

Constructing unlabelled lattices

Volker Gebhardt, Stephen Tawn

28th September 2016

Abstract

We present an improved orderly algorithm for constructing all unlabelled lattices up to a given size, that is, an algorithm that constructs the minimal element of each isomorphism class relative to some total order.

Our algorithm employs a stabiliser chain approach for cutting branches of the search space that cannot contain a minimal lattice; to make this work, we grow lattices by adding a new layer at a time, as opposed to adding one new element at a time, and we use a total order that is compatible with this modified strategy.

The gain in speed is about two orders of magnitude. As an application, we compute the number of unlabelled lattices on 20 elements.

1 Introduction

Enumerating all isomorphism classes of unlabelled lattices, in the sense of systematically constructing a complete list of isomorphism classes up to a certain size threshold, is a difficult combinatorial problem. The number u_n of isomorphism classes of unlabelled lattices on n elements grows exponentially in n [KL71, KW80], as does the typical size of an isomorphism class. Indeed, the largest value of n for which u_n has been published previously is $n = 19$ [JL15, Slo].

When trying to enumerate combinatorial objects modulo isomorphism, one typically faces the problem that the number and the size of the isomorphism classes are so large that trying to weed out isomorphic objects through explicit isomorphism tests is out of the question. Instead, an *orderly algorithm* is needed, that is, an algorithm that traverses the search space in such a way that every isomorphism class is encountered exactly once.

A general strategy for the construction of isomorphism classes of combinatorial objects using *canonical construction paths* was described in [McK98]. Orderly algorithms for enumerating isomorphism classes of unlabelled lattices, as well as special subclasses of unlabelled lattices, were given in [HR02, JL15]. The fastest published method currently is the one described in [JL15]; the computations of u_{18} and u_{19} reported in [JL15] took 26 hours respectively 19 days on 64 CPUs.

Both authors acknowledge support under UWS grant 20721.81112. Volker Gebhardt acknowledges support under the Spanish Project MTM2013-44233.

MSC-class: 05A15 (Primary) 06A07, 05-04 (Secondary)

Keywords: unlabelled lattice, enumeration, orderly algorithm

The algorithms described in [HR02, JL15] follow a similar strategy: (i) A total order $<_{\text{wt}}$ on all labelled lattices of a given size is defined. (ii) Starting from the (unique) $<_{\text{wt}}$ -minimal lattice on 2 elements, the $<_{\text{wt}}$ -minimal labelled representative of each isomorphism class of unlabelled lattices with at most n elements is constructed using a depth first search, where the children of a parent lattice are obtained by adding a single new element covering the minimal element of the parent lattice. (iii) For each parent lattice, one child is obtained for every choice for the covering set of the added element that yields a labelled lattice that is $<_{\text{wt}}$ -minimal in its isomorphism class of unlabelled lattices.

It is the test for $<_{\text{wt}}$ -minimality in step (iii) that takes most of the time: While there are some necessary conditions that are easy to verify, ensuring that the newly constructed labelled lattice is indeed $<_{\text{wt}}$ -minimal in its isomorphism class requires checking the candidate covering set of the added element against all possible relabellings of the elements of the existing lattice; details are given in Section 2. Basically, one has a certain permutation group that acts on a configuration space of covering sets, and one must verify that a given candidate is minimal in its orbit.

It turns out that the elements of a $<_{\text{wt}}$ -minimal labelled lattice are arranged by *levels* (cf. Section 2), and thus it is tempting to construct and test candidate covering sets of a new element level by level, exploiting the levellised structure for a divide-and-conquer approach; such an approach promises two advantages: (i) The orbit of the restriction of a candidate covering set to a given level is potentially much smaller than the orbit of the complete covering set. (ii) The entire branch of the search space that corresponds to the candidate configuration of covers on the given level can potentially be discarded in a single test.

However, we shall see that the constructions from [HR02, JL15] do not adapt well to this levellised approach: In order to make the levellised approach work, we need to modify the depth-first-search to add one *level* at a time as opposed to one *element* at a time, and we need to modify the total order to be level-major.

The structure of the paper is as follows: In Section 2, we recall some results from [HR02, JL15] that are needed later, and we interpret the total order used in [HR02, JL15] as row-major. In Section 3, we describe our new construction using a level-major order and prove the results required to establish its correctness. In Section 4, we remark on implementation details and compare the performance of our new approach to that of those published in [HR02, JL15].

We thank the Institute for Mathematics at the University of Seville (IMUS) for providing access to a 64-node 512 GB RAM computer.

2 Background

We start by giving a brief summary of the algorithms from [HR02, JL15]. We refer to these references for details.

Definition 1. A finite poset L is an n -poset, if the elements of L are labelled $0, 1, \dots, n-1$, where 0 is a lower bound of L and $n-1$ is an upper bound of L . An n -poset that is a lattice is called an n -lattice. To avoid confusion with the numerical order of integers, we denote the partial order of an n -poset L by \sqsubseteq_L and \sqsupseteq_L , or simply \sqsubseteq and \sqsupseteq if the poset is obvious.

Notation 2. Assume that L is an n -poset.

For $a \in L$, we define the *shadow* of a as $\downarrow a = \downarrow_L a = \{x \in L : x \sqsubseteq_L a\}$ and the *shade* of a as $\uparrow a = \uparrow_L a = \{x \in L : x \sqsupseteq_L a\}$. For $A \subseteq L$, we define $\uparrow A = \uparrow_L A = \bigcup_{a \in A} \uparrow_L a$ as well as $\downarrow A = \downarrow_L A = \bigcup_{a \in A} \downarrow_L a$.

We say that $a \in L$ has *depth* $\text{dep}(a) = \text{dep}_L(a) = p$, if $p + 1$ is the length of a maximal chain from 1 to a in L . Given a non-negative integer k , we call $\text{lev}_k(L) = \{a \in L : \text{dep}_L(a) = k\}$ the k -th level of L . We say that L is *levellised*, if $\text{dep}_L(i) \leq \text{dep}_L(j)$ holds for all $0 < i \leq j < n$.

For $a, b \in L$, we write $a \prec_L b$ (or simply $a \prec b$) if a is *covered* by b in L , that is, $a \sqsubseteq_L b$ holds and $a \sqsubseteq_L x \sqsubseteq_L b$ implies $x = a$ or $x = b$. We denote the *covering set* of $a \in L$ by $\bigwedge a = \bigwedge_L a = \{x \in L : a \prec x\}$, and we say that $a \in L$ is an *atom* in L if $0 \prec_L a$ holds.

If L is a lattice and $a, b \in L$, we denote the least common upper bound of a and b in L by $a \vee_L b$ (or simply $a \vee b$), and the greatest common lower bound of a and b in L by $a \wedge_L b$ (or simply $a \wedge b$).

2.1 Canonical representatives

The idea of an orderly algorithm is to construct all those lattices that are minimal, with respect to a suitable total order, in their isomorphism class. In this section, we recall the total order used in [HR02, JL15] and some of its properties.

Definition 3. Let L be an n -poset.

- (a) For $A \subseteq L$, we define $\text{wt}_L(A) = \sum_{j \in A} 2^j$.
- (b) For $i \in L$, we define $\text{wt}_L(i) = \text{wt}_L(\bigwedge_L i)$.
- (c) We define $\text{wt}(L) = (\text{wt}_L(2), \text{wt}_L(3), \dots, \text{wt}_L(n-1))$.

Ordering n -lattices lexicographically with respect to $\text{wt}(L)$, we obtain a total order $<_{\text{wt}}$ on the set of all n -lattices.

Remark 4. An n -poset L is completely defined by its covering relation.

Indeed, the upper bound 1 is not covered by any element, and it covers precisely those elements that are not covered by any other element. Similarly, the lower bound 0 covers no element, and it is covered precisely by those elements that do not cover any other element. Thus, L is completely described by specifying the pairs (i, j) , for $1 < i, j < n$, for which $i \prec j$ holds.

The latter information can be interpreted as an $(n-2)$ by $(n-2)$ matrix over \mathbb{F}_2 , and the total order $<_{\text{wt}}$ from Definition 3 amounts to a row-major lexicographic order on the associated matrices; cf. Figure 1.

Theorem 5 ([HR02, Theorem 1]). *If L is a $<_{\text{wt}}$ -minimal n -lattice and one has $0 < i \leq j < n$, then $\text{dep}_L(i) \leq \text{dep}_L(j)$ holds.*

Corollary 6. *If L is a $<_{\text{wt}}$ -minimal n -lattice and one has $0 < i$ and $i \prec j$, then $j < i$ holds.*

Remark 7. Theorem 5 and Corollary 6 say that a $<_{\text{wt}}$ -minimal n -lattice L is levellised, that is, that the non-minimal levels of L are filled by elements labelled in their numerical order; cf. Figure 2.

Remark 8. Any relabelling $\pi \in \text{Sym}(\{0, \dots, n-1\})$ of the elements of an n -lattice L acts on each level of L , as $\text{dep}_L(\pi(i)) = \text{dep}_L(i)$ holds for every $i \in L$. In particular, if L is a levelled lattice with the elements labelled as in Figure 2, then the relabelled lattice $\pi(L)$ is levelled if and only if

$$\begin{aligned} \pi \in & \text{Sym}(\{0\}) \times \text{Sym}(\{1\}) \times \text{Sym}(\{2, \dots, a_2 - 1\}) \times \\ & \times \text{Sym}(\{a_2, \dots, a_3 - 1\}) \times \dots \times \text{Sym}(\{a_k, \dots, n-1\}) \end{aligned}$$

holds.

Theorem 9 ([HR02, Theorem 2]). *If L is a $<_{\text{wt}}$ -minimal n -lattice, one has*

$$\text{wt}_L(2) \leq \text{wt}_L(3) \leq \dots \leq \text{wt}_L(n-1).$$

Remark 10. Theorem 9 says that the rows of the matrices describing the covering relation of a $<_{\text{wt}}$ -minimal n -lattice L (cf. Figure 1) are sorted in non-decreasing order with respect to a (right-to-left) lexicographic order on the rows.

2.2 Incremental construction

The algorithms from [HR02, JL15] work by traversing a tree of $<_{\text{wt}}$ -minimal n -lattices in a depth-first manner; the root of the tree is the unique 2-lattice, and an $(n+1)$ -lattice \tilde{L} is a descendant of the $<_{\text{wt}}$ -minimal n -lattice L , if \tilde{L} is obtained from L by adding a new cover of 0 (labelled n) and \tilde{L} is $<_{\text{wt}}$ -minimal.

The lattice \tilde{L} is determined by L and the covering set of the new element n ; the possible choices for the latter can be characterised effectively.

Definition 11 ([HR02]). If L is an n -lattice, a non-empty antichain $A \subseteq L \setminus \{0\}$ is called a *lattice-antichain* for L , if $a \wedge_L b \in \{0\} \cup (\uparrow_L A)$ holds for any $a, b \in \uparrow_L A$.

Remark 12. To test the condition in Definition 11, it is clearly sufficient to verify that $a \wedge_L b \in \uparrow_L A$ holds for those pairs (a, b) that are minimal in the set

$$\{(a, b) \in (\uparrow_L A) \times (\uparrow_L A) : a \wedge_L b \neq 0\}$$

with respect to the product partial order in $L \times L$.

Theorem 13 ([HR02, Lemma 2]). *Let L be an n -lattice. A subset $A \subseteq L \setminus \{0\}$ is a lattice-antichain for L , if and only if L is a subposet of an $(n+1)$ -lattice L_A in which $0 \prec_{L_A} n$ (that is, n is an atom in L_A) and $\bigwedge_{L_A} n = A$ hold.*

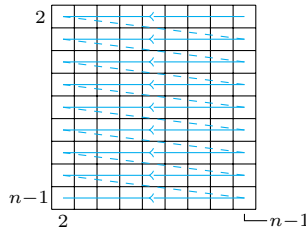


Figure 1: Interpreting the order $<_{\text{wt}}$ as row-major lexicographic order on the matrices specifying the covering relation. Note that adding a column for 1 would not affect the order: An entry in this column is determined by the other entries in the same row, and it is checked after those in the lexicographic comparison.

Remark 14. In the situation of [Theorem 13](#), it is clear that the pair (L, A) uniquely determines L_A and vice versa.

Moreover, the covering relation of L_A is obtained from the covering relation of L by

- (i) adding the pair $(0, n)$;
- (ii) adding all pairs of the form (n, a) for $a \in A$; and
- (iii) removing all pairs of the form $(0, a)$ for $a \in A$ that are present.

As mentioned in [Remark 4](#), the covers of 0 need not be stored explicitly; in this case, only step (ii) is needed.

Indeed, this definition of L_A makes sense for any n -poset L and any $A \subseteq L$. It is obvious from the definitions that one has $\text{wt}_{L_A}(n) = \text{wt}_L(A)$. Moreover, for $1 \leq i < n$, we have $i \not\leq_{L_A} n$ and thus $\text{wt}_{L_A}(i) = \text{wt}_L(i)$ and $\text{dep}_{L_A}(i) = \text{dep}_L(i)$.

The following two results are consequences of [Remark 14](#) and [Theorem 9](#).

Corollary 15 ([\[HR02, §3\]](#)). *If L is an n -lattice and A is a lattice-antichain for L such that L_A is $<_{\text{wt}}$ -minimal, then L is $<_{\text{wt}}$ -minimal.*

Corollary 16 ([\[HR02, §5\]](#)). *If L is an n -lattice and A is a lattice-antichain for L such that L_A is $<_{\text{wt}}$ -minimal, then one has $A \cap (\text{lev}_{k-1}(L) \cup \text{lev}_k(L)) \neq \emptyset$ for $k = \text{dep}_L(n-1)$.*

2.3 Testing for canonicity

In the light of [Corollary 16](#), there are two cases to consider for testing whether the descendant L_A of a $<_{\text{wt}}$ -minimal n -lattice L defined by a lattice-antichain A for L with $\text{dep}_L(n-1) = k$ is $<_{\text{wt}}$ -minimal:

- (A) $A \cap \text{lev}_k(L) \neq \emptyset$, that is, $\text{dep}_{L_A}(n) = k + 1$
- (B) $A \cap \text{lev}_k(L) = \emptyset \neq A \cap \text{lev}_{k-1}(L)$, that is, $\text{dep}_{L_A}(n) = k$

Case (A):

In this case, the new element n forms a separate level of L_A ; cf. [Figure 3\(a\)](#). As L_A is levellised by construction and the non-minimal elements of L correspond to the levels $0, \dots, k$ of L_A , any relabelling π of L_A for which $\pi(L_A)$ is levellised must fix n and induce a relabelling of L by [Remark 8](#). By [Remark 14](#) and the definition of the lexicographic order, $\text{wt}(\pi(L_A)) < \text{wt}(L_A)$ implies $\text{wt}(\pi(L)) \leq \text{wt}(L)$. By [Corollary 15](#), the latter implies $\text{wt}(\pi(L)) = \text{wt}(L)$ and thus $\pi(L) = L$.

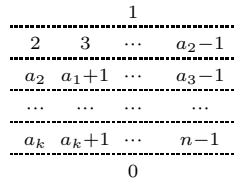


Figure 2: A levellised n -lattice; the dashed lines separate the levels.

To test whether L_A is $<_{\text{wt}}$ -minimal, it is thus sufficient to check that

$$\text{wt}(A) = \min \{ \text{wt}(\pi(A)) : \pi \in \text{Stab}(L) \times \text{Sym}(\{n\}) \cong \text{Stab}(L) \} \quad (1)$$

holds, again using [Remark 14](#).

Case (B):

In this case, the new element n is added to the lowest existing non-trivial level of L ; cf. [Figure 3\(b\)](#). Thus, a relabelling π of L_A for which $\pi(L_A)$ is levelled *need not* fix n and induce a relabelling of L . It will, however, induce a relabelling of the lattice L' induced by the levels $0, \dots, k-1, k+1$ of L (or L_A), and one has $\pi(L') = L'$ by the same arguments as in the previous case.

If the lowest non-trivial level of L_A contains the elements a_k, \dots, n , checking whether L_A is $<_{\text{wt}}$ -minimal means testing that $(\text{wt}_{L_A}(a_k), \dots, \text{wt}_{L_A}(n))$ is lexicographically minimal in its orbit under the group $\text{Stab}(L') \times \text{Sym}(\{a_k, \dots, n\})$.

Observe that one has $\bigwedge_{L_A} i \subseteq L'$ for $a_k \leq i \leq n$, and that the action of $\pi \in \text{Stab}(L') \times \text{Sym}(\{a_k, \dots, n\})$ not only modifies the individual weights $\text{wt}_{L_A}(i)$ (by relabelling the elements of L'), but also permutes their positions in the sequence (by acting on $\{a_k, \dots, n\}$).

2.4 Vertically indecomposable lattices

Definition 17. An n -lattice L is *vertically decomposable*, if there exists an element $i \in L \setminus \{0, 1\}$ that is comparable to every other element of L . Otherwise, L is *vertically indecomposable*.

One can speed up the construction by restricting to lattices that are vertically indecomposable; a straightforward recursion makes it possible to recover all lattices from the vertically indecomposable ones. We refer to [[HR02](#), §5] for details.

3 An improved algorithm

The test for minimality of $\text{wt}(A)$ respectively $(\text{wt}_{L_A}(a_k), \dots, \text{wt}_{L_A}(n))$ in their orbit under the acting permutation group is the most time consuming part of the construction and thus an obvious target for improvement.

In [Section 3.1](#), we sketch the basic idea for a more efficient algorithm, but we will see that the construction of [[HR02](#), [JL15](#)] has to be modified to make this idea work. We describe our modified construction in [Section 3.2](#).

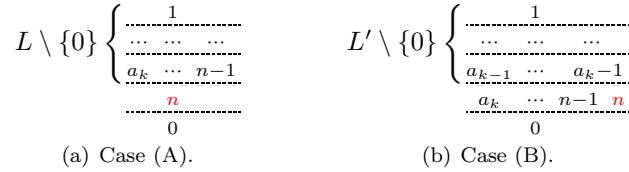


Figure 3: The lattice L_A obtained by adding a new cover n of 0 to the n -lattice L .

3.1 Stabiliser chain approach

Case (A) from [Section 2.3](#) suggests a possible approach, namely the use of a standard technique from computational group theory: stabiliser chains.

Since $\text{Stab}(L)$ preserves each level of L by [Remark 8](#) and L is levelled, it is tempting to construct and test lattice-antichains level by level: Defining $S_k := \text{Stab}(L)$ as well as $A_d := A \cap \text{lev}_d(L)$ and $S_{d-1} := S_d \cap \text{Stab}(A_d)$ for $d = k, \dots, 1$, condition (1) is equivalent to the following condition:

$$\text{wt}(A_d) = \min \{ \text{wt}(\pi(A_d)) : \pi \in S_d \} \quad \text{for } d = k, \dots, 1 \quad (2)$$

The sets A_d for $d = k, \dots, 1$ can be constructed and tested one at a time; this offers two advantages: Firstly, if the test at level d fails, the levels $d - 1, \dots, 1$ don't have to be constructed; an entire branch of the search space is discarded in one step. Secondly, even if the test succeeds on all levels, the cost of testing condition (2) is proportional to $\sum_{d=1}^k |A_d^{S_d}|$, and thus in general much smaller than $|A^{\text{Stab}(L)}| = \prod_{d=1}^k |A_d^{S_d}|$, which is the cost of testing condition (1) directly.

However, when trying to use a similar approach for Case (B), we run into problems: We must compare

$$(\text{wt}_{L_A}(a_k), \dots, \text{wt}_{L_A}(n)) = (\text{wt}_L(a_k), \dots, \text{wt}_L(n-1), \text{wt}_L(A))$$

lexicographically to its images under the elements of the acting permutation group $\text{Stab}(L') \times \text{Sym}(\{a_k, \dots, n\})$, but if A has only been constructed partially, $\text{wt}_{L_A}(n) = \text{wt}_L(A)$ is not completely determined; in the interpretation of [Remark 4](#) and [Figure 1](#), the leftmost entries of the last row of the binary matrix corresponding to L_A are undefined.

The elements of the group $\text{Stab}(L') \times \text{Sym}(\{a_k, \dots, n\})$ can permute the rows of this matrix, so the position of the undefined entries will vary. Clearly, the lexicographic comparison of the two matrices must stop once it reaches an entry that is undefined in one of the matrices being compared; in this situation, the order of the two matrices cannot be decided on the current level.

The problem is that the position in the matrix at which the lexicographic comparison must stop depends on the relabelling that is applied (cf. [Figure 4](#)). A consequence of this is that the subset of elements of $\text{Stab}(L') \times \text{Sym}(\{a_k, \dots, n\})$ for which the parts of the matrices that can be compared are equal does not form a subgroup, so applying a stabiliser chain approach is not possible.

3.2 Levelled construction

The analysis at the end of the preceding section indicates that the problem is that possible relabellings can swap an element whose covering set is only partially determined with an element whose covering set is completely determined, or in other words, that we add a new element to an existing level of L .

The idea for solving this problem is simple: Rather than adding one element at a time, possibly to an already existing level, we only ever add an entire level at a time; that way, the problem of adding elements to an existing level is avoided.

To make the stabiliser chain approach work in this setting, we must use a total order that compares parts of the covering sets in the same order in which they are constructed; that is, we have to compare the entries of the matrices describing the covering relations in level-major order.

Notation 18. For a levelled n -lattice L with $n > 2$ and $\text{dep}_L(n-1) = k$, let L' denote the lattice induced by the levels $0, \dots, k-1, k+1$ of L , that is, the lattice obtained from L by removing its last non-trivial level.

Note that L' is a levelled n' -lattice for some $n' < n$.

The total order we are about to define uses the partition of the covering set of each element according to levels.

Definition 19. Given a levelled n -lattice L with $\text{dep}_L(n-1) = k$, an element $i \in L \setminus \{0\}$ and an integer $d \in \{1, \dots, k-1\}$, we define $\bigwedge_L^d i = (\bigwedge_L i) \cap \text{lev}_d(L)$.

Definition 20. Using induction on n , we define a relation $<$ on the set of levelled n -lattices that are isomorphic as unlabelled lattices as follows:

Given $n > 2$ and levelled n -lattices L_1 and L_2 that are isomorphic as unlabelled lattices, we say $L_1 < L_2$, if one of the following holds:

- $L'_1 < L'_2$
- $L'_1 = L'_2 = L'$ and, denoting $\text{dep}_{L_1}(n-1) = \text{dep}_{L_2}(n-1) = k$ and $\text{lev}_k(L_1) = \text{lev}_k(L_2) = \{a_k, \dots, n-1\}$, there exist $\ell \in \{1, \dots, k-1\}$ as well as $i \in \{a_k, \dots, n-1\}$ such that both of the following hold:
 - $\text{wt}_{L'}(\bigwedge_{L_1}^d j) = \text{wt}_{L'}(\bigwedge_{L_2}^d j)$ if $d > \ell$, or $d = \ell$ and $j \in \{a_k, \dots, i-1\}$
 - $\text{wt}_{L'}(\bigwedge_{L_1}^\ell i) < \text{wt}_{L'}(\bigwedge_{L_2}^\ell i)$

Remark 21. The relation from [Definition 20](#) corresponds to a level-major lexicographic comparison of the binary matrices describing the covering relations of L_1 and L_2 as illustrated in [Figure 5](#).

As L_1 and L_2 are levelled, both matrices are lower block triangular, with the blocks defined by the levels. Moreover, as L_1 and L_2 are isomorphic as unlabelled lattices, the block structures of both matrices are identical.

Notice also that the matrix describing the covering relation of L'_1 (respectively L'_2) is obtained from that of L_1 (respectively L_2) by removing the lowest row and the rightmost column of blocks.

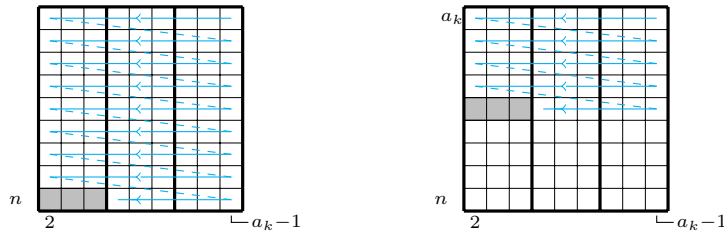


Figure 4: Lexicographic comparison of a lattice L_A obtained from a partially constructed lattice-antichain A (left) and a relabelling (right); only the relevant parts of the matrices are shown. Thick lines indicate the boundaries between levels. The parts of the lattice-antichain not yet constructed are shown in grey.

Lemma 22. *The relation $<$ from Definition 20 is a total order on the set of levellised n -lattices that are isomorphic as unlabelled lattices.*

Proof. Verifying transitivity is routine. Trichotomy holds, as for $s \in \{1, 2\}$ and $i \in \{2, \dots, n-1\}$ with $\text{dep}_{L_1}(i) = \text{dep}_{L_2}(i) = D$, one has

$$\bigwedge_{L_s} i = \begin{cases} \bigcup_{d=1}^{D-1} \bigwedge_{L_s}^d i & ; \text{ if } \bigcup_{d=1}^{D-1} \bigwedge_{L_s}^d i \neq \emptyset \\ \{1\} & ; \text{ if } \bigcup_{d=1}^{D-1} \bigwedge_{L_s}^d i = \emptyset \end{cases}$$

whence $\bigwedge_{L_1} i = \bigwedge_{L_2} i$ holds if and only if one has $\text{wt}_{L'}(\bigwedge_{L_1}^d i) = \text{wt}_{L'}(\bigwedge_{L_2}^d i)$ for $d = 1, \dots, D-1$. \square

Definition 23. An n -lattice L is *canonical* if L is levellised and $<$ -minimal among all levellised n -lattices that are isomorphic to L as unlabelled lattices.

Lemma 24. *Every isomorphism class of unlabelled lattices on n elements contains a unique canonical n -lattice.*

Proof. The set of representatives that are levellised n -lattices is clearly non-empty and finite, and $<$ is a total order on this set by Lemma 22. \square

The following Theorem 26 is an analogue of Corollary 15.

Lemma 25. *If L is a levellised $(n+1)$ -lattice for $n > 1$ and $0 \prec_L n$, then $L \setminus \{n\}$ is a levellised n -lattice and $\text{dep}_{L \setminus \{n\}}(i) = \text{dep}_L(i)$ holds for $i = 1, \dots, n-1$.*

Proof. Let $\bar{L} = L \setminus \{n\}$. As L is levellised, $1 \leq i < n$ implies $\text{dep}_L(i) \leq \text{dep}_L(n)$, so $i \not\prec_L n$, and hence $\text{dep}_{\bar{L}}(i) = \text{dep}_L(i)$. As L is levellised, so is \bar{L} .

A routine verification shows that for $a, b \in \bar{L}$ one has

$$a \vee_{\bar{L}} b = a \vee_L b \quad \text{and} \quad a \wedge_{\bar{L}} b = \begin{cases} a \wedge_L b & ; \text{ if } a \wedge_L b \neq n \\ 0 & ; \text{ if } a \wedge_L b = n \end{cases}$$

so \bar{L} is an n -lattice. \square

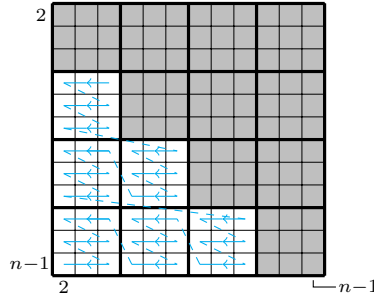


Figure 5: Interpreting the order $<$ as level-major lexicographic order on the matrices specifying the covering relation. Thick lines indicate the boundaries between levels. The entries of the matrix shown in grey are zero.

Theorem 26. *If L is a canonical n -lattice, then L' is canonical.*

Proof. Iterated application of [Lemma 25](#) shows that L' is levelled.

Assume that π' is a relabelling of L' such that $\pi'(L')$ is levelled and one has $\pi'(L') < L'$. We can trivially extend π' to a relabelling π of L such that $(\pi(L))' = \pi'(L') < L'$, contradicting the assumption that L is canonical. \square

[Theorem 26](#) means that we can again construct a tree of canonical n -lattices in a depth-first manner; the root of the tree is the unique 2-lattice, and an $(n+m)$ -lattice \tilde{L} is a descendant of the canonical n -lattice L , if \tilde{L} is obtained from L by adding m new covers of 0 (labelled $n, \dots, n+m-1$) and \tilde{L} is canonical.

The lattice \tilde{L} is determined by L and the covering sets of the new elements $n, \dots, n+m-1$; the possible choices for the latter can again be characterised effectively using lattice-antichains, although this time, extra compatibility conditions are needed. The following [Theorem 29](#), a generalisation of [Theorem 13](#), makes this precise.

Notation 27. Given $m \in \mathbb{N}^+$, an n -poset L , and $A_n, \dots, A_{n+m-1} \subseteq L \setminus \{0\}$, let $L_{A_n, \dots, A_{n+m-1}} = (\dots (L_{A_n}) \dots)_{A_{n+m-1}}$ denote the $(n+m)$ -poset obtained from L by adding m new atoms $n, \dots, n+m-1$ with $\bigwedge_{L_{A_n, \dots, A_{n+m-1}}} i = A_i$ for $i = n, \dots, n+m-1$. (See [Remark 14](#).)

Lemma 28. *Let L be a levelled n -poset with $\text{dep}_L(n-1) = k$, let $m \in \mathbb{N}^+$, and let $A_i \subseteq L \setminus \{0\}$ for $i = n, \dots, n+m-1$. The following are equivalent:*

- (A) $\tilde{L} = L_{A_n, \dots, A_{n+m-1}}$ is a levelled $(n+m)$ -poset with $\text{dep}_{\tilde{L}}(a) = \text{dep}_L(a) \leq k$ for $1 \leq a < n$ and $\text{dep}_{\tilde{L}}(n) = \dots = \text{dep}_{\tilde{L}}(n+m-1) = k+1$.
- (B) $A_i \cap \text{lev}_k(L) \neq \emptyset$ holds for $n \leq i < n+m$.

Proof. As L is levelled, one has $\text{dep}_L(a) \leq \text{dep}_L(n-1) = k$ for all $a \in L \setminus \{0\}$, and induction using [Remark 14](#) shows that $\text{dep}_{\tilde{L}}(a) = \text{dep}_L(a) \leq k$ holds for all $a \in L \setminus \{0\}$. Thus, for any $i = n, \dots, n+m-1$, one has $\text{dep}_{\tilde{L}}(i) = k+1$ if and only if $\emptyset \neq (\bigwedge_{\tilde{L}} i) \cap \text{lev}_k(\tilde{L}) = A_i \cap \text{lev}_k(\tilde{L}) = A_i \cap \text{lev}_k(L)$ holds. Finally, $\text{dep}_{\tilde{L}}(n) = \dots = \text{dep}_{\tilde{L}}(n+m-1) = k+1$ implies that \tilde{L} is levelled. \square

Theorem 29. *Let L be a levelled n -lattice with $\text{dep}_L(n-1) = k$, let $m \in \mathbb{N}^+$, and let $A_i \subseteq L \setminus \{0\}$ for $i = n, \dots, n+m-1$. The following are equivalent:*

- (A) $\tilde{L} = L_{A_n, \dots, A_{n+m-1}}$ is a levelled $(n+m)$ -lattice, $\text{dep}_{\tilde{L}}(a) = \text{dep}_L(a) \leq k$ for $1 \leq a < n$, and $\text{dep}_{\tilde{L}}(n) = \dots = \text{dep}_{\tilde{L}}(n+m-1) = k+1$.
- (B) (i) $A_i \cap \text{lev}_k(L) \neq \emptyset$ holds for $n \leq i < n+m$;
(ii) A_i is a lattice-antichain for L for $n \leq i < n+m$; and
(iii) if $a, b \in (\uparrow_L A_i) \cap (\uparrow_L A_j)$ for $n \leq i < j < n+m$, then $a \wedge_L b \neq 0$.

Proof. We use induction on m . In the case $m = 1$, condition (B)(iii) is vacuous. By [Theorem 13](#), the poset $\tilde{L} = L_{A_n}$ is a lattice if and only if A_n is a lattice-antichain for L . Together with [Lemma 28](#), the claim is shown in this case.

Let $m > 1$ and consider $L^\circ = L_{A_n, \dots, A_{n+m-2}}$ and $A = A_{n+m-1}$; we have $\tilde{L} = (L^\circ)_A$. By [Lemma 28](#), we can assume $\text{dep}_{\tilde{L}}(a) = \text{dep}_{L^\circ}(a) = \text{dep}_L(a) \leq k$

for $1 \leq a < n$ and $\text{dep}_{\tilde{L}}(n) = \dots = \text{dep}_{\tilde{L}}(n+m-1) = \text{dep}_{L^\circ}(n) = \dots = \text{dep}_{L^\circ}(n+m-2) = k+1$. In particular, $\uparrow_{\tilde{L}} A = \uparrow_{L^\circ} A = \uparrow_L A$ holds. Also, for $i = n, \dots, n+m-1$ and $a \in L$, we have $i \sqsubseteq_{\tilde{L}} a$ if and only if $a \in \uparrow_L A_i$ holds.

First assume that (A) holds.

Induction using [Lemma 25](#) shows that the sets A_n, \dots, A_{n+m-2} are lattice-antichains for L , and $a, b \in (\uparrow_L A_i) \cap (\uparrow_L A_j)$ for $n \leq i < j < n+m-1$ implies $a \wedge_L b \neq 0$. Further, by [Theorem 13](#), the set A is a lattice-antichain for L° , which means that for $a, b \in \uparrow_{L^\circ} A = \uparrow_L A$, one has $a \wedge_{L^\circ} b \in \{0\} \cup \uparrow_{L^\circ} A = \{0\} \cup \uparrow_L A$.

Let $a, b \in \uparrow_L A$. If we had $a, b \in \uparrow_L A_i$ for some $n \leq i < n+m-1$, we would have $i \sqsubseteq_{\tilde{L}} a \wedge_{\tilde{L}} b$ and $(n+m-1) \sqsubseteq_{\tilde{L}} a \wedge_{\tilde{L}} b$. As i and $n+m-1$ are distinct atoms of \tilde{L} , this contradicts the hypothesis that \tilde{L} is a lattice. Thus, we have condition (B)(iii). Moreover, $a \wedge_{L^\circ} b = a \wedge_L b$ for all $a, b \in \uparrow_L A$, and thus A is a lattice-antichain for L . Together with [Lemma 28](#), we have thus shown (B).

Now assume that (B) holds.

By induction, L° is a levelled $(n+m-1)$ -lattice. Let $a, b \in \uparrow_{L^\circ} A = \uparrow_L A$. If we have $a \wedge_L b \neq 0$, then $a \wedge_{L^\circ} b = a \wedge_L b \in \uparrow_L A = \uparrow_{L^\circ} A$ holds, as A is a lattice-antichain for L by assumption. On the other hand, $a \wedge_L b = 0$ implies $a \wedge_{L^\circ} b = 0$, as by assumption, there is no $i \in \{n, \dots, n+m-2\}$ such that $a, b \in \uparrow_L A_i$ holds. Thus, A is a lattice-antichain for L° , whence \tilde{L} is a lattice by [Theorem 13](#). Together with [Lemma 28](#), we have thus shown (A). \square

Notation 30. Let L be a levelled n -lattice with $\text{dep}_L(n-1) = k$, let $m \in \mathbb{N}^+$, and let $A_i \subseteq L \setminus \{0\}$ for $i = n, \dots, n+m-1$. For $d = 1, \dots, k$ we define $\mathcal{LW}_L^d(A_n, \dots, A_{n+m-1})$ as the sequence

$$\left(\text{wt}_L(A_n \cap \text{lev}_d(L)), \dots, \text{wt}_L(A_{n+m-1} \cap \text{lev}_d(L)) \right)$$

and we define the sequence $\mathcal{LW}_L(A_n, \dots, A_{n+m-1})$ as the concatenation of $\mathcal{LW}_L^k(A_n, \dots, A_{n+m-1}), \dots, \mathcal{LW}_L^1(A_n, \dots, A_{n+m-1})$ in this order.

Theorem 31. *Let L be a levelled n -lattice with $\text{dep}_L(n-1) = k$, let $m \in \mathbb{N}^+$, and assume that $A_i \subseteq L \setminus \{0\}$ for $i = n, \dots, n+m-1$ satisfy condition (B) from [Theorem 29](#). Then $L_{A_n, \dots, A_{n+m-1}}$ is a canonical $(n+m)$ -lattice if and only if:*

- (i) L is canonical; and
- (ii) the sequence $\mathcal{LW}_L(A_n, \dots, A_{n+m-1})$ is lexicographically minimal under the action of $\text{Stab}(L) \times \text{Sym}(\{n, \dots, n+m-1\})$ given by

$$\pi \left(\mathcal{LW}_L(A_n, \dots, A_{n+m-1}) \right) = \mathcal{LW}_L \left(\pi(A_{\pi^{-1}(n)}), \dots, \pi(A_{\pi^{-1}(n+m-1)}) \right)$$

for $\pi \in \text{Stab}(L) \times \text{Sym}(\{n, \dots, n+m-1\})$.

Proof. As any $\pi \in \text{Stab}(L) \times \text{Sym}(\{n, \dots, n+m-1\})$ induces a relabelling of the elements of L , the action is well-defined. Moreover, defining $\tilde{L} = L_{A_n, \dots, A_{n+m-1}}$, [Theorem 29](#) implies $\downarrow_{\tilde{L}}^d i = A_i \cap \text{lev}_d(L)$ for $i = n, \dots, n+m-1$ and $1 \leq d \leq k$.

First assume that \tilde{L} is canonical. By [Theorem 26](#), $L = \tilde{L}'$ is canonical, so (i) holds. If (ii) does not hold, there exist $\pi \in \text{Stab}(L) \times \text{Sym}(\{n, \dots, n+m-1\})$ as well as $\ell \in \{1, \dots, k\}$ and $i \in \{n, \dots, n+m-1\}$ such that one has

- $\text{wt}_L(\pi(A_{\pi^{-1}(j)}) \cap \text{lev}_d(L)) = \text{wt}_L(A_j \cap \text{lev}_d(L))$ if $d > \ell$, or $d = \ell$ and $j \in \{n, \dots, i-1\}$; and
- $\text{wt}_L(\pi(A_{\pi^{-1}(i)}) \cap \text{lev}_\ell(L)) < \text{wt}_L(A_i \cap \text{lev}_\ell(L))$.

By [Theorem 29](#), $\widehat{L} = L_{\pi(A_{\pi^{-1}(n)}), \dots, \pi(A_{\pi^{-1}(n+m-1)})}$ is a levellised $(n+m)$ -lattice, and $\widehat{L}' = L = \widetilde{L}'$ holds by construction. Hence, the above conditions mean that \widetilde{L} is not canonical, contradicting the assumption. Thus (ii) holds.

Conversely, if \widetilde{L} is not canonical, there is a relabelling π of \widetilde{L} such that $\widehat{L} = \pi(\widetilde{L})$ is levellised and $\widehat{L} < \widetilde{L}$ holds. As π acts on the levels of \widetilde{L} it induces a relabelling of L , and we have $\widehat{L}' = (\pi(\widetilde{L}))' = \pi(\widetilde{L}') = \pi(L)$, which is levellised as well. If (i) holds, we cannot have $\pi(L) < L$, so $\widehat{L} < \widetilde{L}$ implies that one has $\widehat{L}' = \widetilde{L}' = L$ and there exist $\ell \in \{1, \dots, k\}$ as well as $i \in \{n, \dots, n+m-1\}$ such that both of the following hold:

- $\text{wt}_L(\lambda_{\widehat{L}}^d j) = \text{wt}_L(\lambda_{\widetilde{L}}^d j)$ if $d > \ell$, or $d = \ell$ and $j \in \{n, \dots, i-1\}$
- $\text{wt}_L(\lambda_{\widehat{L}}^\ell i) < \text{wt}_L(\lambda_{\widetilde{L}}^\ell i)$.

As we have $\lambda_{\widehat{L}}^d j = \pi(A_{\pi^{-1}(j)}) \cap \text{lev}_d(L)$ for $j = n, \dots, n+m-1$ and $1 \leq d \leq k$, the above conditions imply that

$$\mathcal{LW}_L(\pi(A_{\pi^{-1}(n)}), \dots, \pi(A_{\pi^{-1}(n+m-1)})) = \pi(\mathcal{LW}_L(A_n, \dots, A_{n+m-1}))$$

is lexicographically smaller than $\mathcal{LW}_L(A_n, \dots, A_{n+m-1})$. Since $\pi(L) = \widehat{L}' = L$, we have $\pi \in \text{Stab}(L) \times \text{Sym}(\{n, \dots, n+m-1\})$, so (ii) does not hold. \square

Corollary 32. *Let L be a levellised n -lattice with $\text{dep}_L(n-1) = k$, let $m \in \mathbb{N}^+$, and assume that $A_i \subseteq L \setminus \{0\}$ for $i = n, \dots, n+m-1$ satisfy condition (B) from [Theorem 29](#). Then $L_{A_n, \dots, A_{n+m-1}}$ is a canonical $(n+m)$ -lattice if and only if:*

- (i) L is canonical; and
- (ii) for $d = k, \dots, 1$, the sequence $\mathcal{LW}_L^d(A_n, \dots, A_{n+m-1})$ is lexicographically minimal under the action of S_d given by

$$\pi(\mathcal{LW}_L^d(A_n, \dots, A_{n+m-1})) = \mathcal{LW}_L^d(\pi(A_{\pi^{-1}(n)}), \dots, \pi(A_{\pi^{-1}(n+m-1)}))$$

for $\pi \in S_d$, where we define $S_k = \text{Stab}(L) \times \text{Sym}(\{n, \dots, n+m-1\})$ and $S_{d-1} = S_d \cap \text{Stab}(\mathcal{LW}_L^d(A_n, \dots, A_{n+m-1}))$ for $d = k, \dots, 2$.

Proof. Since the sequence $\mathcal{LW}_L(A_n, \dots, A_{n+m-1})$ is the concatenation of the sequences $\mathcal{LW}_L^k(A_n, \dots, A_{n+m-1}), \dots, \mathcal{LW}_L^1(A_n, \dots, A_{n+m-1})$ in this order, condition (ii) from [Corollary 32](#) is equivalent to condition (ii) of this corollary. \square

Remark 33. [Corollary 32](#) makes it possible to construct the lattice-antichains A_n, \dots, A_{n+m-1} level by level: The comparison at step d in condition (ii) only involves the elements of A_n, \dots, A_{n+m-1} that live on the level d of L . In particular, the benefits of using stabiliser chains mentioned in [Section 3.1](#) apply:

- (i) If the test at level d fails, the levels $d-1, \dots, 1$ do not have to be constructed; an entire branch of the search space is discarded in one step.
- (ii) The cost of testing condition (ii) of [Corollary 32](#) is in general much smaller than the cost of testing condition (ii) of [Theorem 31](#): The former is proportional to

$$\sum_{d=1}^k \left| (\mathcal{LW}_L^d(A_n, \dots, A_{n+m-1}))^{S_d} \right|,$$

while the latter is proportional to

$$\left| (\mathcal{LW}_L(A_n, \dots, A_{n+m-1}))^{S_k} \right| = \prod_{d=1}^k \left| (\mathcal{LW}_L^d(A_n, \dots, A_{n+m-1}))^{S_d} \right|.$$

[Figure 6](#) shows the comparisons that are made when testing one step of condition (ii) of [Corollary 32](#). Note that a reordering of the rows of the matrix does not change the position at which the lexicographic comparison stops; this property is necessary for the stabiliser chain approach to work.

4 Implementation and results

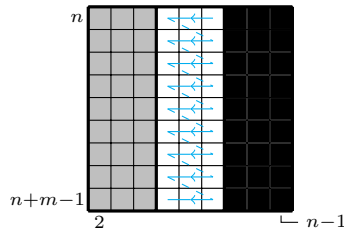
4.1 Implementation notes

This section sketches some ideas that are crucial for an efficient implementation of the algorithm presented in the preceding sections.

4.1.1 Representing antichains using up-closed sets

While the theoretical results of [Section 3](#) are formulated in terms of antichains, it is easier and computationally more efficient to work with sets S that are *up-closed*, meaning that $\uparrow S = S$ holds. (For instance, testing whether $A \subseteq L$ is a lattice-antichain for L only involves $\uparrow_L A$.)

Clearly, if A is an antichain, then $\uparrow_L A$ is up-closed. Moreover, the set of minimal elements of $\uparrow_L A$ is equal to A .



[Figure 6](#): Lexicographic comparison of a lattice $L_{A_n, \dots, A_{n+m-1}}$ obtained from a sequence of partially constructed lattice-antichains and a relabelling; only the relevant part of the matrix is shown. Thick lines indicate the boundaries between levels. Parts of the lattice-antichains not yet constructed are shown in grey. Parts of the lattice-antichains known to coincide are shown in black.

Lemma 34. *Let L be a levelled n -lattice with $\text{dep}_L(n-1) = k$, let A and B be antichains in L , and let $\ell \in \{1, \dots, k\}$. The following are equivalent:*

- (i) *One has $\text{wt}_L(A \cap \text{lev}_d(L)) = \text{wt}_L(B \cap \text{lev}_d(L))$ for $d = k, \dots, \ell-1$, and $\text{wt}_L(A \cap \text{lev}_\ell(L)) < \text{wt}_L(B \cap \text{lev}_\ell(L))$.*
- (ii) *One has $\text{wt}_L((\uparrow_L A) \cap \text{lev}_d(L)) = \text{wt}_L((\uparrow_L B) \cap \text{lev}_d(L))$ for $d = k, \dots, \ell-1$, and $\text{wt}_L((\uparrow_L A) \cap \text{lev}_\ell(L)) < \text{wt}_L((\uparrow_L B) \cap \text{lev}_\ell(L))$.*

Proof. If (i) holds, one has $((\uparrow_L A) \setminus A) \cap \text{lev}_d(L) = ((\uparrow_L B) \setminus B) \cap \text{lev}_d(L)$ for $d = k, \dots, \ell$, since L is levelled. In particular, one has

$$\begin{aligned} ((\uparrow_L B) \setminus (\uparrow_L A)) \cap \text{lev}_\ell(L) &= (B \setminus A) \cap \text{lev}_\ell(L) \text{ and} \\ ((\uparrow_L A) \setminus (\uparrow_L B)) \cap \text{lev}_\ell(L) &= (A \setminus B) \cap \text{lev}_\ell(L), \end{aligned}$$

which together with (i) imply (ii).

As A and B are the sets of minimal elements of $\uparrow_L A$ respectively $\uparrow_L B$, the converse implication is obvious. \square

Corollary 35. *Let L be a levelled n -lattice with $\text{dep}_L(n-1) = k$, let $m \in \mathbb{N}^+$, and assume that $A_i \subseteq L \setminus \{0\}$ for $i = n, \dots, n+m-1$ satisfy condition (B) from [Theorem 29](#). Then $L_{A_n, \dots, A_{n+m-1}}$ is a canonical $(n+m)$ -lattice if and only if:*

- (i) *L is canonical; and*
- (ii) *for $d = k, \dots, 1$, the sequence $\mathcal{LW}_L^d(\uparrow_L A_n, \dots, \uparrow_L A_{n+m-1})$ is lexicographically minimal under the action of S_d as in [Corollary 32](#).*

Proof. The claim follows from [Corollary 32](#) with [Lemma 34](#) and the observation that one has $\uparrow_L(\pi(A)) = \pi(\uparrow_L A)$ for any $A \subseteq L$ and $\pi \in S_d$. \square

4.1.2 Packed representation of antichains and Beneš networks

Let L be a canonical n -lattice with $\text{dep}_L(n-1) = k$, and let $m \in \mathbb{N}^+$. To generate the descendants of L with m elements on level $k+1$, we use a backtrack search to construct the sets $(\uparrow_L A_i) \cap \text{lev}_d(L)$ for $d = k, \dots, 1$ (outer loop) and $i = n, \dots, n+m-1$ (inner loop).

Every time a candidate set $(\uparrow_L A_i) \cap \text{lev}_d(L)$ has been chosen, we use condition (B) from [Theorem 29](#) to check for possible contradictions (backtracking if there are any), and to keep track of any elements whose presence in $(\uparrow_L A_i) \cap \text{lev}_{d'}(L)$ for some $d > d' \geq 1$ is forced by the choices made so far (restricting the possible choices later in the backtrack search if there are any).

Once all candidate sets on the current level have been chosen, we check for minimality under the action of the appropriate stabiliser S_d (cf. [Corollary 35](#)), backtracking if necessary.

Given the large number of configurations that have to be generated and tested for canonicity, it is critical to use an efficient data structure to store a configuration of antichains.

The sets $(\uparrow_L A_n) \cap \text{lev}_d(L), \dots, (\uparrow_L A_{n+m-1}) \cap \text{lev}_d(L)$ are encoded as a single $(m \cdot |\text{lev}_d(L)|)$ -bit integer. That way, a lexicographic comparison of two configurations reduces to a single comparison of two $(m \cdot |\text{lev}_d(L)|)$ -bit integers.

When constructing lattices with up to 18 elements, $m \cdot |\text{lev}_d(L)|$ is at most 64; when constructing lattices with up to 23 elements, $m \cdot |\text{lev}_d(L)|$ is at most 128. Thus, on a 64-bit CPU, a lexicographic comparison of two configurations costs only very few clock cycles.

To be able to apply permutations to a packed representation as described above effectively, we pre-compute a Beneš network [Knu09, § 7.1.3] for each generator of the stabiliser S_d . That way, the application of the generator to the sequence $((\uparrow_L A_n) \cap \text{lev}_d(L), \dots, (\uparrow_L A_{n+m-1}) \cap \text{lev}_d(L))$ is realised by a sequence of bit-wise operations (XOR and shift operations) on the $(m \cdot |\text{lev}_d(L)|)$ -bit integer representation.

If the sequence $((\uparrow_L A_n) \cap \text{lev}_d(L), \dots, (\uparrow_L A_{n+m-1}) \cap \text{lev}_d(L))$ is lexicographically minimal in its orbit under the action of S_d , then the computation of this orbit also yields generators of S_{d-1} [Cam99, § 1.13]; we limit the number of generators by applying a technique known as *Jerrum's filter* [Cam99, § 1.14].

4.1.3 Vertically indecomposable lattices

Restricting the construction to vertically indecomposable lattices is very easy:

Lemma 36. *Let L be a levelled n -lattice.*

- (a) *If L is vertically decomposable, then any levelled descendant $L_{A_n, \dots, A_{n+m-1}}$ of L is vertically decomposable.*
- (b) *If L is vertically indecomposable, then a levelled descendant $L_{A_n, \dots, A_{n+m-1}}$ of L is vertically decomposable if and only if $m = 1$ and $\uparrow_L A_n = L \setminus \{0\}$ hold.*

Proof. Let $k = \text{dep}_L(n-1)$ and let $\tilde{L} = L_{A_n, \dots, A_{n+m-1}}$.

- (a) Choose $i \in L$ such that $j \sqsubseteq_L i$ holds for any $j \in (A_n \cup \dots \cup A_{n+m-1}) \cap \text{lev}_k(L)$. For any $a \in \{n, \dots, n+m-1\}$, one has $A_a \cap \text{lev}_k(L) \neq \emptyset$, and thus $a \sqsubseteq_{\tilde{L}} i$.
- (b) This is obvious, as \tilde{L} is vertically decomposable if and only if there exists $a \in \{n, \dots, n+m-1\}$, such that one has $a \sqsubseteq_{\tilde{L}} i$ for all $i \in \tilde{L} \setminus \{0\}$. \square

4.2 Results and performance

Table 1 shows the number i_n of isomorphism classes of vertically indecomposable unlabelled lattices on n elements, and the number u_n of isomorphism classes of unlabelled lattices on n elements for $n \leq 20$; the values i_{20} and u_{20} are new.

Table 2 and Figure 7 show the total CPU time and the real time taken by the computations for $n \geq 14$ for two configurations:

- (A) 4 threads on a system with one 4-core Intel Xeon E5-1620 v2 CPU (clock frequency 3.70 GHz). The system load was just over 4 during the tests.
- (B) 32 threads on a system with eight 8-core Intel Xeon E7-8837 CPUs (clock frequency 2.67 GHz). The system load was around 55 during the tests.

All computations were done with a C-implementation of the described algorithm written by the first author, compiled using GCC with maximal optimisations for the respective architecture. The compiler version was 4.8.1 for configuration (A) and 4.4.7 for configuration (B).

Remark 37.

1. The computing times reported in [JL15] for $n = 18$ and $n = 19$ were 26 hours respectively 19 days on 64 CPUs, that is, 69 CPU-core days respectively 1216 CPU-core days.

Our values for configuration (B) are 1.6 respectively 19.1 CPU-core days. Assuming that the CPU clock speeds are comparable, this would correspond to a speedup by a factor of 43 respectively 63. In particular, it seems that the speedup increases with n .

Obtaining a meaningful complexity analysis seems out of reach, as estimating the average case complexity would require a detailed understanding of the tree of canonical lattices. Experimentally, the algorithm seems to be close to optimal in the sense that the computation time grows roughly linearly in the size of the output.

2. The throughput on configuration (A) corresponds to roughly 2 200 CPU clock cycles per lattice, including pre-computations and inter-thread communication.
3. On configuration (A), the L2 and L3 cache hit rates are on average around 45% respectively 75%; on configuration (B), this information could not be obtained. These data suggest that memory bandwidth is a main limiting factor, at least for the current implementation.
4. On configuration (B), it was not possible to influence the allocation of threads to individual nodes. Given the suboptimal scaling behaviour, which was worse than on configuration (A) even for 4 threads, we suspect that inter-node communication posed a problem on this architecture. However, as we did not have exclusive use of this machine, we were unable to investigate this question in detail.

References

- [Cam99] Peter J. Cameron. *Permutation groups*, volume 45 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 1999. [MR1721031](#)

n	1	2	3	4	5	6	7	8	9	10	11
i_n	1	1	0	1	2	7	27	126	664	3 954	26 190
u_n	1	1	1	2	5	15	53	222	1 078	5 994	37 622
n	12		13			14			15		16
i_n	190 754		1 514 332			12 998 035			119 803 771		1 178 740 932
u_n	262 776		2 018 305			16 873 364			152 233 518		1 471 613 387
n	17				18			19		20	
i_n	12 316 480 222				136 060 611 189			1 582 930 919 092		19 328 253 734 491	
u_n	15 150 569 446				165 269 824 761			1 901 910 625 578		23 003 059 864 006	

Table 1: Numbers i_n and u_n of isomorphism classes of vertically indecomposable unlabelled lattices, respectively arbitrary unlabelled lattices, on n elements.

- [HR02] Jobst Heitzig and Jürgen Reinhold. Counting finite lattices. *Algebra Universalis*, 48(1):43–53, 2002. [MR1930032](#)
- [JL15] Peter Jipsen and Nathan Lawless. Generating all finite modular lattices of a given size. *Algebra Universalis*, 74(3-4):253–264, 2015. [MR3397437](#)
- [KL71] Walter Klotz and Lutz Lucht. Endliche Verbände. *J. Reine Angew. Math.*, 247:58–68, 1971. [MR0276146](#)
- [Knu09] Donald E. Knuth. *The art of computer programming. Vol. 4, Fasc. 1*. Addison-Wesley, Upper Saddle River, NJ, 2009. Bitwise tricks & techniques; Binary decision diagrams.
- [KW80] D. J. Kleitman and K. J. Winston. The asymptotic number of lattices. *Ann. Discrete Math.*, 6:243–249, 1980. Combinatorial mathematics, optimal designs and their applications (Proc. Sympos. Combin. Math. and Optimal Design, Colorado State Univ., Fort Collins, Colo., 1978). [MR593536](#)
- [McK98] Brendan D. McKay. Isomorph-free exhaustive generation. *J. Algorithms*, 26(2):306–324, 1998. [MR1606516](#)
- [Slo] Neil J. A. Sloane. The on-line encyclopedia of integer sequences. <https://oeis.org>. Sequence A006966.

Volker Gebhardt
v.gebhardt@westernsydney.edu.au

Stephen Tawn
stephen@tawn.co.uk
<http://www.stephentawn.info>

Western Sydney University
Centre for Research in Mathematics
Locked Bag 1797, Penrith NSW 2751, Australia
<http://www.westernsydney.edu.au/crm>

	n	14	15	16	17	18	19	20
(A)	CPU [s]	8.3	70.7	694.7	7 357.9	83 752.1	—	—
	real [s]	2.3	18.5	178.2	1 895.3	21 493.8	—	—
(B)	CPU [s]	15.1	130.2	1 208.8	12 107.7	$1.4 \cdot 10^5$	$1.6 \cdot 10^6$	$2.1 \cdot 10^7$
	real [s]	1.5	7.3	51.2	446.7	5 267.2	65 152.9	$8.3 \cdot 10^5$

Table 2: Total CPU time and real time for the longer computations from [Table 1](#).

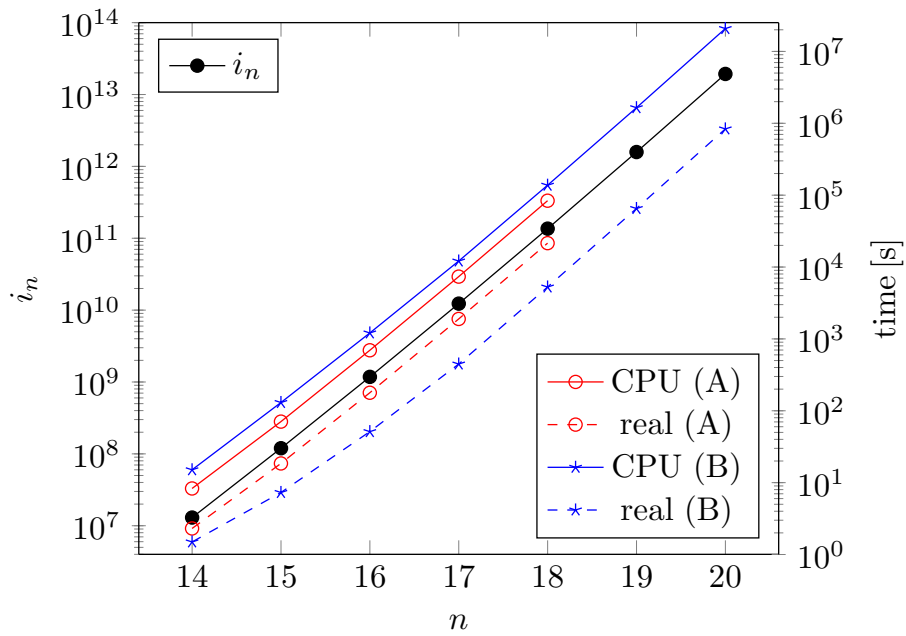


Figure 7: Growth of the number i_n of vertically indecomposable lattices, as well as of the required CPU time and real time on configurations (A) and (B), in terms of n .