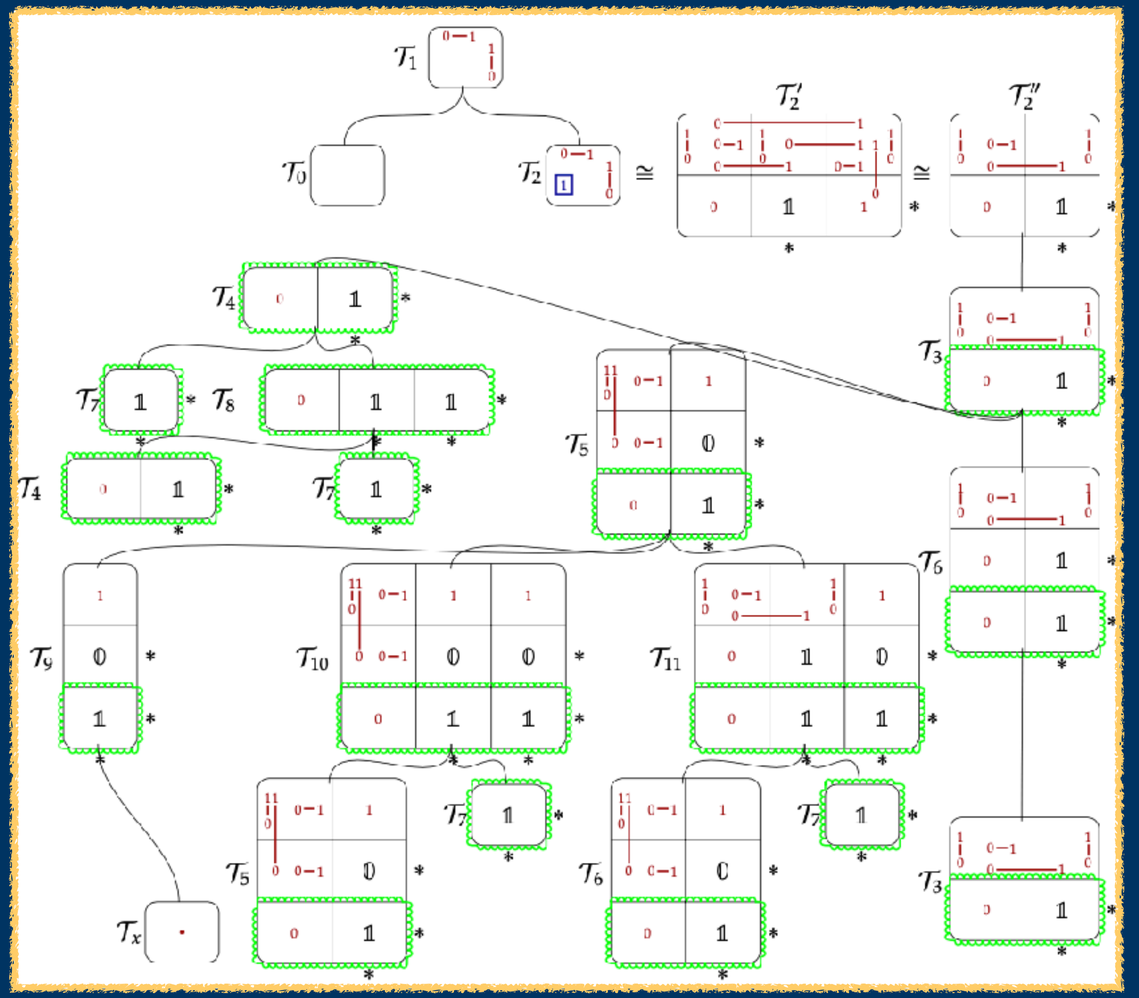
Combinatorial Exploration

Set Partitions



$$T_1(x) = 1 + (x + x^2)T_1(x) + x^3\frac{d}{dx}T_1(x)$$

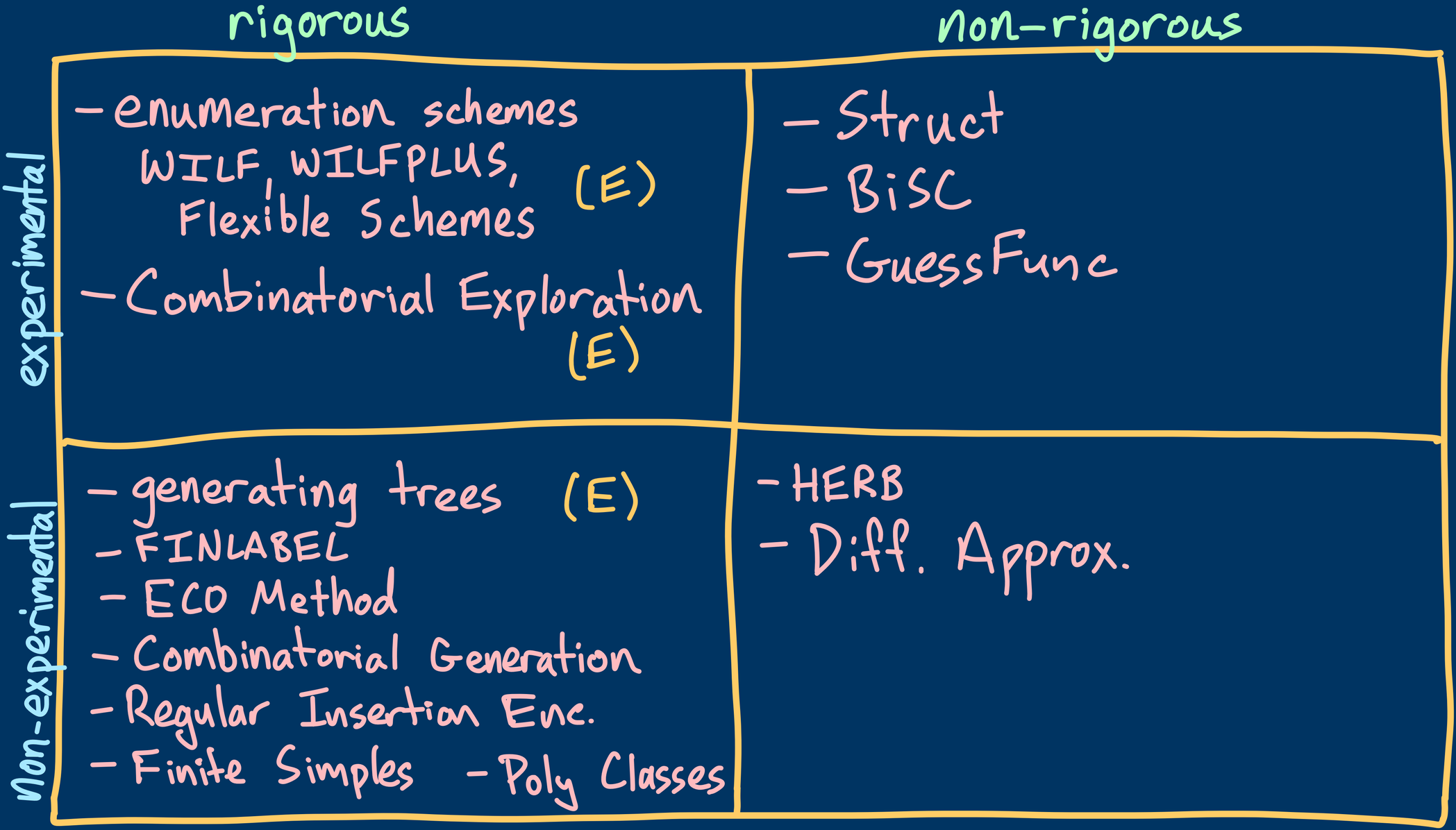
Polyominoes



$$T_1(x) = \prod_{i=1}^{\infty} \frac{1}{1 - x^i}$$

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E) - Combinatorial Exploration (E)	- Struct - BiSC
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E) - Combinatorial Exploration (E)	- Struct - BiSC - GuessFunc
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB


rigorous
non-rigorous
experimental
non-experimental
- enumeration schemes
WILF, WILFPLUS,
Flexible Schemes (E)
- Combinatorial Exploration (E)
- Struct
- BiSC
- GuessFunc
- generating trees (E)
- FINLABEL
- ECO Method
- Combinatorial Generation
- Regular Insertion Enc.
- Finite Simples
- Poly Classes
- HERB
- Diff. Approx.

PermLab

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E) - Combinatorial Exploration (E)	- Struct - BiSC - GuessFunc
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB - Diff. Approx.

PermLab

Permpy

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E) - Combinatorial Exploration (E)	- Struct - BiSC - GuessFunc
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB - Diff. Approx.

PermLab

Permuta

Permpy

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E) - Combinatorial Exploration (E)	- Struct - BiSC - GuessFunc
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB - Diff. Approx.

PermLab

Permuta

Permpy

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E) - Combinatorial Exploration (E)	- Struct - BiSC - GuessFunc
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB - Diff. Approx.

TTDD

PermLab

Permuta

Permpy

	rigorous	non-rigorous
experimental	- enumeration schemes WILF, WILFPLUS, Flexible Schemes (E) - Combinatorial Exploration (E)	- Struct - BiSC - GuessFunc
non-experimental	- generating trees (E) - FINLABEL - ECO Method - Combinatorial Generation - Regular Insertion Enc. - Finite Simples - Poly Classes	- HERB - Diff. Approx.

TTDD

Kuszmaul

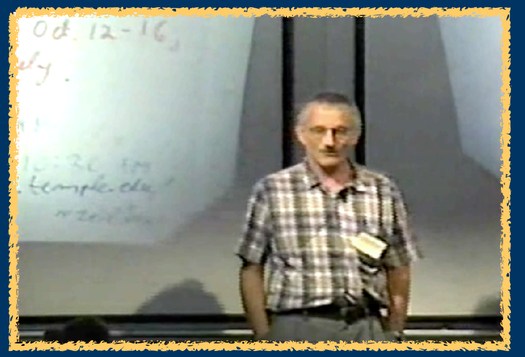
28 years ago...

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998



Apology. The success rate of the present method, in its present state, is somewhat disappointing. Ekhad was able to reproduce the classical cases and a few new ones, but for most patterns and sets of patterns, it failed to find a scheme (defined below) of reasonable depth. But the present framework for setting up a scheme could be modified and extended in various ways. We do believe that an appropriate enhancement of the present method would yield, if not a 100% success rate, at least close to it.

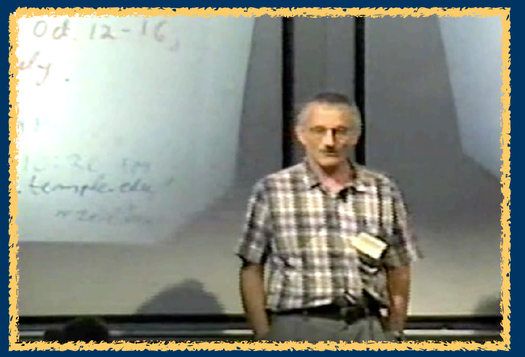
28 years ago...

Enumeration Schemes and, More Importantly, Their Automatic Generation

Doron Zeilberger*

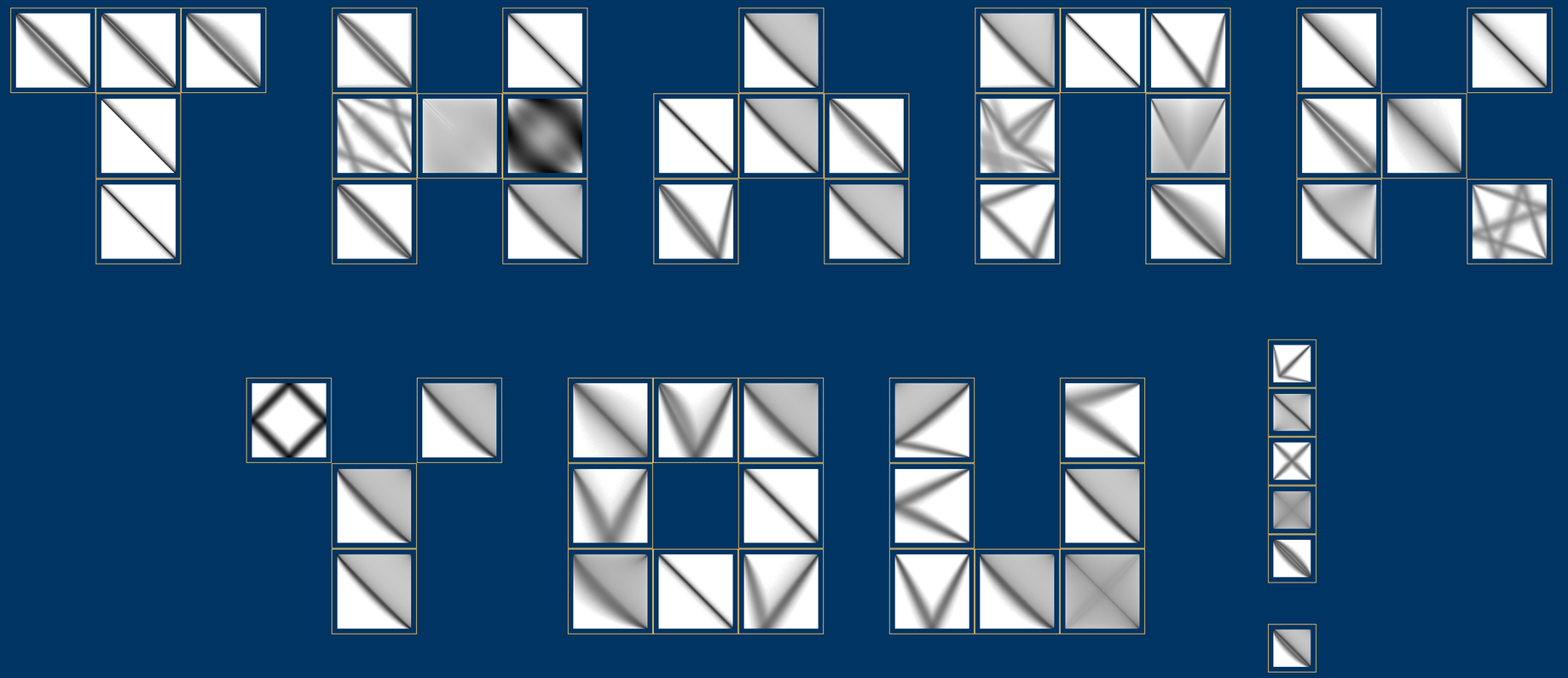
Department of Mathematics, Temple University, Philadelphia, PA 19122, USA
zeilberg@math.temple.edu, http://www.math.temple.edu/~zeilberg

Received May 27, 1998



Apology. The success rate of the present method, in its present state, is somewhat disappointing. Ekhad was able to reproduce the classical cases and a few new ones, but for most patterns and sets of patterns, it failed to find a scheme (defined below) of reasonable depth. But the present framework for setting up a scheme could be modified and extended in various ways. We do believe that an appropriate enhancement of the present method would yield, if not a 100% success rate, at least close to it.

A bit optimistic, but progress continues!


Thanks
you!