Generating all modular lattices of a given size

ADAM 2013

Nathan Lawless

Chapman University

June 6-8, 2013

Outline

- Introduction to Lattice Theory: Modular Lattices
- The Objective: Generating and Counting Modular Lattices
- First Approach: From Finite Lattices
- Second Approach: From Distributive Lattices
- Results
- Conclusion

What are lattices?

First, let a **partially ordered set or poset** be a binary relation "$\leq$" over a set P such that $\forall a, b, c \in P$ satisfies:

- Reflexivity: $a \leq a$.
- Antisymmetry: if $a \leq b$ and $b \leq a$, then $a = b$.
- Transitivity: if $a \leq b$ and $b \leq c$, then $a \leq c$.

What are lattices?

First, let a **partially ordered set or poset** be a binary relation "$\leq$" over a set P such that $\forall a, b, c \in P$ satisfies:

- Reflexivity: $a \leq a$.
- Antisymmetry: if $a \leq b$ and $b \leq a$, then $a = b$.
- Transitivity: if $a \leq b$ and $b \leq c$, then $a \leq c$.

Then, we say a partially ordered set $(L, \leq)$ is a **lattice** if it satisfies the following axioms:

- For any two elements a and b of L, there exists a least upper bound, refered to as the *join* $a \vee b$.
- For any two elements a and b of L, there exists a greatest lower bound, refered to as the *meet* $a \wedge b$.

Example 1

- For any set X, all the subsets of the power set $P(X)$ form a lattice with the order relation $A \leq B$ meaning A is a subset of B ($A \subseteq B$). Then, $A \vee B = A \cup B$ and $A \wedge B = A \cap B$.
  Example: $P(\{a, b, c\}) =$
  $\{\{a, b, c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a\}, \{b\}, \{c\}, \emptyset\}$.

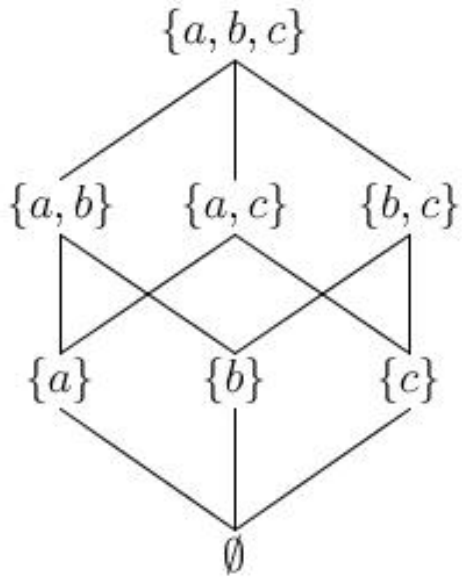


Figure 1: Power Set Lattice of {a,b,c}.

Example 2

- However, the following poset is **not** a lattice since a and b do not have a least upper bound, as we do not know which is smaller among c and d.

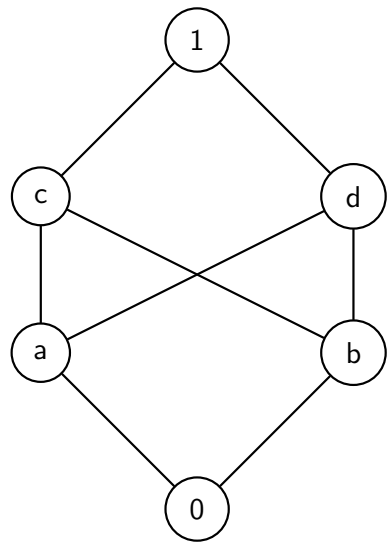


Figure 2. Example of a poset which is not a lattice.

Distributive Lattices

- A **distributive lattice** D is a lattice that satisfies the distributive law $x, y, z \in D$:
  $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$.

Distributive Lattices

- A **distributive lattice** D is a lattice that satisfies the distributive law $x, y, z \in D$:
  $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$.
- An alternative way to view distributive lattices is by **Birkhoff's Theorem**: *L is a nondistributive lattice iff M_3 or N_5 can be embedded into L.*

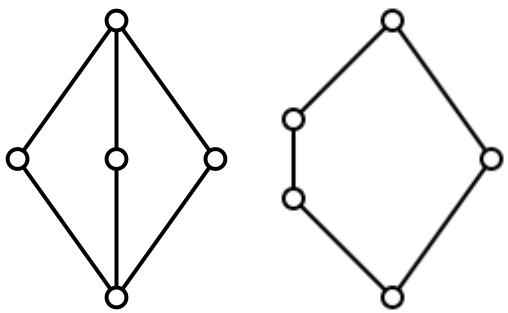


Figure 3: M_3 and N_5.

Distributive Lattices

- A **distributive lattice** D is a lattice that satisfies the distributive law $x, y, z \in D$:
  $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$.
- An alternative way to view distributive lattices is by **Birkhoff's Theorem**: *L is a nondistributive lattice iff M_3 or N_5 can be embedded into L.*

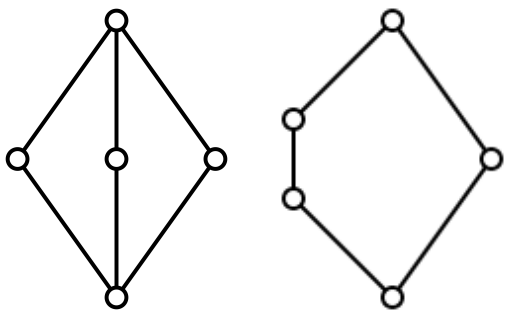


Figure 3: M_3 and N_5.

- Examples of distributive lattices are:

Distributive Lattices

- A **distributive lattice** D is a lattice that satisfies the distributive law $x, y, z \in D$:
  $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$.
- An alternative way to view distributive lattices is by **Birkhoff's Theorem**: *L is a nondistributive lattice iff M_3 or N_5 can be embedded into L.*

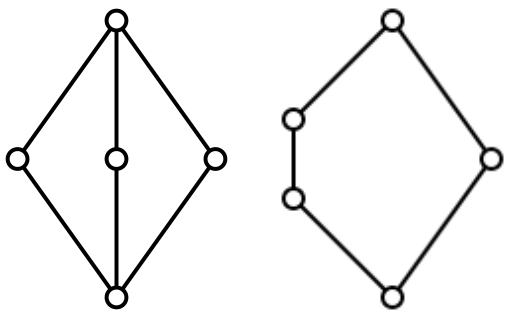


Figure 3: M_3 and N_5.

- Examples of distributive lattices are:
  - Boolean algebras.

Distributive Lattices

- A **distributive lattice** D is a lattice that satisfies the distributive law $x, y, z \in D$:
  $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$.
- An alternative way to view distributive lattices is by **Birkhoff's Theorem**: *L is a nondistributive lattice iff M_3 or N_5 can be embedded into L.*

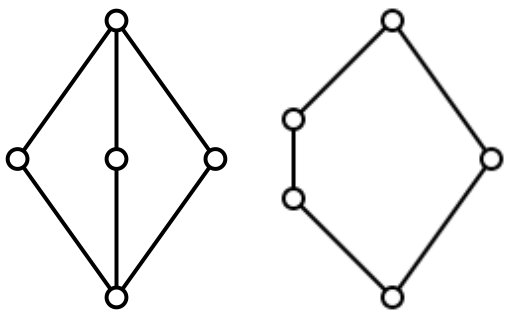


Figure 3: M_3 and N_5.

- Examples of distributive lattices are:
  - Boolean algebras.
  - Natural numbers with the gratest common divisor as meet and the least common multiple as join.

Modular Lattices

- A **modular lattice** M is a lattice that satisfies the modular law $x, y, z \in M$:
  $(x \wedge y) \vee (y \wedge z) = y \wedge [(x \wedge y) \vee z)]$.

Modular Lattices

- A **modular lattice** M is a lattice that satisfies the modular law $x, y, z \in M$:
  $(x \wedge y) \vee (y \wedge z) = y \wedge [(x \wedge y) \vee z)]$.
- An alternative way to view modular lattices is by **Dedekind's Theorem**: *L is a nonmodular lattice iff N_5 can be embedded into L.*

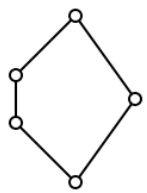


Figure 4: N_5.

Modular Lattices

- A **modular lattice** M is a lattice that satisfies the modular law $x, y, z \in M$:
  $(x \wedge y) \vee (y \wedge z) = y \wedge [(x \wedge y) \vee z)]$.
- An alternative way to view modular lattices is by **Dedekind's Theorem**: *L is a nonmodular lattice iff N_5 can be embedded into L.*

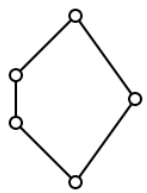


Figure 4: N_5.

- All distributive lattices are modular lattices.

Modular Lattices

- A **modular lattice** M is a lattice that satisfies the modular law $x, y, z \in M$:
  $(x \wedge y) \vee (y \wedge z) = y \wedge [(x \wedge y) \vee z)]$.
- An alternative way to view modular lattices is by **Dedekind's Theorem**: *L is a nonmodular lattice iff N_5 can be embedded into L.*

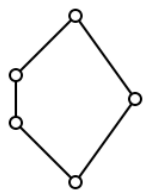


Figure 4: N_5.

- All distributive lattices are modular lattices.
- Examples of modular lattices are:

Modular Lattices

- A **modular lattice** M is a lattice that satisfies the modular law $x, y, z \in M$:
  $(x \wedge y) \vee (y \wedge z) = y \wedge [(x \wedge y) \vee z)]$.
- An alternative way to view modular lattices is by **Dedekind's Theorem**: *L is a nonmodular lattice iff N_5 can be embedded into L.*

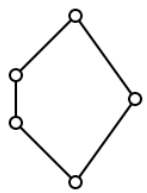


Figure 4: N_5.

- All distributive lattices are modular lattices.
- Examples of modular lattices are:
  - Lattices of subspaces of vector spaces.

Modular Lattices

- A **modular lattice** M is a lattice that satisfies the modular law $x, y, z \in M$:
  $(x \wedge y) \vee (y \wedge z) = y \wedge [(x \wedge y) \vee z)]$.
- An alternative way to view modular lattices is by **Dedekind's Theorem**: *L is a nonmodular lattice iff N_5 can be embedded into L.*

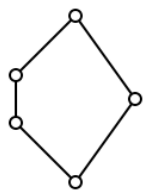


Figure 4: N_5.

- All distributive lattices are modular lattices.
- Examples of modular lattices are:
  - Lattices of subspaces of vector spaces.
  - Lattices of ideals of a ring.

Modular Lattices

- A **modular lattice** M is a lattice that satisfies the modular law $x, y, z \in M$:
  $(x \wedge y) \vee (y \wedge z) = y \wedge [(x \wedge y) \vee z)]$.
- An alternative way to view modular lattices is by **Dedekind's Theorem**: *L is a nonmodular lattice iff N_5 can be embedded into L.*

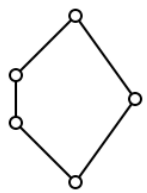


Figure 4: N_5.

- All distributive lattices are modular lattices.
- Examples of modular lattices are:
  - Lattices of subspaces of vector spaces.
  - Lattices of ideals of a ring.
  - Lattices of normal subgroups of a group.

Our Objective

We wish to come up with an algorithm which can efficiently generate all possible finite modular lattices of a given size n up to isomorphism.

Why is this important?

1. The generated modular lattices can provide a tool for other scientists to verify conjectures and/or find counterexamples.
2. Better understanding of modular lattices.
3. Discovering new structural properties of modular lattices.

Counting Finite Lattices

Heitzig and Reinhold [2000] developed an orderly algorithm to enumerate all finite lattices and used it to count the number of lattices up to size 18. To explain their algorithm, we give some definitions related to posets and lattices:

- We say that b is a **cover** of a if $a < b$ and there is no element c such that $a < c < b$, and denote this by $a \prec b$.

Counting Finite Lattices

Heitzig and Reinhold [2000] developed an orderly algorithm to enumerate all finite lattices and used it to count the number of lattices up to size 18. To explain their algorithm, we give some definitions related to posets and lattices:

- We say that b is a **cover** of a if $a < b$ and there is no element c such that $a < c < b$, and denote this by $a \prec b$.
- We call $\uparrow A = \{x \in L \mid a \leq x \text{ for some } a \in A\}$ the **upper set** of A.

Counting Finite Lattices

Heitzig and Reinhold [2000] developed an orderly algorithm to enumerate all finite lattices and used it to count the number of lattices up to size 18. To explain their algorithm, we give some definitions related to posets and lattices:

- We say that b is a **cover** of a if $a < b$ and there is no element c such that $a < c < b$, and denote this by $a \prec b$.
- We call $\uparrow A = \{x \in L \mid a \leq x \text{ for some } a \in A\}$ the **upper set** of A.
- The set of all maximal elements in L is called the first level of L ($lev_1(L)$). The **(m+1)-th level** of P can be recursively defined by $lev_{m+1}(L) = lev_1(L - \bigcup_{i=1}^{m} lev_i(L))$.

Counting Finite Lattices

Heitzig and Reinhold [2000] developed an orderly algorithm to enumerate all finite lattices and used it to count the number of lattices up to size 18. To explain their algorithm, we give some definitions related to posets and lattices:

- We say that b is a **cover** of a if $a < b$ and there is no element c such that $a < c < b$, and denote this by $a \prec b$.
- We call $\uparrow A = \{x \in L \mid a \leq x \text{ for some } a \in A\}$ the **upper set** of A.
- The set of all maximal elements in L is called the first level of L ($lev_1(L)$). The **(m+1)-th level** of P can be recursively defined by $lev_{m+1}(L) = lev_1(L - \bigcup_{i=1}^{m} lev_i(L))$.
- An **antichain** is a subset of L in which any two elements in the subset are incomparable.

Counting Finite Lattices

Heitzig and Reinhold [2000] developed an orderly algorithm to enumerate all finite lattices and used it to count the number of lattices up to size 18. To explain their algorithm, we give some definitions related to posets and lattices:

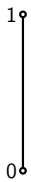
- We say that b is a **cover** of a if $a < b$ and there is no element c such that $a < c < b$, and denote this by $a \prec b$.
- We call $\uparrow A = \{x \in L \mid a \leq x \text{ for some } a \in A\}$ the **upper set** of A.
- The set of all maximal elements in L is called the first level of L ($lev_1(L)$). The **(m+1)-th level** of P can be recursively defined by $lev_{m+1}(L) = lev_1(L - \bigcup_{i=1}^{m} lev_i(L))$.
- An **antichain** is a subset of L in which any two elements in the subset are incomparable.
- A **lattice antichain** A is an antichain in which for any $a, b \in \uparrow A$, $a \wedge b \in \uparrow A \cup \{0\}$.

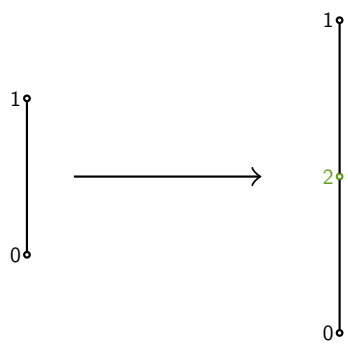
Counting Finite Lattices (continued)

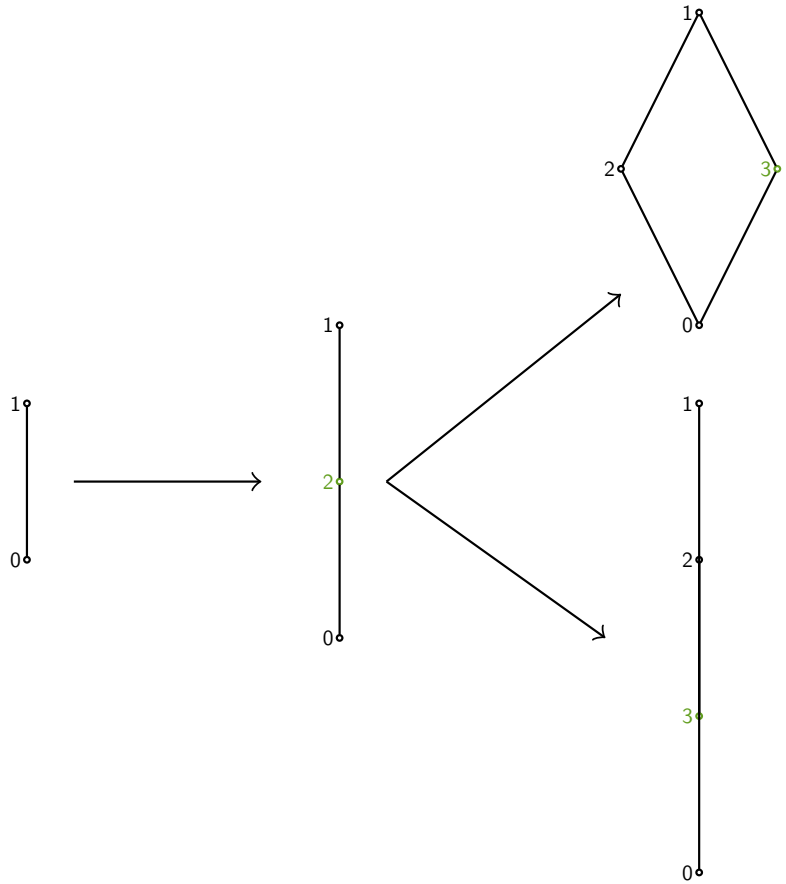
Using these definitions, a recursive algorithm can be formulated that generates for a given natural number $n \geq 2$ exactly all canonical lattices up to n elements starting with the two element lattice, where L^A is the lattice obtained from L by adding a new element with the subset $A \in L$ as a cover:

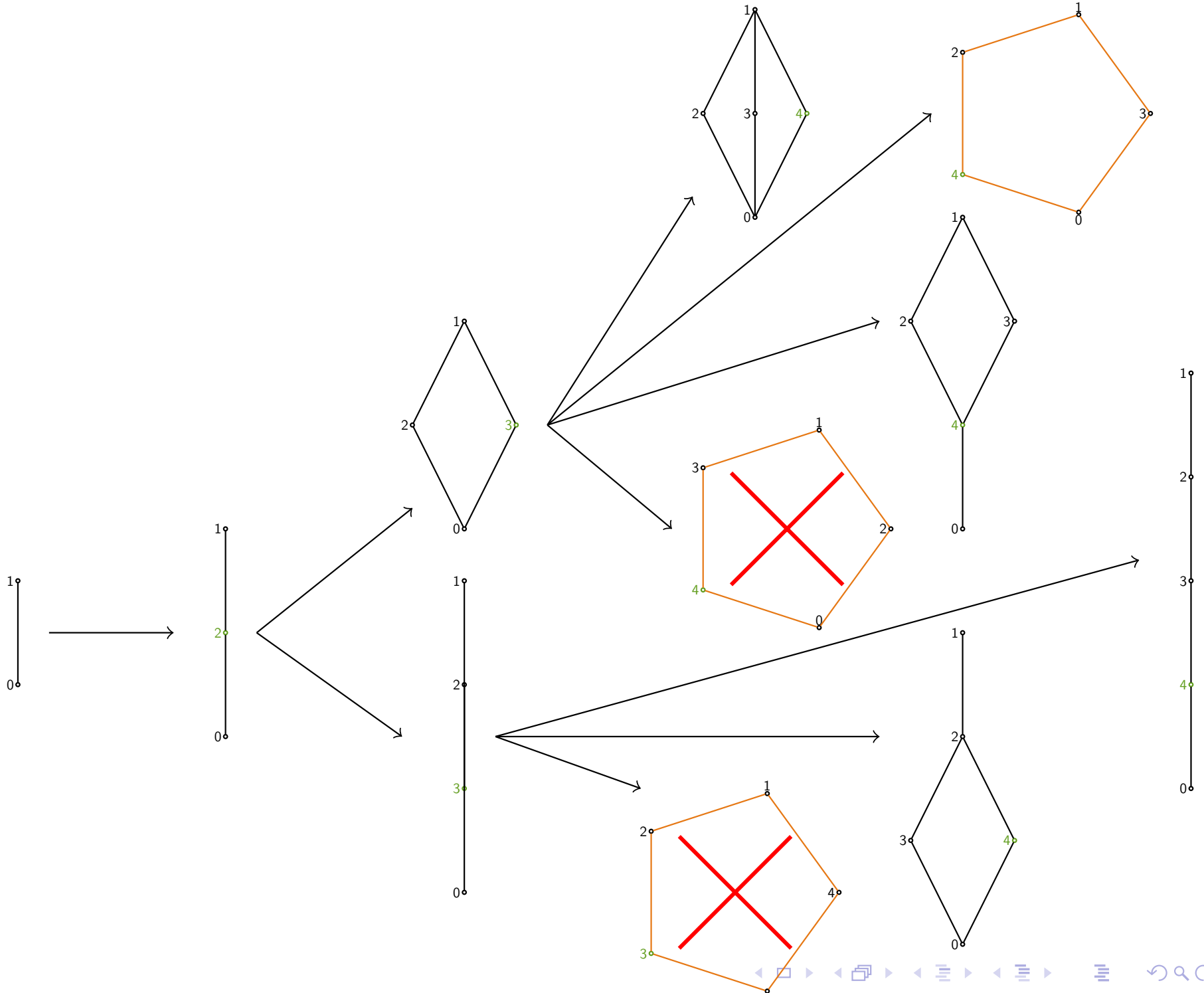
```
next_lattice(integer m, canonical m-lattice L)
begin
    if m < n then
        for each lattice-antichain A of L do
            if L^A is a canonical lattice then
                next_lattice (m + 1, L^A)
    if m = n then output L
end
```

Most of the time in this algorithm is spent in testing if L^A is canonical, as all permutations of L^A have to be checked to see if L^A is canonical. However, some properties allow to reduce the number of checkings needed.


1
0


1
0
1
2
0


1
0
1
2
0
1
2
3
0
1
2
3
0


1
0
1
2
0
1
2
3
0
1
2
3
4
0
1
2
3
4
0
1
2
3
4
0
1
2
3
4
0
1
2
3
4
0
1
2
3
4
0
1
2
3
4
0

Counting Finite Lattices: Modular Lattices

This algorithm can be modified such that when a lattice of size n is generated, the algorithm checks if it is modular.
Some properties can be used to reduce the number of checkings:

- Lemma: If the lattice L' with the elements in the third to last level of L as atoms is non-modular, then all descendant lattices of L are non-modular.
- Lemma: If there are any two elements $a, b \in A$ contained in different levels of L, then all descendants are non-modular.
- Lemma: When constructing all modular lattices, it suffices to use lattice antichains that are subsets of the bottom level or subsets of the second lowest level. Subsets of the bottom level need only be used if there are no atoms in the second lowest level.

Dealing with Isomorphisms

- The canonical checking in the first algorithm is of order $O(n!)$,
- The time spent in this check can be drastically reduced by using the method of *generation by canonical construction path* introduced by McKay.
- This method consists on generating the lattices in a way such that it is not necessary to test if the lattice has the lowest weight among all possible permutations.
- Instead, it uses a defined path construction incorporating Nauty canonical labeling.

```
procedure scan(X : labelled object, n : integer)
    output X
        for each orbit A of the action of Aut(X) on U(X) do
            select any X^ ∈ A
            if f'(X^) ≠ ∅ then
                select any Y^v ∈ f'(X^)
                if o(Y) ≤ n and Y^v ∈ m(Y) then scan(Y,n)
endprocedure
```

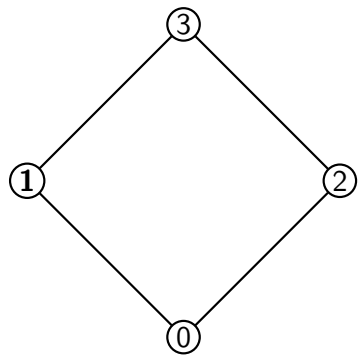
Counting Distributive Lattices

A second and cheaper approach comes from an orderly algorithm used by Erné, Heitzig and Reinhold [2002] to generate and count the distributive lattices up to n=49; a subclass of modular lattices.

- Consider a distributive lattice $D = (k, \leq)$ of size k and choose an element $z \in D$.
- Find the principal filter $I = \uparrow z$.
- The principal filter I is "doubled" to generate the set $I^{\uparrow} = \{k, \ldots, k + |I| - 1\}$ with an isomorphism mapping $\psi : I^{\uparrow} \to I$.
- Define a new relation $\leq^{\uparrow}$ on the set $k + |I|$ by: $x \leq^{\uparrow} y \Leftrightarrow (x, y < k$ and $x \leq y)$ or $(x, y \geq k$ and $\psi(x) \leq \psi(y))$ or $(x < k \leq y$ and $x \leq \psi(y))$.
- The generated lattice $D \uparrow z := (k + |I|, \leq^{\uparrow})$ is again a distributive lattice.
- **Theorem:** *Every distributive lattice of finite cardinality is isomorphic to a lattice of the form* $D_0 \uparrow z_1 \uparrow \ldots \uparrow z_n$ *with* $|D_0| = 1$ *and a sequence with* $0 = z_1 \leq z_2 \leq \ldots \leq z_n$.

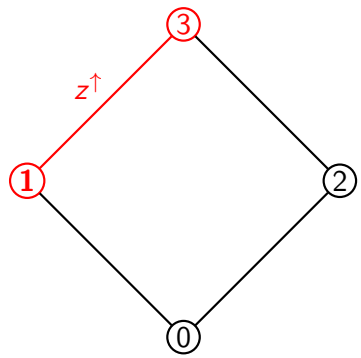
Counting Distributive Lattices: Example

First, we chose an element, say $z = 1$.



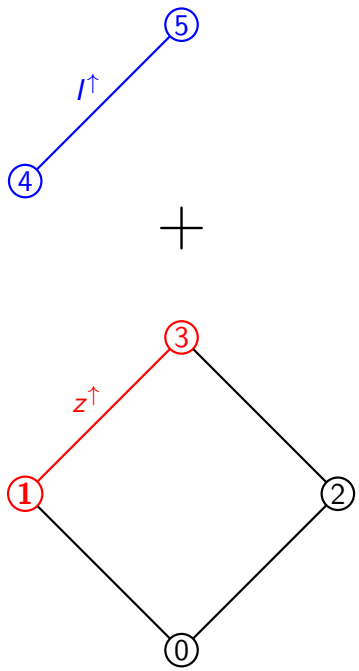
Counting Distributive Lattices: Example

For the selected z, we calculate $I = {\uparrow} z = \{1, 3\}$.



Counting Distributive Lattices: Example

Next, we "double" I, obtaining $I^{\uparrow} = \{4, 5\}$.



Counting Distributive Lattices: Example

Then, we connect the points in $I^{\uparrow}$ to their corresponding images under ψ and update the values of $\leq^{\uparrow}$.

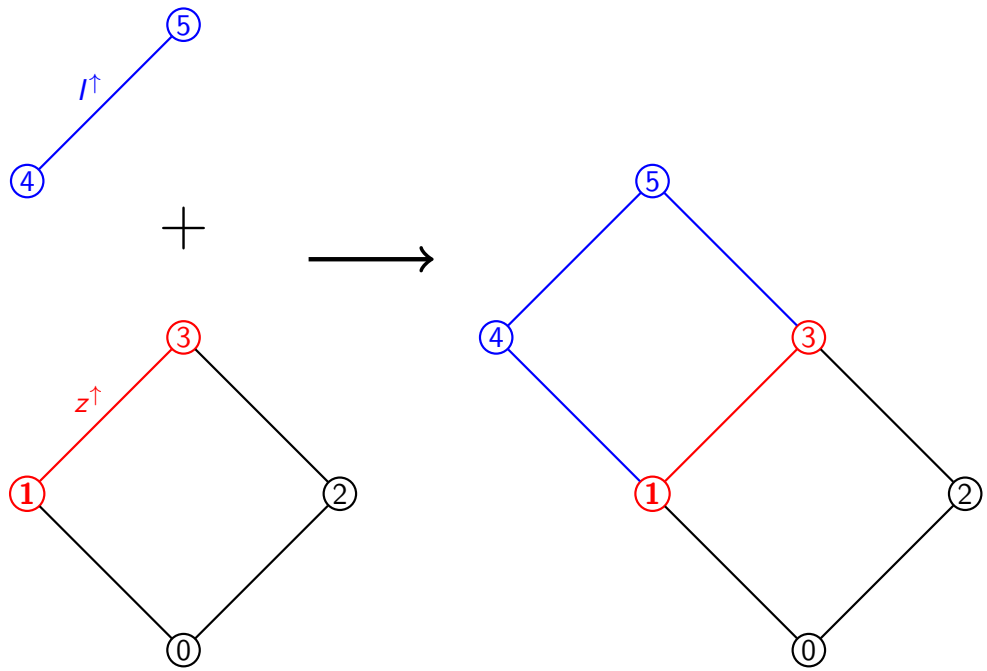
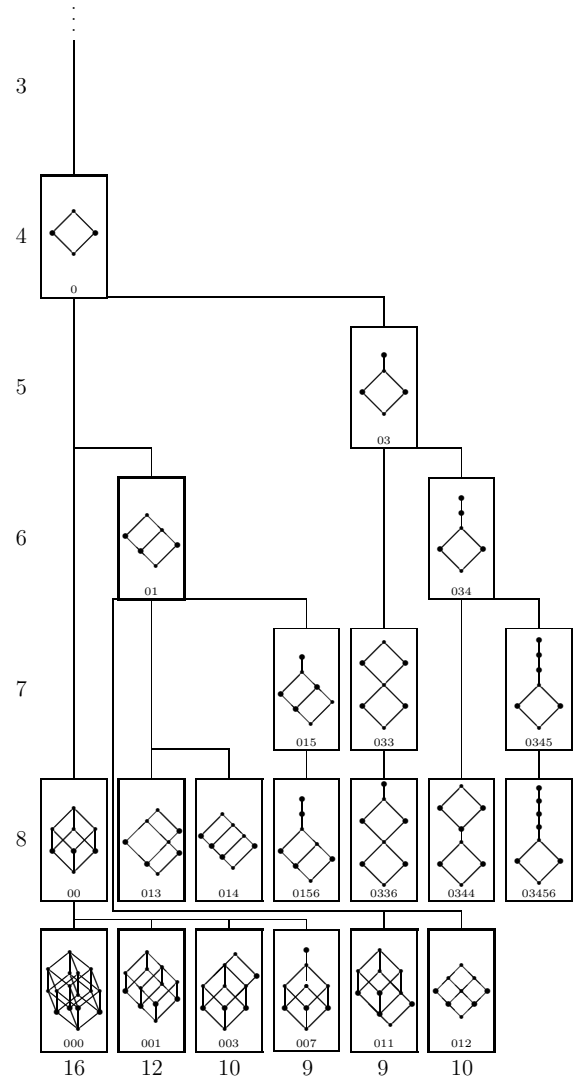
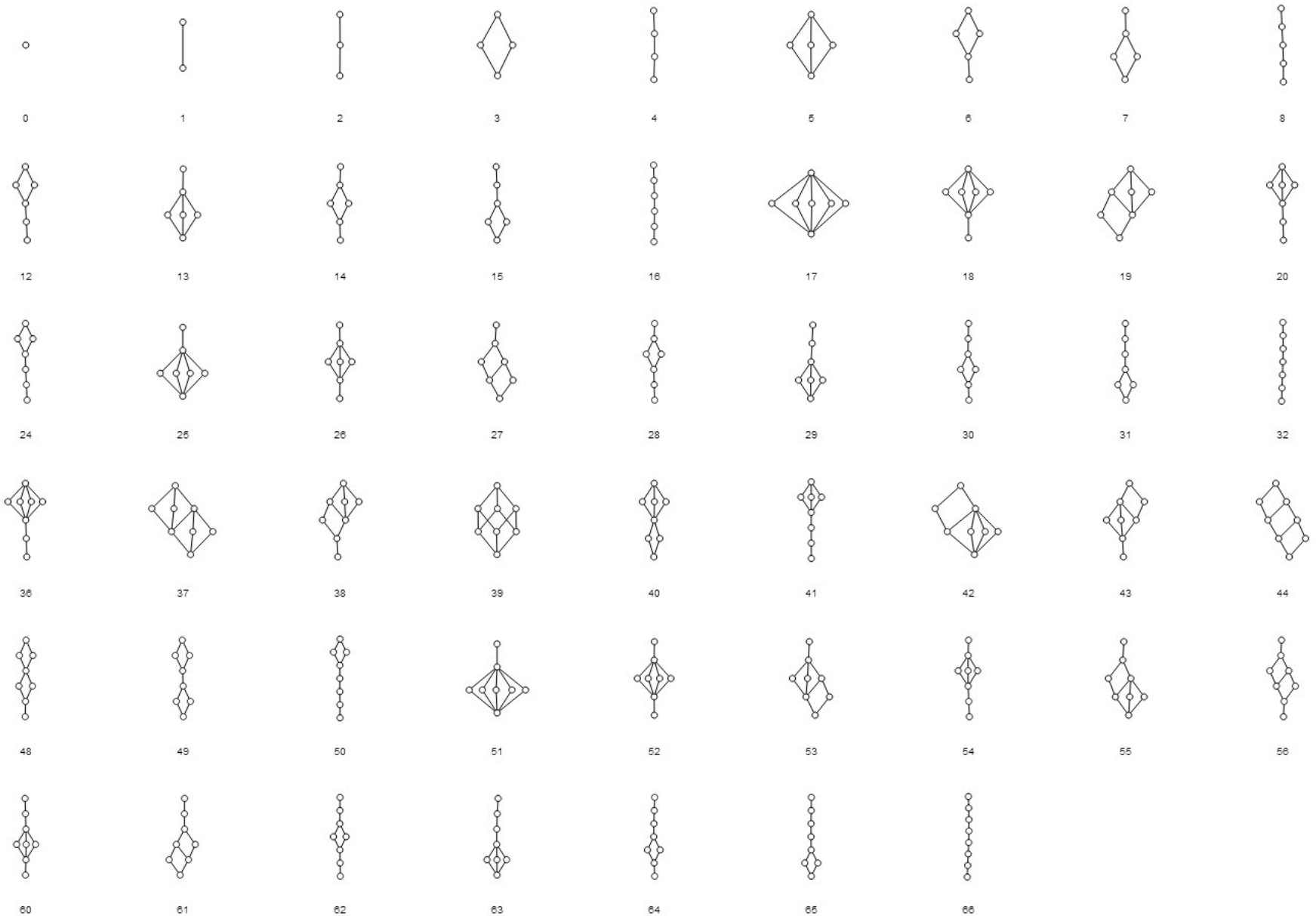


Figure 1: A handy network of distributive lattices of size $\leqslant 8$ or height $\leqslant 4$



Modular Lattices: Example



Counting Distributive Lattices: Extension

- The following figure shows a distributive (and modular) lattice (left) and a non-distributive modular lattice (right).

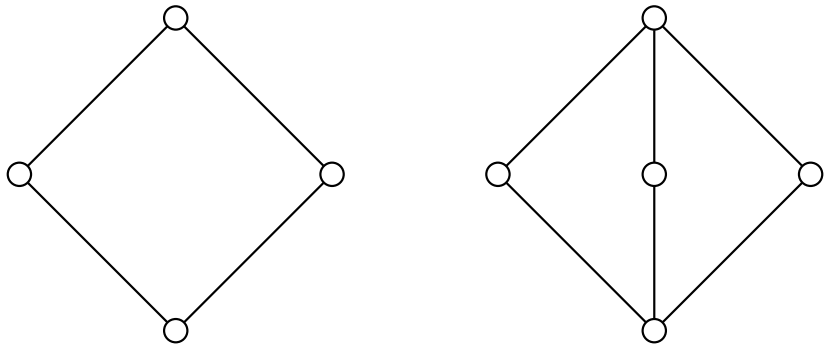


Figure 5

- Idea: Must find a way to insert vertices such that the extended poset is still a lattice and is modular.
- **Theorem:** *Let $a, b \in M$ for a modular lattice M such that $a < b$ and the longest chain from a to b is of length 3 and Covers(a) = Co-covers(b), then the lattice produced by inserting a new point between a and b is a modular lattice.*

Counting Distributive Lattices: Extension

However, with the doubling and adding point operations, there are still some modular lattices which are not generated. For example, the **subspace lattices of projective geometries**, a special type of geometric lattice. We add these as building blocks.

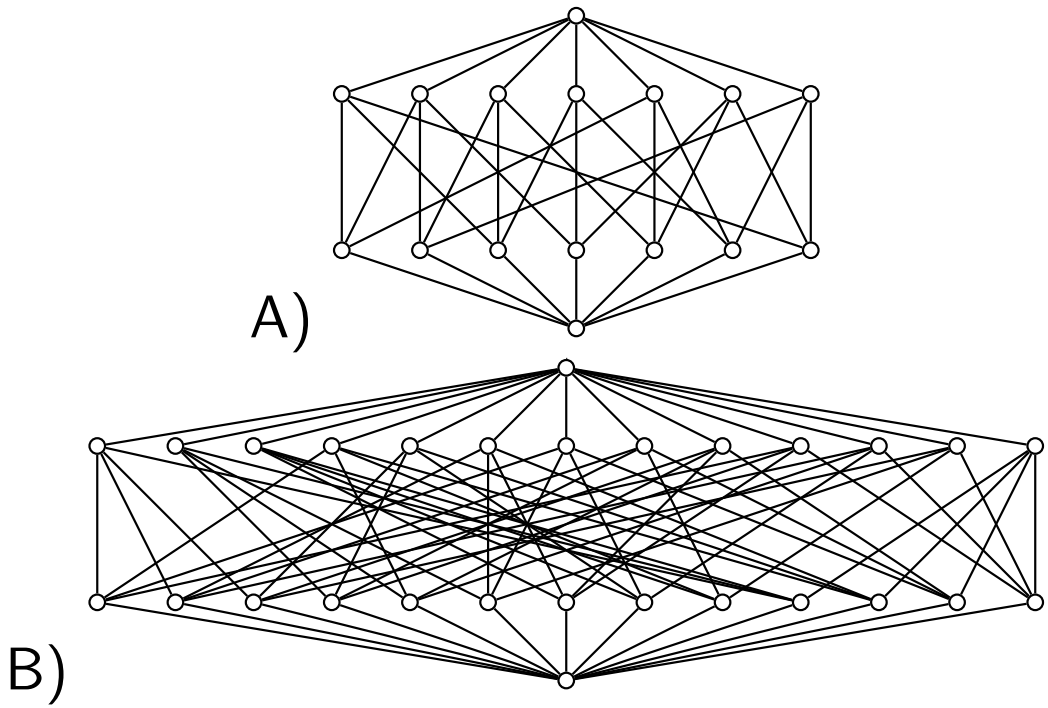


Figure : Geometric lattices of subspaces of A) $\mathbb{Z}_2^3$ ($n = 16$) and B) $\mathbb{Z}_3^3$ ($n = 28$).

Results

n	Lattices	Mod. Lattices	Alg.1(min)	Alg.1 Mod.(min)	Alg.2(min)
1	1	1	0	0	0
2	1	1	0	0	0
3	1	1	0	0	0
4	2	2	0	0	0
5	5	4	0	0	0
6	15	8	0	0	0
7	53	16	0	0	0
8	222	34	0	0	0
9	1,078	72	0	0	0
10	5,994	157	0	0	0
11	37,622	343	0	0	0
12	262,776	766	0.8	0	0.08
13	2,018,305	**1,718**	17	0.2	0.3
14	16,873,364	**3,899**	308	1	1.9
15	152,233,518	**8,898**	5320	5	10
16	1,471,613,387	**20,475**	–	30	59
17	15,150,569,446	**47,321**	–	130	–
18	165,269,824,761	**110,024**	–	510	–
19	–	**256,791**	–	2,000	–

Table : Number of lattices and modular lattices up to isomorphism from n=1 to n=19 with corresponding times of the different algorithms. New numbers are in bold.

Conclusion

1. The number of unique modular lattices up to isomorphism has been counted up to size $n = 19$ (they had previously been counted up to $n = 12$).

Conclusion

1. The number of unique modular lattices up to isomorphism has been counted up to size $n = 19$ (they had previously been counted up to $n = 12$).
2. The algorithm used by Heitzig and Reinhold to generate all finite lattices has been improved with McKay's isomorph-free exhaustive generation.

Conclusion

1. The number of unique modular lattices up to isomorphism has been counted up to size $n = 19$ (they had previously been counted up to $n = 12$).
2. The algorithm used by Heitzig and Reinhold to generate all finite lattices has been improved with McKay's isomorph-free exhaustive generation.
3. A second algorithm which generates only modular lattices has been developed. However, it does not necessarily generate all modular lattices, so is only used for verification.

Conclusion

1. The number of unique modular lattices up to isomorphism has been counted up to size $n = 19$ (they had previously been counted up to $n = 12$).
2. The algorithm used by Heitzig and Reinhold to generate all finite lattices has been improved with McKay's isomorph-free exhaustive generation.
3. A second algorithm which generates only modular lattices has been developed. However, it does not necessarily generate all modular lattices, so is only used for verification.
4. This algorithm can be modified to work for other types of lattice structures, such as:

Conclusion

1. The number of unique modular lattices up to isomorphism has been counted up to size $n = 19$ (they had previously been counted up to $n = 12$).
2. The algorithm used by Heitzig and Reinhold to generate all finite lattices has been improved with McKay's isomorph-free exhaustive generation.
3. A second algorithm which generates only modular lattices has been developed. However, it does not necessarily generate all modular lattices, so is only used for verification.
4. This algorithm can be modified to work for other types of lattice structures, such as:
   - Semimodular lattices.

Conclusion

1. The number of unique modular lattices up to isomorphism has been counted up to size $n = 19$ (they had previously been counted up to $n = 12$).
2. The algorithm used by Heitzig and Reinhold to generate all finite lattices has been improved with McKay's isomorph-free exhaustive generation.
3. A second algorithm which generates only modular lattices has been developed. However, it does not necessarily generate all modular lattices, so is only used for verification.
4. This algorithm can be modified to work for other types of lattice structures, such as:
   - Semimodular lattices.
   - Semidistributive lattices.

Conclusion

1. The number of unique modular lattices up to isomorphism has been counted up to size $n = 19$ (they had previously been counted up to $n = 12$).
2. The algorithm used by Heitzig and Reinhold to generate all finite lattices has been improved with McKay's isomorph-free exhaustive generation.
3. A second algorithm which generates only modular lattices has been developed. However, it does not necessarily generate all modular lattices, so is only used for verification.
4. This algorithm can be modified to work for other types of lattice structures, such as:
   - Semimodular lattices.
   - Semidistributive lattices.
   - Almost distributive lattices.

References

1. R. Belohlavek and V. Vychodil, *Residuated Lattices of Size* $\leq$ *12*, Order 27 (2010), 147–161.
2. G. Birkhoff, *On the structure of abstract algebras*, Proc. Camb. Phil. Soc. 31 (1935), 433–454.
3. R. Dedekind, *Über die von drei Moduln erzeugte Dualgruppe*, Math. Ann. 53 (1900), 371–403.
4. M. Erné, J. Heitzig and J. Reinhold, *On the number of distributive lattices*, Electron. J. Combin. 9 (2002), 23 pp.
5. J. Heitzig and J. Reinhold, *Counting Finite Lattices*, Algebra Univers. 48 (2002) 43–53.
6. B. D. McKay, *Isomorph-free exhaustive generation*, J. Algorithms 26 (1998) 306–324.