A counterexample to Las Vergnas' strong map conjecture on realizable oriented matroids: supplementary material

Pei Wu
Department of Mathematics,
Simon Fraser University,
8888 University Drive,
Burnaby BC V5A 1S6,
Canada
Tel.: +1236-999-5439
Fax: +1778-782-4947
email: wupei5060309031@gmail.com
Present address:
E3 Building, Central Road,
Airport Economic Zone,
Dongli District,
Tianjin,
China

April 19, 2019

Abstract

The file explains the code *om6.cpp*, which implements the brute-force seach algorithm of enumerating all uniform oriented matroid $\mathcal{M}'$ with $\mathcal{M}_1 \to \mathcal{M}' \to \mathcal{M}_2$. In which $\mathcal{M}_1$ being rank 4 alternating oriented matroid on [6] and $\mathcal{M}_2$ being the rank 2 oriented matroid w.r.t. linear ordering $2 < 1 < 4 < 3 < 6 < 5$.

1 Usage

The file *om6.cpp* is a C++ program of brute-force searching of oriented matroid $\mathcal{M}'$, with $\mathcal{M}_1 \to \mathcal{M}' \to \mathcal{M}_2$. To compile it, use standard C++ compiler supporting C++11. For example, in Linux:

```
g++ om6.cpp -o om6
```

to execute, use command:

```
./om6
```

The output should be identical to the file *om6.txt*.

2 Searching of $\mathcal{M}'$

In this program, signed vectors are encoded by vectors with components ± 1, set of singed vectors encodes by double vectors. And most functions are "pass by reference".

For enumerate all possible $\mathcal{M}'$ we will decide topes $\mathcal{T}(\mathcal{M}')$, which should satisfy $\#\mathcal{T}(\mathcal{M}') = 16$ and $\mathcal{T}(\mathcal{M}_2) \subset \mathcal{T}(\mathcal{M}') \subset \mathcal{T}(\mathcal{M}_1)$, note that $\#\mathcal{T}(\mathcal{M}_1) = 26$ and $\#\mathcal{T}(\mathcal{M}_2) = 6$, so there are $\binom{26-6}{16-6} = 184756$ cases to check. Another constraint is, for every $Q \in \binom{[6]}{4}$, there exist a singed vector c_Q supported on Q such that $c_Q \perp T$ for every $T \in \mathcal{T}(\mathcal{M}')$. For convenience, we consider only those signed vectors with first non-zero component positive and ignore the "all-positive" signed vector $\mathbf{1}$.

Firstly we decide $\mathcal{T}(\mathcal{M}_1)$, which is the set of signed vectors with the form $(+\cdots+-\cdots-)$, $(+\cdots+-\cdots-+\cdots+)$ or $(+\cdots+-\cdots-+\cdots+-\cdots-)$, and $\mathcal{T}(\mathcal{M}_2)$ is a subset of $\mathcal{T}(\mathcal{M}_1)$. The function BuildTopeInd(N, M1, not_M2, M2) takes input N ($N = 6$ in our case), and saving topes in $M1$ as double vectors, $M2$ stores the indexes of topes of $\mathcal{M}_1$ which also topes of $\mathcal{M}_2$, not_M2 is complement of $M2$. The latter two are both vectors.

Then we consider the second constraint, for our purpose we will construct the set of "forbidden quadruples". Consider any four element in [6], say let $Q = \{1, 2, 3, 4\}$. Firstly we can rule out 3 possibilities of c_Q. For example $(+-++++) \in \mathcal{T}(\mathcal{M}_2) \subset \mathcal{T}(\mathcal{M}')$, so $c_Q \neq (+-++)$, similarly $c_Q \neq (++--), (++-+)$, which leave out only four choice of c_Q, i.e., $(+++-)$, $(+-+-)$, $(+--+)$, $(+---)$. So if $T_1|_Q = (+++-), \ldots, T_4|_Q = (+---)$, then T_1, T_2, T_3, T_4 cannot be topes of $\mathcal{M}'$ simultaneously. We call those "forbidden quadruple". The ConstructForbidden(M1, forbidden) construct the set of forbidden quadruple and store the hash values of quadruples.

Finally we enumerate all set $\mathcal{T}$ with $\#\mathcal{T} = 16$ and $\mathcal{T}(\mathcal{M}_2) \subset \mathcal{T} \subset \mathcal{T}(\mathcal{M}_1)$, and check whether it contains forbidden quadruple. The results were output in the following format: each row represents one such set and each component is the index in $\mathcal{T}(\mathcal{M}_1)$.