Equivalence-Free Exhaustive Generation of Matroid Representations*

Petr Hliněný

Department of Computer Science,
VŠB – Technical University Ostrava,
17. listopadu 15, 70833 Ostrava, Czech Republic
petr.hlineny@vsb.cz

September 6, 2004

Abstract. In this paper we present an algorithm for the problem of exhaustive equivalence-free generation of 3-connected matroids which are represented by a matrix over some finite (partial) field, and which contain a given minor. The nature of this problem is exponential, and it appears to be much harder than, say, isomorph-free generation of graphs. Still, our algorithm is very suitable for practical use, and it has been successfully implemented in our matroid computing package MACEK [http://www.mcs.vuw.ac.nz/research/macek, 2002–2004].

Keywords: matroid representation, matroid extension, exhaustive generation, canonical construction path.

2000 Math Subjects Classification: 05-04, 05B35, 68R05.

1 Introduction

Matroids represented over a finite (partial) field play an important role in structural matroid theory, similar to the role that graphs embedded on a surface play in structural graph theory. However, unlike for embedded graphs, it is difficult to visualize a matroid in rank higher than 3, even when it is given as a matrix or a vector configuration. It is even more difficult to examine basic structural properties of given matroids like minors, connectivity, branch-width, representability, or matroid extensions.

It is often the case that proving a theorem in structural matroid theory requires one to check all the small cases (on, say, 8 or 10 elements) by hand, or to verify specific properties of such small matroids, which are often given by matrices over finite fields. As matroid researchers know very well themselves, checking all the "small cases" can be quite long and painful, and prone to errors. That is why a reasonably efficient algorithm for generation of matroids would

* This paper is originally based on research that the author performed at the Victoria University of Wellington in 2000–2002, supported by a New Zealand Marsden Fund research grant to Geoff Whittle.

be very helpful, serving as a base for an automated "small case analysis". Moreover, it turns out that in a typical case we are interested in matroids which are "extensions" of a certain small matroid, and so we do not want to generate all small matroids from scratch. Hence we focus here on an extension generating algorithm for matroids represented over finite fields.

This paper is structured as follows: The next section gives a brief overview of matroid terms, and it brings a more thorough consideration of matroid representability issues since these are crucial to proper understanding of the algorithm. (Moreover, relevant terminology is not quite settled, and so we have to clarify our use of terms.) Our main result – Algorithm 3.2 for generation of matroid extensions, is described and proved in Section 3. Formal description of the algorithm output, and of its consequences, are stated in the next Section 4. Finally, the appendix, Section 5, summarizes notes about practical implementation of the algorithm in our matroid computing package MACEK [4], and presents a brief running time analysis.

2 Basics of Matroids

We mostly follow Oxley [8] in matroid terminology. A *matroid* is a pair $M = (E, \mathcal{I})$ where $E = E(M)$ is the ground set of M (the *elements* of M), and $\mathcal{I} \subseteq 2^E$ is a nonempty collection of *independent sets* of M. For example, if $\boldsymbol{A}$ is a matrix, then the matroid formed by the column vectors of $\boldsymbol{A}$ with usual linear dependency is called the *vector matroid* of $\boldsymbol{A}$. For a graph G, the *cycle matroid* of G is formed on $E(G)$ by subforests of G. See an illustration in Fig. 1. Let $U_{r,n}$ denote the n-element uniform matroid of rank r.

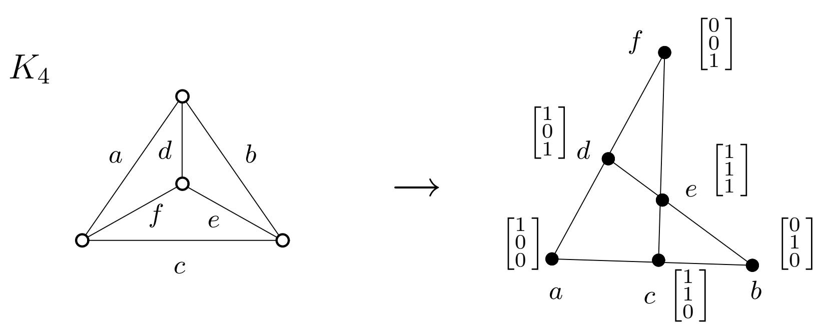

Fig. 1. An example of a vector representation of the cycle matroid of K_4. The matroid elements are depicted by dots, and their (linear) dependency is shown using lines.

We denote by $M \setminus e$ and M/e the matroids obtained by deleting and contracting, respectively, an element e in M. A *minor* of a matroid is obtained by a sequence of deletions and contractions of elements (these operations commute). Conversely, a matroid M' is a *one-element extension* (*coextension*) of M if $M = M' \setminus e$ ($M = M'/e$) for some element e. An *extension* of M is simply a matroid containing M as a minor.

For $n > 1$, a matroid M is *n-connected* if it has no k-separation for $k = 1, 2, \ldots, n-1$, and $|E(M)| \geq 2n-2$. (In particular, unlike [8], the matroid $U_{2,3}$ is not 3-connected.) Of particular interest to us are 3-connected matroids, which capture the core of most structural properties and problems on matroids. 3-connected matroids can be reasonably easily handled using Seymour's so called Splitter Theorem [12]:

Theorem 2.1. (Seymour) *Let M, N be 3-connected matroids such that N is a minor of M. Suppose that if N is a wheel (a whirl), then M has no larger wheel (no larger whirl) as a minor. Then there is a 3-connected matroid N_1 such that $|E(N_1)| = |E(N)| + 1$, and that M has an N_1-minor.*

The *k-wheel* W_k, $k \geq 3$, is the cycle matroid of the k-wheel graph. The *k-whirl* $\mathcal{W}^k$, $k \geq 2$, is obtained from the k-wheel by relaxing (making independent) the rim circuit. Specially $\mathcal{W}^2 \simeq U_{2,4}$. We say that a 3-connected matroid M is 3C-*reducible* to a matroid N if there is a sequence of 3-connected matroids $N_0 = N, N_1, \ldots, N_t = M$ such that $|E(N_i)| = |E(N_{i-1})| + 1$, and N_i has an N_{i-1}-minor for $i = 1, \ldots, t$. The following is a well-known corollary of Theorem 2.1:

Corollary 2.2. (Seymour) *Let M, N be 3-connected matroids such that N is a minor of M. If N is neither a wheel nor a whirl, then M is 3C-reducible to N. If N is a wheel (a whirl), and M has no larger wheel (no larger whirl) as a minor, then M is also 3C-reducible to N.*

Matroid Representations

We now turn our attention to matroids represented over a fixed finite field $\mathbb{F}$. This is a crucial part of our introductory definitions. A *representation* of a matroid M is a matrix $\boldsymbol{A}$ over $\mathbb{F}$ whose columns correspond to the elements of M, and linearly independent subsets of columns form the independent sets of M. Clearly, the matroid of $\boldsymbol{A}$ is unchanged when columns are scaled by non-zero elements of $\mathbb{F}$. So we may alternatively view the matrix $\boldsymbol{A}$ as a point configuration in a projective space over $\mathbb{F}$.

A matroid M is *regular* if M is representable by a totally-unimodular matrix. A regular matroid is then representable over all fields. A matroid M is *binary*, or *ternary*, if M is representable over the fields $GF(2)$, or $GF(3)$, respectively. We remark that cycle matroids of graphs are regular. Not all matroids are representable over a field $\mathbb{F}$, some of them are even representable over no field at all. One also has to consider the problem that representable matroids typically do not have "unique" representations. As a simple example, we present in Fig. 2 two point configurations representing the same 9-element rank-3 matroid which are not "equivalent" in any reasonable geometric meaning of equivalence.

Another issue, which has to be particularly considered in the context of exhaustive generation, is the one of labeled vs. unlabeled objects: We are interested in generating unlabeled objects to avoid unnecessary duplicities, while the objects generated by a computer are (usually) implicitly labeled.

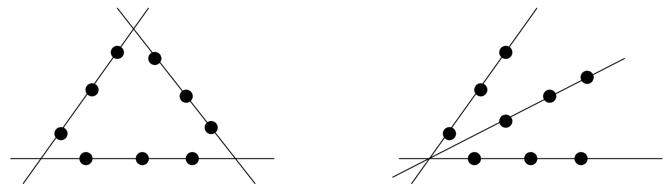

Fig. 2. Two non-equivalent representations of a 9-element rank-3 matroid.

Let the finite field $\mathbb{F}$ be fixed from now on. We denote by $[r, s] = \{r, r+1, \ldots, s\}$. An $r \times n$ matrix $\boldsymbol{A}$ for $r \leq n$ is called here a *labeled matrix* if $\boldsymbol{A}$ has rank r and the columns of $\boldsymbol{A}$ are labeled by numbers $1, 2, \ldots, n$ (rows are not labeled). Then $\boldsymbol{A}$ represents a rank-r matroid $M = M(\boldsymbol{A})$ on the ground set $E(M) = [1, n]$, and we speak about the columns of $\boldsymbol{A}$ as about matroid elements. We say that $\boldsymbol{A}$ is in a *standard form* if some basis of M is displayed as an $r \times r$ identity submatrix in $\boldsymbol{A}$, i.e. $\boldsymbol{A} = [\boldsymbol{I}_r \mid \boldsymbol{A}']$. Then the matrix $\boldsymbol{A}'$ is called a *reduced* matrix of the matroid M (or of $\boldsymbol{A}$). Naturally, the rows of $\boldsymbol{A}'$ are labeled by the column labels of $\boldsymbol{I}_r$. Moreover, we say that a matrix $\boldsymbol{A}$ is in *ordered* standard form if the columns of $\boldsymbol{A}$ are ordered by their labels, so that the lexicographically minimal basis of M is shown as the identity submatrix $\boldsymbol{I}_r$.

Let $\mathcal{A}$ be the class of all labeled matrices over $\mathbb{F}$. Each matrix $\boldsymbol{A} \in \mathcal{A}$ can be turned into a standard form by elementary row operations (which do not change the matroid). We say that two matrices $\boldsymbol{A_1}, \boldsymbol{A_2} \in \mathcal{A}$ are equivalent if their ordered standard forms are equal up to non-zero scaling of rows and columns. These equivalence classes on $\mathcal{A}$ are called *labeled represented matroids.* A labeled represented matroid with n elements has the ground set $[1, n]$. In the language of [8, Chapter 6], two matrices $\boldsymbol{A}_1, \boldsymbol{A}_2$ belong to one labeled represented matroid if and only if $\boldsymbol{A}_1, \boldsymbol{A}_2$ are equivalent without use of $\mathbb{F}$-automorphisms, otherwise called *strongly equivalent* in matroid theory for distinction.

If M is a labeled represented matroid on n elements, and $\pi \in S_n$ is a permutation of $[1, n]$, then π is called a *relabeling*, and M^π is the labeled represented matroid obtained from M by applying π to the column labels. Let $\mathcal{L}$ be the set of all labeled represented matroids. The orbits of $\mathcal{L}$ under the action of the relabeling symmetric group are called *unlabeled represented matroids*, and their set is denoted by $\mathcal{U}$. Note that unlabeled represented matroids refine the isomorphism classes of the underlying abstract matroids. (For example, these two notions are identical over binary matroids.) Table 1 presents, for an illustration, the numbers of labeled and unlabeled represented matroids isomorphic to selected small matroids over several fields. (The numbers have been computed with Macek [4], using its representability-testing feature. We refer to Section 5 for more details.)

Further in the paper, it is very practical to work only with the reduced matrix $\boldsymbol{A}'$ instead of $\boldsymbol{A} = [\boldsymbol{I}_r \mid \boldsymbol{A}']$. We say that such $\boldsymbol{A}'$ *displays* a basis B of $M(\boldsymbol{A})$ where B is formed by the labels of $\boldsymbol{I}_r$. We note that $\boldsymbol{A}'^T$ is a reduced matrix of the dual matroid $M(\boldsymbol{A})^*$, and that removing a column (a row) of $\boldsymbol{A}'$ means deleting (contracting) the corresponding element in the matroid $M(\boldsymbol{A})$. (When

Table 1. The numbers of labeled / unlabeled represented matroids over small fields.

represented \ *matroids*	$U_{2,4}$	$U_{2,5}$	$U_{2,6}$	$U_{3,6}$	$\mathcal{W}^3$	$U_{2,7}$	$U_{3,7}$
$GF(5)$	3/1	6/1	6/1	6/1	3/2	0/0	0/0
$GF(7)$	5/2	20/1	60/1	140/3	5/3	120/1	120/1
$GF(8)$	6/1	30/1	120/1	390/5	6/3	360/1	1200/2
$GF(9)$	7/2	42/2	210/2	882/7	7/4	840/1	6120/4

going to contract a non-loop element x in M, one can always display a basis $B \ni x$ first.)

Lastly, we define the notion of a represented minor. Let M, N be unlabeled represented matroids, and let $\boldsymbol{A}_0$ be a reduced matrix for (some labeling of) N. We say that N is a *represented minor* of M if a reduced matrix displaying some basis of M contains a submatrix equal to $\boldsymbol{A}_0$ up to non-zero scaling. It is an easy linear-algebra exercise to show that this definition does not depend on a particular choice of $\boldsymbol{A}_0$ for N. If N is a represented minor of M, then N is isomorphic to a (usual) minor of M, but the converse may not always be true if there is more than one unlabeled represented matroid of the same abstract matroid N.

3 Generating Matroid Elimination Sequences

Recall that we work over a fixed finite field $\mathbb{F}$. We extend the definition of "3C-reducible" from the previous section to represented minors in the natural way. The set of all 3-connected unlabeled represented matroids (over $\mathbb{F}$) is denoted by $\mathcal{U}_3$. Let $Q \in \mathcal{U}_3$ be a fixed unlabeled represented matroid, called the *base minor*. We define $\mathcal{U}_3^Q \subseteq \mathcal{U}_3$ as the subset of all $M \in \mathcal{U}_3$ such that Q is a represented minor of M.

Our task now is to *generate unique labeled representatives* for the unlabeled represented matroids from $\mathcal{U}_3^Q$, i.e. 3-connected *represented extensions* of Q. Since we are going to apply Corollary 2.2, there will be some minor exceptions specified later. Our approach follows the general scheme of McKay [7] – generation via a "canonical construction path". However, significant difference of represented matroids from graphs (and from other classical objects) makes it very complicated to express our generating algorithm within the mentioned general scheme, and so we have chosen to present the algorithm as a standalone procedure. The presented procedure directly leads to the practical implementation in Macek [4].

Elimination sequences

Let p, m, r, n be integers, where $p \leq m, r \leq n$ and $m-p \leq n-r$, let $q = (q_i)_{i=m+1}^{n}$ be a $\{0,1\}$-sequence, and let $\boldsymbol{A}_0 \in \mathbb{F}^{p \times (m-p)}$ and $\boldsymbol{A} \in \mathbb{F}^{r \times (n-r)}$ be matrices. Suppose that $\sum_{i=m+1}^{n} q_i = n-r-m+p$, and denote by $\boldsymbol{A}_l$, $l \in [m,n]$ the upper-left submatrix of $\boldsymbol{A}$ on $j = p + \sum_{i=m+1}^{l}(1-q_i)$ rows and $k = l - j$ columns.

Then the triple $S = (\boldsymbol{A}_0, \boldsymbol{A}, q)$ is called an *elimination sequence* if the following are true:

- $\boldsymbol{A}_m = \boldsymbol{A}_0$,
- each $\boldsymbol{A}_l$, $l \in [m, n]$ is a reduced matrix of a 3-connected matroid, and
- the first non-zero entry is 1 for each row or column in $\boldsymbol{A}$ not intersecting $\boldsymbol{A}_0$.

The *length* of S is $\|S\| = n - m$. For further reference, denote by $\boldsymbol{A}_i(S)$ the submatrix $\boldsymbol{A}_i$ defined above, by $q(S)$ the *sequence signature* q, and by $\boldsymbol{u}_i(S)$, $i \in [m+1, n]$ the vector (row or column) which is added to $\boldsymbol{A}_{i-1}(S)$ to form $\boldsymbol{A}_i(S)$. See an illustration in Fig. 3.

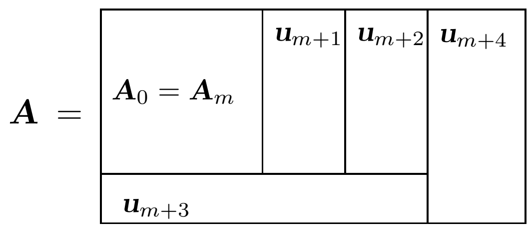

Fig. 3. An illustration of an elimination sequence $(\boldsymbol{A}_0, \boldsymbol{A}, q)$ where $q = (1, 1, 0, 1)$.

For simplicity, we no longer explicitly speak about labeled matroids in this section, but, instead, we assume that our matroids have implicit labelings associated with their elimination sequences. (In particular, we first assign some implicit labeling to the base minor $M(\boldsymbol{A}_0)$.) Keeping this in mind, we say that S *produces* the labeled represented matroid $M(\boldsymbol{A})$. We let $\mathcal{S}(\boldsymbol{A}_0, \boldsymbol{A})$ be the set of all elimination sequences $(\boldsymbol{A}_0, \boldsymbol{A}, q)$ for admissible choices of q; and $\mathcal{S}(\boldsymbol{A}_0, M)$, $M \in \mathcal{U}_3$ be the union of $\mathcal{S}(\boldsymbol{A}_0, \boldsymbol{A}_M)$ over all reduced matrices $\boldsymbol{A}_M$ such that $M(\boldsymbol{A}_M) \in M$. (Note that, by definition, $\mathcal{S}(\boldsymbol{A}_0, \boldsymbol{A}_M) = \emptyset$ if $\boldsymbol{A}_M$ is not compatible with $\boldsymbol{A}_0$.) In other words, $\mathcal{S}(\boldsymbol{A}_0, M)$ is the set of all elimination sequences with the base $\boldsymbol{A}_0$ producing some labeling of unlabeled represented matroid M. We say that the elimination sequences in $\mathcal{S}(\boldsymbol{A}_0, M)$ are pairwise *equivalent*.

We now define a linear order $\prec$ on $\mathcal{S}(\boldsymbol{A}_0, M)$ as follows: $S_1 \prec S_2$ for equivalent S_1, S_2 iff $q(S_1)$ is lexicographically smaller than $q(S_2)$, or $q(S_1) = q(S_2)$ and the sequence of vectors $\big(\boldsymbol{u}_i(S_1)\big)_{i=m+1}^{n}$ is lexicographically smaller than the sequence $\big(\boldsymbol{u}_i(S_2)\big)_{i=m+1}^{n}$. We say that an elimination sequence $S' = (\boldsymbol{A}_0, \boldsymbol{A}', q')$ is a subsequence of S if $\boldsymbol{A}' = \boldsymbol{A}_i(S)$ for some $i \in [m, n]$, and q' is the corresponding prefix of $q(S)$. An elimination sequence S is a *k-step extension* of an elimination sequence S' if S' is a subsequence of S and $\|S\| - \|S'\| = k$.

Lemma 3.1. *Let S_1', S_2' be equivalent elimination sequences, and let S_1, S_2, respectively, be 1-step extensions of S_1', S_2'. If $S_1' \prec S_2'$, and S_1, S_2 are also equivalent, then $S_1 \prec S_2$.*

Proof. If $q(S_1')$ is lexicographically smaller than $q(S_2')$, then so is $q(S_1)$ smaller than $q(S_2)$, and hence $S_1 \prec S_2$. On the other hand, if $q(S_1') = q(S_2')$, then $q(S_1) = q(S_2)$ since, in particular, the matrices $\boldsymbol{A}(S_1)$ and $\boldsymbol{A}(S_2)$ must have the same size. Hence the second lexicographic criterion applies also to comparing S_1 and S_2, and thus $S_1 \prec S_2$ again. ∎

The generating algorithm

Let $Q \in \mathcal{U}_3$ (a base minor), and let $\boldsymbol{A}_Q$ be a reduced matrix representing Q. Let $\mathcal{V}_3^Q \subseteq \mathcal{U}_3^Q$ be the set of all unlabeled represented matroids $M \in \mathcal{U}_3^Q$ that are 3C-reducible to the represented minor Q. By Corollary 2.2, $\mathcal{V}_3^Q = \mathcal{U}_3^Q$ unless Q is a wheel or a whirl.

For each unlabeled represented matroid $M \in \mathcal{V}_3^Q$, we are going to generate the smallest (with respect to $\prec$) elimination sequence $S_M \in \mathcal{S}(\boldsymbol{A}_Q, M)$ by the following recursive procedure:

Algorithm 3.2 *Recursive generation of all non-equivalent k-step extensions, $k \leq \ell$, of an elimination sequence S based on $\boldsymbol{A}_Q$.*

```
procedure mgenerate(S : 𝒮(A_Q, ·))
   output S;
   if ‖S‖ ≥ ℓ then return;
   for  each 1-step extension S′ of S  do
      let M ∈ 𝒰₃ be the matroid produced by S′ (s.t. S′ ∈ 𝒮(A_Q, M));
      set 𝒟 = all reduced matrices A representing M such that
                  the top-left submatrix of A is equal to A_Q;
      set d = ‖S′‖;
      for  each q ∈ {0,1}^d and each D ∈ 𝒟  do
         if  (A_Q, D, q) is an elimination sequence  then
            if  (A_Q, D, q) ≺ S′  then return;
      done
      # (Tests for other required structural properties may be inserted above...)
      mgenerate(S′);
   done
end.
```

Lemma 3.3. *Let $S_Q = (\boldsymbol{A}_Q, \boldsymbol{A}_Q, \emptyset)$ for Q and $\boldsymbol{A}_Q$ as above. A call to* `mgenerate`(S_Q) *for $\ell \geq 0$ outputs exactly one elimination sequence $S_M \in \mathcal{S}(\boldsymbol{A}_Q, M)$ for each unlabeled represented matroid $M \in \mathcal{V}_3^Q$ such that $|E(M)| \leq \ell + |E(Q)|$.*

In other words, `mgenerate`(S_Q) outputs the collection of all non-equivalent elimination sequences (without duplicities) of length at most ℓ extending S_Q.

Proof. We prove the statement by induction on ℓ. If $\ell = 0$, then `mgenerate`(S_Q) outputs S_Q and quits. Suppose that the statement is proved for $\ell - 1$ where $\ell > 0$.

Let $M \in \mathcal{V}_3^Q$ be an unlabeled represented matroid on $n = \ell + |E(Q)|$ elements. Since M is 3C-reducible to the represented minor Q, the set $\mathcal{S}(\boldsymbol{A}_Q, M)$ is nonempty by definition. Let $S_0 \in \mathcal{S}(\boldsymbol{A}_Q, M)$ be minimal with respect to

$\prec$, and let N be the unlabeled represented matroid given by the reduced matrix $\boldsymbol{A}_{n-1}(S_0)$. By the inductive assumption, our procedure outputs a sequence $S_1 \in \mathcal{S}(\boldsymbol{A}_Q, N)$, $S_1 = (\boldsymbol{A}_Q, \boldsymbol{A}_N, q^1)$, which is minimal with respect to $\prec$. Then there is always a reduced matrix $\boldsymbol{C}_M$ representing M such that $\boldsymbol{A}_N$ is a top-left submatrix of $\boldsymbol{C}_M$. (Actually, $\boldsymbol{C}_M$ extends $\boldsymbol{A}_N$ just by one row or column, depending on the signature $q(S_0)$.) We let q^2 be the signature obtained from q^1 by appending the last element of $q(S_0)$. Hence $S_2 = (\boldsymbol{A}_Q, \boldsymbol{C}_M, q^2) \in \mathcal{S}(\boldsymbol{A}_Q, M)$ is an elimination sequence which is considered among the 1-step extensions of S_1 in the procedure `mgenerate`.

We claim that, for some $\boldsymbol{C}_M$ chosen as above, the sequence $S_2 = (\boldsymbol{A}_Q, \boldsymbol{C}_M, q^2)$ is minimal in $\mathcal{S}(\boldsymbol{A}_Q, M)$ with respect to $\prec$, i.e. that $S_2 = S_0$. Indeed, let S_0' be the subsequence of S_0 of length $n-1$. If $\boldsymbol{A}_{n-1}(S_0') \neq \boldsymbol{A}_N = \boldsymbol{A}_{n-1}(S_2)$ or $q(S_0') \neq q^1$, then $S_0' \neq S_1$, and so $S_1 \prec S_0'$ by the minimality of S_1. Now Lemma 3.1 implies $S_2 \prec S_0$, a contradiction to minimality of S_0. Therefore, $\boldsymbol{A}_{n-1}(S_0') = \boldsymbol{A}_N$ and $q(S_0') = q^1$, so also $q(S_0) = q^2$. Then the (minimal) sequence S_2 for $\boldsymbol{C}_M = \boldsymbol{A}_n(S_0)$ is generated in the algorithm. The claim is proved.

We now assume, for a contradiction, that the procedure `mgenerate` outputs two $S_1, S_2 \in \mathcal{S}(\boldsymbol{A}_Q, M)$. If S_1, S_2 were not identical, then the larger one in $\prec$ would be rejected in the inner `for` cycle of the procedure. Hence $S_1 = S_2$, which means that the subsequence $S_1' = S_2' \in \mathcal{S}(\boldsymbol{A}_Q, N)$ of length $n-1$ was output twice by the procedure, a contradiction. ∎

Application notes

The procedure `mgenerate` of Algorithm 3.2 contains two nontrivial steps — generating all 1-step extensions of an elimination sequence S, and generating the set $\mathcal{D}$. The first task is not difficult when working over a finite field $\mathbb{F}$. We simply produce all row and column vectors over $\mathbb{F}$ of appropriate length, and starting with the first non-zero entry 1. (However, this task gets complicated when considering a partial field $\mathbb{F}$, as will be discussed in Section 5.)

The set $\mathcal{D}$ of matrices can be huge, but we need to generate the whole of $\mathcal{D}$ only if we are going to accept the extension S'. If we want to reject S', it is enough to guess suitable q and $\boldsymbol{D}$ such that $S_0 = (\boldsymbol{A}_Q, \boldsymbol{D}, q) \prec S'$. Our implementation of Algorithm 3.2 in [4] uses several heuristics to guess such a smaller equivalent sequence S_0, to speed-up rejection of non-minimal S'. Notice that generating the set $\mathcal{D}$ includes the task of finding all represented minors equal to Q in M, which itself is an interesting and useful procedure. We refer to Section 5 for a more detailed discussion of this topic.

In many situations we are interested only in those members of $\mathcal{V}_3^Q$ that satisfy some additional structural conditions, for example, that the matroid contains no represented minor in a given set of forbidden minors. It is useful to implement tests for such additional conditions directly in the procedure `mgenerate`, since these tests may reject an unsuitable extension S' faster than the canonical minimality test. Moreover, it may be desired to implement structural restrictions not only on the resulting unlabeled represented matroids, but also on the elimination sequences themselves. As examples we mention possible requirements of "sequential 4-connectivity" or of absence of "long fans" along the elimination sequence.

Finally, we present a practical small example of generating 1-step extensions of the 3-wheel $Q = W_3$ in the ternary field $GF(3)$, as it is implemented in MACEK [4]. We choose a reduced matrix

$$\boldsymbol{A}_Q = \begin{bmatrix} 1 & 0 & 2 \\ 2 & 1 & 0 \\ 0 & 2 & 1 \end{bmatrix} .$$

Since W_3 is self-dual, we focus only at column-extensions of $\boldsymbol{A}_Q$, i.e. at 7-element rank-3 ternary unlabeled represented matroids containing W_3 as a represented minor. See an illustration in Fig. 4.

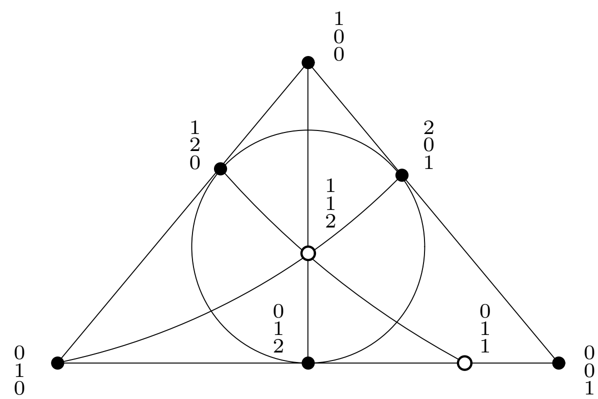

Fig. 4. Two non-equivalent ternary one-step extensions to the 3-wheel W_3 (solid points are the elements of W_3, and hollow points are the extension points).

The procedure `mgenerate` in Algorithm 3.2 generates 14 column vectors from $GF(3)^3$, and 7 of them violate the 3-connectivity condition. Among the remaining 7 extensions, the four ones $(1,0,1)^T, (1,1,0)^T, (1,2,1)^T, (1,2,2)^T$ are quickly rejected since smaller equivalent sequences are found using heuristics based on rich symmetries of the matrix $\boldsymbol{A}_Q$. An extension given by the vector $(1,1,1)^T$ is rejected since a smaller equivalent elimination sequence is found by an exhaustive search. The remaining two vectors $(0,1,1)^T$ and $(1,1,2)^T$ pass the minimality test, and they give two non-equivalent extensions of $Q = W_3$ over $GF(3)$, as depicted in Fig. 4.

Typically, Algorithm 3.2 shall be used to generate larger extensions than in this example, say on 10 or 12 elements. Then the number of extension vectors, which have to be considered in each 1-step extension, grows exponentially, and the complexity of the involved operations also grows rapidly. Besides that, the computing performance heavily depends on the size of the field $\mathbb{F}$. A brief practical performance analysis of our matroid generator is included in Section 5. Here we would like to mention that possible symmetries of the base matrix $\boldsymbol{A}_Q$ do not seem to influence the performance of the computation much. The likely explanation is that, while symmetries allow for various heuristic speed-ups in the implementation of Algorithm 3.2, symmetric matroids also generate more equivalent extensions that have to be rejected by an exhaustive search. Remarkably, the computation seems to be slower for base matroids Q with highly transitive automorphism group.

4 About Matroid Generation

We now summarize the properties and consequences of our generating algorithm.

Theorem 4.1. *Let $\ell \geq 0$, $Q \in \mathcal{U}_3$ be a 3-connected unlabeled represented matroid, and $S_Q = (\boldsymbol{A}_Q, \boldsymbol{A}_Q, \emptyset)$ be an elimination sequence, where $\boldsymbol{A}_Q$ is a reduced representation of Q. For each unlabeled represented matroid $M \in \mathcal{V}_3^Q$ on $n \leq \ell + |E(Q)|$ elements, Algorithm 3.2 on S_Q and ℓ produces exactly one labeled represented matroid (a representative) $L \in M$.*

Proof. This statement is a straightforward reformulation of Lemma 3.3. As it was already noted there, each matroid $M \in \mathcal{V}_3^Q$ is 3C-reducible to the unlabeled represented matroid Q, and hence there exists an elimination sequence producing a labeled represented matroid $L \in M$ from $\boldsymbol{A}_Q$. Formally, $\mathcal{S}(\boldsymbol{A}_Q, M) \neq \emptyset$. On the other hand, we know from Lemma 3.3 that only one elimination sequence from $\mathcal{S}(\boldsymbol{A}_Q, M)$ is generated, and hence only one representative $L \in M$ is produced. ∎

In connection with Corollary 2.2, and with the definitions of represented matroids from page 4, we formulate the following immediate conclusion:

Corollary 4.2. *Let Q be a 3-connected unlabeled represented matroid over $\mathbb{F}$, that is neither a wheel, nor a whirl. Then Algorithm 3.2 produces the collection of all unlabeled strongly non-equivalent $\mathbb{F}$-representations of the matroids on $n \leq \ell + |E(Q)|$ elements, containing Q as a represented minor.* ∎

There are several important notes explaining the consequences of this statement: Say, even if one is interested in abstract matroids only, it is necessary to generate all inequivalent representations in our algorithm, since not all representations of the same abstract matroid extend further in the same way. In particular, if one wants to exhaustively enumerate all non-isomorphic $\mathbb{F}$-representable matroid extensions of abstract Q, (s)he has to consider all unlabeled represented matroids forming Q as the starting points. Isomorphic pairs of matroids (in the case of fields $\mathbb{F}$ larger than $GF(3)$) could be removed afterward.

Moreover, it is very important for practical large-scale computations that the implementation our matroid generator can be easily parallelized, even without need for inter-process communication. This is formally described in the following claim which is immediate from the description of Algorithm 3.2:

Proposition 4.3. *Let $\ell, \ell' > 0$, and let $S_Q = (\boldsymbol{A}_Q, \boldsymbol{A}_Q, \emptyset)$ be an elimination sequence of length 0 with a base minor Q. Denote by $\mathcal{T}$ the set of all ℓ-step extensions of the sequence S_Q generated by Algorithm 3.2 on S_Q and l. For each unlabeled represented matroid $M \in \mathcal{V}_3^Q$ on n, $\ell < n - |E(Q)| \leq \ell + \ell'$, elements, there is exactly one $S_1 \in \mathcal{T}$ such that Algorithm 3.2, run on S_1 and ℓ', produces a labeled represented matroid $L \in M$.*

∎

Speaking informally, at any step of a generation process, one may redistribute the set of generated elimination sequences $\mathcal{T}$ among several computers, and to continue the generation process in independent parallel threads.

Exhaustive generation of matroids

So far, our approach to matroid generating focused on matroid extensions. Here we briefly mention how to exhaustively generate all (small) 3-connected matroids representable over some small field using Algorithm 3.2 (Theorem 4.1). See in Table 2.

By Tutte's "Wheels and Whirls" Theorem [14] (a predecessor of Theorem 2.1), every 3-connected matroid is 3C-reducible to some wheel or a whirl. However, it is technically rather difficult (though possible) to compute extensions from a starting list of many matroids of different sizes, and to avoid duplicities in the results. That is why we have used such an approach only for generation of regular matroids, where there are only few of them.

We exhaustively generate all 3-connected non-regular binary matroids as the 3-connected extensions of the Fano plane F_7, using a result of [13]:

Theorem 4.4. (Tutte) *A binary matroid is not regular if and only if it contains an F_7-minor.*

By another result of Tutte [13], all non-binary matroids contain a $U_{2,4}$-minor. Unfortunately, $U_{2,4}$ (isomorphic to the 2-whirl) is one of the exceptions in Theorem 2.1. An enhancement of this theorem [1] (also in [8, Section 11.3]) states that, briefly saying, the exceptions in Theorem 2.1 can be narrowed down to the 2- and 3-whirls in a non-binary case. We present the following formulation:

Theorem 4.5. (Coullard) *Each 3-connected non-binary matroid that is not a whirl has a 3-connected single-element extension or coextension of $U_{2,4}$ or of the 3-whirl $\mathcal{W}^3$ as a minor.*

Using Corollary 2.2 and the fact that $U_{2,5}$ is not $GF(3)$-representable, it is now clear that all 3-connected ternary non-binary matroids except the whirls are 3C-reducible to $\mathcal{W}^3$. Hence we can generate the 3-connected ternary non-binary matroids as extensions of $\mathcal{W}^3$, and then add all the other whirls (which are all $GF(3)$-representable).

Stepping further to larger fields, we use the result of [10]:

Theorem 4.6. (Semple, Whittle) *A 3-connected non-binary non-ternary matroid M representable over some field has a $U_{2,5}$-minor, unless M is isomorphic to $U_{k,k+2}$ for some $k \geq 3$.*

Since $U_{2,5}$ has only one unlabeled represented matroid over $GF(q)$ for $q = 4, 5, 7, 8$, this theorem allows us to exhaustively generate all non-binary non-ternary matroids represented over $GF(q)$, starting from some representation of $U_{2,5}$. Of course, we then have to include representations of $U_{k,k+2}$ for appropriate values of k. (Note, however, that $U_{2,5}$ has two unlabeled represented matroids over $GF(9)$, and so exhaustive generation over $GF(9)$ is not as straighforward in this setting.) Moreover, considering that matroids may have non-equivalent representations (and typically they have) over fields larger than $GF(3)$, we may finally want to remove isomorphic pairs of matroids from the resulting lists.

We summarize the matroid enumeration results that we have obtained with Macek, using the above described procedures, in Table 2.

Table 2. The numbers of small 3-connected matroids representable over small fields (generated all as unlabeled represented matroids).

representable \ elements	4	5	6	7	8	9	10	11	12	13	14	15
regular:	0	0	1	0	1	4	7	10	33	84	260	908
$GF(2)$, non-regular:	0	0	0	2	2	4	17	70	337	2080	16739	181834
$GF(3)$, non-regular:	1	0	1	6	23	120	1045	14116	330470	?	?	?

(Next we present both the numbers of non-equivalent and of non-isomorphic ones.)

representable \ elements	4	5	6	7	8	9	10	11
$GF(4)$, non-$GF(2,3)$:	0	2	2	8	78	1040	26494	1241588
– non-isomorphic:	0	2	2	8	69	748	15305	?
$GF(5)$, non-$GF(2,3,4)$:	0	0	3	16	271	8336	497558	?
– non-isomorphic:	0	0	3	12	192	6590	?	?
$GF(7)$, non-$GF(2,3,4,5)$:	0	0	0	18	1922	252438	?	?
– non-isomorphic:	0	0	0	10	277	97106	?	?
$GF(8)$, non-$GF(2,3,4,5,7)$:	0	0	0	0	94	?	?	?
– non-isomorphic:	0	0	0	0	20	?	?	?

5 Appendix: Implementation and Practical Use

In this appendix we add few words about practical implementation of our matroid generation algorithm in Macek [4]. We note that Macek has been successfully used in the exhaustive search for the excluded minors for matroids of branch-width three [5] and [6], and that some other researchers have also reported success with Macek assisting their research, such as [15], [16].

The Macek Program

We have developed the computer program Macek [4] for practical structural computations with matroids represented over finite (also partial) fields. The Macek program is free, distributed under the terms of the GNU General Public License as published by the Free Software Foundation. See [4] for information about how to obtain and install the program. Macek supports easy manipulation and computations with matrices representing matroids. Among matroidal functions, one can test for matroid minors, equivalence, representability, isomorphism, branch-width three, connectivity, etc. A key feature of the program is an implementation of our exhaustive generation Algorithm 3.2.

We use a bit of space in the appendix to outline the ways how we have implemented other important structural matroid functions in Macek – finding a represented minor, testing abstract isomorphism, and testing / generating matroid representations over other fields. (The first task – finding represented minor, actually is a key ingredience in computing the set $\mathcal{D}$ in Algorithm 3.2. The other two tasks are not needed in that algorithm, but they are very useful in practical applications, and particularly in testing correctness of our generator implementation.)

We, however, remark that the following outlined routines are of little theoretical interest since they present just clever implementations of basic brute-force approaches. The usual instances of the input in these cases are so small that a fast implementation of a brute-force algorithm is better than more sophisticated algorithms.

- Finding a represented minor N in an unlabeled represented matroid M: Let N be represented by a reduced matrix $\boldsymbol{A}_N$, and M by $\boldsymbol{A}_M$. By successive pivoting in $\boldsymbol{A}_M$, we obtain all the (unlabeled) representations of M displaying each of the bases of M. Then, in each of them, we search for submatrices which are, up to line order and scaling, equal to $\boldsymbol{A}_N$. To speed up the search (quite significantly, in fact), we use matching of patterns of zero entries and of zero-valued 2×2-subdeterminants in those matrices.
- Testing abstract isomorphism between two matroids M_1, M_2: We precompute various element-based invariants of the matroids, based mostly on the structure of small flats (and hence polytime). If those invariants of M_1 and M_2 match each other, and the matroids have the same number of bases, then we take a reduced matrix $\boldsymbol{A}_1$ of M_1 and generate the reduced matrices $\boldsymbol{A}_2^i$ displaying all bases of M_2. For each such $\boldsymbol{A}_2^i$, we check all orderings of the lines of $\boldsymbol{A}_2^i$ (with help of the above invariants), and compare all the subdeterminants of $\boldsymbol{A}_1$ against those of $\boldsymbol{A}_2^i$. If we (ever) find a match, then M_1 and M_2 are isomorphic.
- Finding all representations over $\mathbb{F}$ of a matroid M (or, testing representability): Let M be represented by a reduced matrix $\boldsymbol{A}_M$ (possibly over a different field). We are building a sequence of matrices, starting from empty and ending with one of size equal to $\boldsymbol{A}_M$. At each step, we add one row or column to the previous matrix – we try all unit-scaled vectors over $\mathbb{F}$ and choose those which produce a matrix representation which is (as an

abstract matroid) isomorphic to the corresponding submatrix of $\boldsymbol{A}_M$. At the end, we get all labeled represented matroids over $\mathbb{F}$ isomorphic to M.

Representations over partial fields

In addition to finite fields, MACEK can work with matroids represented over finite partial fields. A *partial field* is a generalization of a field, in which addition is a partial operation. We refer to [11] for a formal definition and properties of partial fields. A well-known example (though not under this name) is the *regular* partial field consisting of the integers $-1, 0, 1$ with usual addition and multiplication. A matrix $\boldsymbol{A}$ over a partial field $\mathbb{P}$ is *proper* if all subdeterminants of $\boldsymbol{A}$ are defined in $\mathbb{P}$. For example, proper regular matrices are traditionally known as totally unimodular. A matroid N is representable over $\mathbb{P}$ if there is a proper matrix $\boldsymbol{A}$ over $\mathbb{P}$ such that $N \simeq M(\boldsymbol{A})$.

A partial field is called *finite* if the equation $x - 1 = y$ has finitely many solutions in $\mathbb{P}$. All finite fields are clearly finite in this sense. However, a finite partial field may have infinitely many elements. (The reason for our terminology is that a fixed-rank simple matroid representable over a finite partial field may have only finite number of elements.)

We briefly describe how our generation algorithm can be used to generate matroid extensions over finite partial fields. Basically, all parts of Algorithm 3.2 run smoothly here, except the step generating all extension vectors to a matrix over $\mathbb{F} = \mathbb{P}$: There is, potentially, an infinite number of values for each new entry. Fortunately, the first nonzero entry is always 1, and for each next entry there is a subdeterminant in the matrix whose definability over $\mathbb{P}$ reduces to finding the finitely many solutions to $x - 1 = y$ in $\mathbb{P}$. Hence we can efficiently generate the finitely many potential extension vectors, and then select those producing proper matrices over $\mathbb{P}$.

Reliability of Computation

Theoretical correctness of our matroid generator – Algorithm 3.2, is proved in Theorem 4.1. However, a natural question arises about reliability of its implementation in MACEK. There is, unfortunately, no large-scale computation data about matroid generation available in the literature, and so we have little chance to compare our computing results with other reliable sources. We briefly mention two of just a few small exceptions:

- J. Dharmatilake [2] implemented an exhaustive search of binary matroids up to 12 elements, for the purpose of finding all binary excluded minors for the class of matroids of branch-width three. Our generation algorithm in MACEK turned out to be much more efficient than his approach, and so we were able to easily finish the search of binary matroids up to 14 elements [5] (which is a known upper bound on size of such an excluded minor). Besides that, we have also compared parts of his computing data with our results.

– R. Pendavingh [9] has recently carried out an enumeration of small matroids, to find excluded minors for matroid representability over the fields $GF(5)$ and $GF(7)$. Independently from him, we have run similar search with Macek, and the common parts of our results matched each other.

Besides those, we are left with another possibility – to compare our computation results with other results obtained with the same program (Macek). Although that may sound almost like cheating, we are going to convince the reader that, in a specific case, it indeed is a serious and reliable sort of testing.

Firstly, all important parts of the Macek program are equipped with numerous low-level internal self-checks. (By the way, such a careful programming design has proved very successful in catching bugs during program development.) Secondly, interested user may instruct Macek to produce verbose debugging messages, and so to follow all steps of the computation. Thirdly, we have carried out with Macek many involved self-testing computations, which can be generally described by the following two schemes.

– Let us choose a field $\mathbb{F} = GF(q)$, and an unlabeled represented matroid $Q \in \mathcal{U}_3$. Generate (Algorithm 3.2), say for $k = 3$, the list $\mathcal{K}$ of all non-equivalent k-step extensions of Q over $\mathbb{F}$.
– Select from $\mathcal{K}$ the representatives of isomorphism classes, and generate all possible labeled representations of them, producing a list $\mathcal{L}_1$. Then make a list $\mathcal{L}_2$ by selecting representatives of all distinct unlabeled represented matroids from $\mathcal{L}_1$, which contain Q as a represented minor.
– Verify that $\mathcal{K} = \mathcal{L}_2$.

We propose to the reader that, for the practically observed positive outcomes of the above described procedure, the only imaginable good reason is that the list $\mathcal{K}$ really contains all non-equivalent representations of the generated matroid extensions. Moreover, the next test scheme verifies in a nontrivial way that very likely no possible (3C-reducible) abstract extension of a matroid is omitted with the Macek generation routine.

– Let us choose distinct fields $\mathbb{F} = GF(q)$ and $\mathbb{F}' = GF(q')$, and an unlabeled represented matroid $Q \in \mathcal{U}_3$ which is uniquely representable over both $\mathbb{F}, \mathbb{F}'$. (Such as, say, a regular matroid Q.) Generate, say for $k = 3$, the list $\mathcal{K}_1$ of all non-equivalent k-step extensions of Q over $\mathbb{F}$, and the list $\mathcal{K}'_1$ of all k-step extensions of Q over $\mathbb{F}'$.
– Create a list $\mathcal{K}_2$ by selecting from $\mathcal{K}_1$ all matroids representable over the other field $\mathbb{F}'$, and then by selecting the representatives of isomorphism classes. Analogously produce $\mathcal{K}'_2$ from $\mathcal{K}'_1$.
– Verify that $\mathcal{K}_2 = \mathcal{K}'_2$.
– Moreover, in some specific cases like, for example $q = 2$ and $q' = 5$, verify that the list $\mathcal{K}_2$ equals the list of all k-step extensions of Q over the regular partial field.

Again, it is hard to imagine any good reason for the practically observed positive outcomes of the above described procedure, other than that all abstract matroid extensions of Q representable over the respective fields $\mathbb{F}, \mathbb{F}'$ are generated by MACEK. Interested users are welcome to install MACEK and run their own testing computations.

Running time analysis

With such a complex generator like Algorithm 3.2, there is likely no hope to provide a thorough theoretical analysis of its running time. (Obviously, the algorithm has an exponential time complexity, at least.) In the appendix we provide a brief sample analysis of real running time of the algorithm implementation in MACEK. As it is usual in this field, we measure average time needed to generate one matroid extension.

Table 3. Running time analysis of matroid extension generation by Algorithm 3.2 in MACEK (in seconds per extension, normalized to 1GHz CPU).

1-step extensions

field \ *elements*	7	8	9	10	11	12	13
$GF(2)$:	0.012	0.016	0.018	0.037	0.070	0.125	0.308
(# samples)	$1 \to 2$	$1 \to 2$	$4 \to 32$	$17 \to 224$	$70 \to 1736$	$155 \to 7771$	$15 \to 875$
$GF(3)$:	0.007	0.011	0.019	0.042	0.067	0.124	?
(# samples)	$6 \to 60$	$22 \to 482$	$120 \to 7400$	$15 \to 1716$	$9 \to 4115$	$3 \to 2597$	
$GF(4)$:	0.007	0.009	0.017	0.029	0.056	?	?
(# samples)	$8 \to 226$	$78 \to 6950$	$8 \to 2444$	$4 \to 3220$	$4 \to 6731$		

2-step extensions

field \ *elements*	6	7	8	9	10	11
$GF(2)$:	0.030	0.031	0.041	0.060	0.112	0.240
(# samples)	$1 \to 2$	$1 \to 6$	$2 \to 29$	$4 \to 210$	$4 \to 595$	$2 \to 1806$
$GF(3)$:	0.021	0.024	0.048	0.067	0.156	?
(# samples)	$1 \to 28$	$3 \to 289$	$4 \to 2473$	$2 \to 7379$	$1 \to 11683$	
$GF(4)$:	0.023	0.032	0.063	0.116	?	?
(# samples)	$2 \to 104$	$2 \to 1189$	$4 \to 6687$	$1 \to 35528$		

Firstly, Table 3 shows average time for generating 1-step and 2-step extensions of small matroids over $GF(2)$, $GF(3)$, and $GF(4)$. Surprisingly, experimental time does not seem to depend much on the size of the field $\mathbb{F}$; it is likely that the greater complexity of computations over larger fields is compensated with much larger numbers of generated extensions. (That has been also verified by a few random experiments over the fields $GF(8)$ and $GF(9)$.) One may roughly say that average time for generating a 1-step extension of an n-element matroid grows as $\Theta(2^n)$ regardless of the underlying field. Results of the 2-step extension experiments are not that conclusive since we have sampled only few small matroids, but they show a similar behavior, with a bit more influence of the field size.

Table 4. Running time analysis of exhaustive matroid generation based on Algorithm 3.2 (Section 4) in Macek (in seconds per matroid, normalized to 1GHz CPU).

field \ *elements*	8	9	10	11	12	13	14
$GF(2)$:	~ 0.0	~ 0.0	0.06	0.35	1.1	3.3	11
(# generated)	2	4	17	70	337	2080	16739
$GF(3)$:	~ 0.0	0.08	0.25	0.95	3.0	?	?
(# generated)	1	120	1045	14116	330470		
$GF(4)$:	0.08	0.30	1.2	3.2	?	?	?
(# generated)	78	1040	26494	1241588			
$GF(5)$:	0.10	0.33	1.5	?	?	?	?
(# generated)	365	9172	505723				
$GF(8)$:	0.10	0.48	?	?	?	?	?
(# generated)	11237	1220128					

Secondly, Table 4 summarizes average time needed to exhaustively generate a small matroid representation in Macek, according to the ideas presented in Section 4 (i.e. a multistep generation in which the canonical tests take importance). The first three lines show a very regular behavior – the average time grows about 3-4 times with each additional element, and it similarly grows also with the size of the three smallest fields. The last two lines exhibit a somehow different behavior, probably related to the existence of many inequivalent representations of matroids over larger fields. Note the difference between the numbers of generated matroids in Table 2, and here in the last two lines of Table 4 where the $GF(q)$-representable matroids for $q = 4$, resp. $q = 4, 5, 7$, are *not* excluded.

Acknowledgments

The author, besides his current grant support, acknowledges generous support from the Victoria University of Wellington and from the New Zealand Marsden Fund during his stay in Wellington in 2000–2002, where development of the Macek program has begun. The author also thanks Geoff Whittle for helpful ideas and comments in early stages of Macek development.

The large-scale computations leading to the enumeration data summarized in Table 2 have been run (in 2003–2004) on the `minos` cluster at the West Bohemia University (The ITI center, supported by the Ministry of Education of the Czech Republic as the project LN00A056).

References

1. C.R. Coullard, *Minors of* 3*-Connected Matroids and Adjoints of Binary Matroids*, Ph.D. thesis, Northwestern University, 1985.
2. J.S. Dharmatilake, *Binary Matroids of Branch-Width* 3, PhD. dissertation, Ohio State University, 1994.
3. J. Geelen, J. Oxley, D. Vertigan, G. Whittle, *Weak Maps and Stabilizers of Classes of Matroids*, Advances in Appl. Math. 21 (1998), 305–341.
4. P. Hliněný, *The* MACEK *Program*, `http://www.mcs.vuw.ac.nz/research/macek`, `http://www.cs.vsb.cz/hlineny/MACEK`, 2002–2004.
5. P. Hliněný, *On the Excluded Minors for Matroids of Branch-Width Three*, Electronic Journal of Combinatorics 9 (2002), #R32.
6. P. Hliněný, *Using a Computer in Matroid Theory Research*, submitted, 2004.
7. B.D. McKay, *Isomorph-free exhaustive generation*, J. Algorithms 26 (1998), 306–324.
8. J.G. Oxley, Matroid Theory, Oxford University Press 1992.
9. R. Pendavingh, personal communication, 2004.
10. C. Semple, G.P. Whittle, *On Representable Matroids Having Neither* $U_{2,5}$*- nor* $U_{3,5}$*-minors*, In: Matroid Theory, Contemporary Math. 197 (Amer. Math. Soc. 1995), 377–386.
11. C. Semple, G.P. Whittle, *Partial Fields and Matroid Representation*, Advances in Appl. Math. 17 (1996), 184–208.
12. P.D. Seymour, *Decomposition of Regular Matroids*, J. Combin. Theory Ser. B 28 (1980), 305–359.
13. W.T. Tutte, *A Homotopy Theorem for Matroids*, Trans. Amer. Math. Soc. 88 (1958), 144–174.
14. W.T. Tutte, *Connectivity in Matroids*, Cand. J. Math. 18 (1966), 1301–1324.
15. X. Zhou, *Some Excluded Minor Theorems for Binary Matroids*, PhD. Thesis, The Ohio State University (2003).
16. X. Zhou, *On Internally* 4*-connected Non-regular Binary Matroids*, J. Combin. Theory Ser. B to appear (2004).