Petr Hliněný

MACEK:
Practical computations with represented matroids

Dept. of Computer Science, FEI VŠB – TU Ostrava
17. listopadu 15, 708 33 Ostrava
and
Faculty of Informatics, Masaryk University in Brno
Botanická 68a, 602 00 Brno
Czech Republic

e-mail: hlineny@fi.muni.cz
http://www.fi.muni.cz/~hlineny

MACEK is now supported by Czech research grant GAČR 201/05/0050.
1 Matroids and MACEK

Question: What really are matroids?

- A common combinatorial generalization of graphs and finite geometries.
- A new look at (some) structural graph properties.

Question: What can matroids bring to us?

- Interesting objects to study (and difficult, indeed!).
- More general view some concepts brings interesting applications (e.g. the greedy algorithm, or recently the graph rank-width).
1.1 Definitions

A **matroid** $M$ on $E$ is a set system $B \subseteq 2^E$ of **bases**, satisf. the **exch. axiom**

$$\forall B_1, B_2 \in B \ a \ \forall x \in B_1 - B_2, \ \exists y \in B_2 - B_1 : (B_1 - \{x\}) \cup \{y\} \in B.$$ 

The subsets of bases are called **independent**.

**Matroids coming from graphs and from vectors**

**Cycle matroid of a graph** $M(G)$ – on the **edges** of $G$, where acyclic sets are independent.

**Vector matroid of a matrix** $M(A)$ – on the (column) **vectors** of $A$, with usual linear independence.

$$K_4$$

$$M(K_4)$$

$M(A)$
Matrix representation $A$ of a matroid $M$ – the vector matroid

- Elements of $M$ are vectors over $\mathbb{F}$ – the columns of a matrix
  \[ A \in \mathbb{F}^{r \times n}. \]
- Matroid independence is usual linear independence.
- **Equivalence** of representations $\simeq$ row operations on matrices.

Not all matroids have matrix representation over chosen $\mathbb{F}$, some even over no $\mathbb{F}$ at all.

An example – a matrix representation of a rank-3 matroid with 8 elements over $GF(3)$:
1.2 Representing matroids in MACEK

**Matrix representation** $A' = [I | A] \rightarrow$ the *reduced representation* $A$

(stripping the unit submatrix).

$$
\begin{pmatrix}
1 & 0 & 0 & 1 & 2 & 0 & 0 & 1 \\
0 & 1 & 0 & 2 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 2 & 1 & 2
\end{pmatrix}
\rightarrow
\begin{pmatrix}
1 & 2 & 0 & 0 & 1 \\
2 & 0 & 1 & 1 & 1 \\
0 & 1 & 2 & 1 & 2
\end{pmatrix}
$$

- Now matroid elements label both the columns and rows of $A$.
- The rows *display* a basis of $M(A)$.
- *Pivoting* changes to other bases. . .
- (Matrix equivalence now means a sequence of *pivots* and non-zero *scalings*.)

Normally, matrix representations in MACEK are **unlabeled**!
(Though some default labels are printed out for readability. . .)
1.3 Computing matroid properties

- Printing out thorough information about matroids:
  bases, flats, separations, connectivity, girth, automorphism group, representability over other fields, etc.

- Testing matroid properties (including batch-processing):
  minors, isomorphism, connectivity, representability, branch-width, etc.

- Some operations over a matroid:
  deletions/contractions of elements, pivoting, generating other representations of the same matroid, etc.

- A command-line user interface, very suitable for batch-jobs.

- Matroid generation...
2 Exhaustive Generation

A simple approach to combinatorial generation:

- **Exhaustively** construct all possible “presentations” of the objects.
- Then select one **representative** of each isomorphism class by means of an isomorphism tester.
- **Slow**, and problems with ineq. repres. giving different extensions...

The “*canonical construction path*” technique [McKay]:

- Select a small *base* object.
- Then, out of all ways how to construct our big object by single-element steps from the base object (*construction paths*), define the lexicographically smallest one (the *canonical* construction path).
- During generation, throw immediately away non-canonical extensions at each step.
- A big advantage – no explicit pairwise-isomorphism tests are necessary!
2.1 Canonically Generating Matroids

Actually, generating inequivalent matrix representations...

Matrix representation $A' = [I | A] \rightarrow$ reduced representation $A$
(stripping the unit submatrix).

- Base object $\sim$ a submatrix (minor),
- construction path $\sim$ an elimination sequence
  — in reverse order, stripping the excess rows and columns one by one,
- canonical ordering $\sim$ lexic. order on the excess vectors after unit-scaling,
- in a picture:

$$A = \begin{bmatrix} A_0 & u_1 & u_2 & u_4 \\ u_3 & & & \end{bmatrix}$$

$\rightarrow$ an elimination sequence $S = (A_0, A, (1101)_2)$. 

Petr Hliněný, Axiom Workshop RISC 06
Algorithm 2.1.  Recursive generation of (up to) \( \ell \)-step extensions of the matroid generated by a matrix \( A_0 \) over \( \mathbb{F} \).

\[ S_0 = (A_0, A_0, \emptyset) \]
matroid-generate\((S_0)\);

procedure matroid-generate\((S = (A_0, A, q))\)
\[ \text{output the matroid generated by } A; \]
if \( \text{length}(S) \geq \ell \) then exit;

\[ s_0 = \text{number of rows of } A; \quad s_1 = \text{number of columns of } A; \]
for \( x \in \{0, 1\} \), and \( \vec{z} \in \mathbb{F}^{s_x} \) do
\[ q_1 = (q, x); \]
\[ A_1 = A \text{ with added } \vec{z} \text{ as the last row (} x = 0 \text{) or column (} x = 1 \text{);} \]
\[ S_1 = (A_0, A_1, q_1); \]
if \( \neg \text{unit-check}(S_1) \) then continue;
if \( \neg \text{sequence-check}(S_1) \) then continue;
if \( \neg \text{structure-check}(S_1) \) then continue;
if \( \neg \text{canonical-check}(S_1) \) then continue;
matroid-generate\((S_1)\);

end.

• unit-check: unit-scaling of the vectors.

• sequence-check: user-specified, like connectivity, etc.

• structure-check: user-specified, inherited to all minors.

• canonical-check:

Algorithm 2.2.  Testing canonical elimination sequence $S$ with base $A_0$.

procedure canonical-check($S = (A_0, A, q)$)

for $q' \leq$ lexicographically $q$, and all $A'$ equivalent to $A$
such that $A_0$ is a top-left submatrix of $A'$
do
$k = \text{length}(S); \ S' = (A_0, A', q');$
$S'_i = \text{the } i\text{-th step subsequence of } S', \ i = 1, 2, \ldots, k;$
if $\neg \text{unit-check}(S'_i), \ i = 1, \ldots, k$ then continue;
if $\neg \text{sequence-check}(S'_i), \ i = 1, \ldots, k$ then continue;
if $q' < \text{lexicographically } q,$ or
\[ (\vec{u}'_1, \ldots, \vec{u}'_k) \text{ of } S' < \text{lex. } (\vec{u}_1, \ldots, \vec{u}_k) \text{ of } S \]
return false;
done
return true;
end.
2.2 Using Generation in MACEK

- Generating all inequivalent (multi-step) extensions of a given matroid over a fixed finite field. (Easy to split for independent parallel generation.)
- Generation can internally maintain additional structural properties (simplicity, 3-connectivity, excluded minors, etc).
- More tools are provided for involved filtering of generated extensions.

How can MACEK help in research?

- Some computer-assisted proofs
  (e.g. [P. Hliněný, *On the Excluded Minors for Matroids of Branch-Width Three*, Electronic Journal of Combinatorics 9 (2002), #R32.])
- And a very easy generation of nasty counterexamples...
- Say, want to check whether $R_{10}$ is a splitter for the class of near-regular matroids? (Piece of cake...)

Petr Hliněný, Axiom Workshop RISC 06

MACEK: Practical computations with represented matroids
3 Matroid Enumeration Results

Enumeration of *binary combinatorial geometries* (i.e. simple binary matroids).

- Acketa [1984], by hand.
- Kingan, Kingan, Myrvold [2003], using computer and Oid.

<table>
<thead>
<tr>
<th>rank\el.</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>4</td>
<td>8</td>
<td>15</td>
<td>29</td>
<td>46</td>
<td>64</td>
<td>89</td>
<td>*112</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>1</td>
<td>5</td>
<td>14</td>
<td>38</td>
<td>105</td>
<td>273</td>
<td>*700</td>
<td>*1794</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>1</td>
<td>6</td>
<td>22</td>
<td>80</td>
<td>312</td>
<td>*1285</td>
<td>*5632</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>7</td>
<td>32</td>
<td>151</td>
<td>*821</td>
<td>*5098</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>8</td>
<td>44</td>
<td>266</td>
<td>*1948</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>9</td>
<td>59</td>
<td>*440</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>10</td>
<td>*76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
The numbers of labeled / unlabeled represented matroids over small fields.

- The unlabeled case not studied so far to our knowledge.
- A really simple task for MACEK!

<table>
<thead>
<tr>
<th>repr. \ matroid</th>
<th>$U_{2,4}$</th>
<th>$U_{2,5}$</th>
<th>$U_{2,6}$</th>
<th>$U_{3,6}$</th>
<th>$W^3$</th>
<th>$U_{2,7}$</th>
<th>$U_{3,7}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$GF(5)$</td>
<td>3/1</td>
<td>6/1</td>
<td>6/1</td>
<td>6/1</td>
<td>3/2</td>
<td>0/0</td>
<td>0/0</td>
</tr>
<tr>
<td>$GF(7)$</td>
<td>5/2</td>
<td>20/1</td>
<td>60/1</td>
<td>140/3</td>
<td>5/3</td>
<td>120/1</td>
<td>120/1</td>
</tr>
<tr>
<td>$GF(8)$</td>
<td>6/1</td>
<td>30/1</td>
<td>120/1</td>
<td>390/5</td>
<td>6/3</td>
<td>360/1</td>
<td>1200/2</td>
</tr>
<tr>
<td>$GF(9)$</td>
<td>7/2</td>
<td>42/2</td>
<td>210/2</td>
<td>882/7</td>
<td>7/4</td>
<td>840/1</td>
<td>6120/4</td>
</tr>
</tbody>
</table>
The numbers of small 3-connected matroids representable over small fields (generated all as unlabeled represented matroids).

- Computed [2003–4] with MACEK, but no such independent results exist to compare with (to our knowledge).

<table>
<thead>
<tr>
<th>represent. \ elem.</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>regular:</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>7</td>
<td>10</td>
<td>33</td>
<td>84</td>
<td>260</td>
<td>908</td>
</tr>
<tr>
<td>$GF(2)$, non-reg:</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>17</td>
<td>70</td>
<td>337</td>
<td>2080</td>
<td>16739</td>
<td>181834</td>
</tr>
<tr>
<td>$GF(3)$, non-reg:</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>23</td>
<td>120</td>
<td>1045</td>
<td>14116</td>
<td>330470</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

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

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

Want to try? Go to

http://www.cs.vsb.cz/hlineny/MACEK,

read the manual and find out whether MACEK is useful for you...
(Now with a new online interface - TRY IT yourself easily!)

What about correctness?

– Theoretical correctness of MACEK’s algorithms.
– Debugging self-tests implemented in the program code.
– Some highly nontrivial self-reducing computations for comparism.

Anyway,

what is “MACEK” in Czech?