Advanced Graph Algorithms Jan-Apr 2014

Lecture 20 — 24 MARCH, 2014

Lecturer: Saket Saurabh *Scribe: Aditya Potukuchi*

1 Overview

In the last lecture we saw an algorithm that returns whether or not a linear matroid M has a parity basis and runs in time $\mathcal{O}(m^\omega)$.

In this lecture we will improve that algorithm by running the *Buildparity* algorithm from the previous lecture on smaller blocks which are easy to compute. Also, we will finally reduce finding the maximum parity problem to parity basis problem in time. Note that even the reduction must be efficient.

2 An $mr^{\omega-1}$ algorithm for Matroid Parity

2.1 Preliminary Ideas

We shall make use of *Schur's formula* to compute the inverse of a matrix

Theorem 1 (Schur's formula)**.** *For a matrix*

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

If A is non-singular, $det(M) = det(A)det(S)$, where $S = D - CA^{-1}B$. Further, if S is also non-singular, then

$$M^{-1} = \begin{bmatrix} A^{-1} + A^{-1}BS^{-1}CA^{-1} & -A^{-1}BS^{-1} \\ -S^{-1}CA^{-1} & S^{-1} \end{bmatrix}$$

The matrix S is called the Schur's Complement of A

We will continue working with the matrix

$$Z = \begin{bmatrix} 0 & M \\ -M^T & T \end{bmatrix}$$

Let Y be the Schur's complement ot T, $Y = MT^{-1}M^T$. Thus we may write

$$Z^{-1} = \begin{bmatrix} Y^{-1} & -Y^{-1}MT^{-1} \\ T^{-1}M^TY^{-1} & T^{-1} - T^{-1}M^TY^{-1}MT^{-1} \end{bmatrix}$$

Lemma 2. *Y^{-1} can be computed in $\mathcal{O}(mr^{\omega-1})$ time*

Proof. First, we show that T^{-1} can be computed in r^2 time. This can be done by observing that T is a skew symmetric matrix in the following form:

$$T_{i,j} = \begin{cases} t_k & \text{if } (i,j) = (2k-1, 2k) \text{ for some integer } k \\ -t_k & \text{if } (i,j) = (2k, 2k-1) \text{ for some integer } k \\ 0 & \text{otherwise} \end{cases}$$

Hence, T^{-1} may be written as follows

$$T^{-1}_{i,j} = \begin{cases} -1/t_k & \text{if } (i,j) = (2k-1, 2k) \text{ for some integer } k \\ 1/t_k & \text{if } (i,j) = (2k, 2k-1) \text{ for some integer } k \\ 0 & \text{otherwise} \end{cases}$$

Using this, MT^{-1} can also be computed in $\mathcal{O}(mr)$ time by evaluating each term as follows:

$$T^{-1}M^T_{i,j} = \begin{cases} -m_{i,j-1}/t_k & \text{if } j = 2k \text{ for some integer } k \\ m_{i,j+1}/t_k & \text{if } j = 2k-1 \text{ for some integer } k \\ 0 & \text{otherwise} \end{cases}$$

This gives a $m \times r$ matrix, which can be split into one roe containing m/r $r \times r$ blocks. Similatly, M^T can be written as one column of m/r $r \times r$ blocks as shown.

$$T^{-1} = \begin{pmatrix} 1 & & \\ & 2 & \\ & & \ddots \end{pmatrix}^{\frac{m}{r}}_{\frac{m}{r}}_{\vdots} \quad M^T = \begin{pmatrix} \overset{\frac{m}{r}}{1} & \overset{\frac{m}{r}}{2} & \overset{\cdots}{\ldots} \end{pmatrix}$$

The resultant multiplication requires m/r matric multiplications, each taking time $\mathcal{O}(r^\omega)$. Therefore, Y can be computed in $\mathcal{O}(mr^{\omega-1})$ time, and since it is an $r \times r$ matrix, its inverse can be computed in $\mathcal{O}(r^\omega)$.

□

And with the following lemma, we arrive at the algorithm which saves time by dividing the matrix into smaller blocks whose inverses can be found easier.

Lemma 3. *Given Y^{-1}, for $A, B \subseteq S$ such that $|A|, |B| \leq r$, $Z^{-1}_{A,B}$ can be computed in time $\mathcal{O}(r^\omega)$*

Proof. For any $A, B \subseteq S$, we may write

$$Z^{-1}_{A,B} = T^{-1}_{A,B} - (T^{-1}M^T)_{A,*}Y^{-1}(MT^{-1})_{*,B}$$

. $(T^{-1}M^T)_{A,*}$ and $(MT^{-1})_{*,B}$ are $r \times r$ matrices and can be computed in $\mathcal{O}(r^2)$ in a similar way as in *Lemma* 2. Hence, we can finish the whole computation in $\mathcal{O}(r^\omega)$.

□

With this, we are ready to present the modified algorithm.

3 The Modified Algorithm

MatroidParity(M)

Construct Z and substitute t_is with numbers randomly chosen from a large enough field
$Y \leftarrow MT^{-1}M^T$
$J \leftarrow \emptyset$;
Partition S into two m/r equal size sets
for $1 \leq i \leq m/r$ **do**
  $N \leftarrow Z(J)^{-1}_{S_\omega,S_\omega}$
  $J_1 \leftarrow$ BuildParity(S_1,N,J)
  $J \leftarrow J \cup J_1$)
end
return J;

Algorithm 1: An $mr^{\omega-1}$ algorithm for checking if a matroid has a parity basis

3.1 Running Time and Correctness

In order to prove both, running time and correctness, it is enough to show how the matrix $Z(J)^{-1}_{S_\omega,S_\omega}$ can be computed in $\mathcal{O}(r^\omega)$ time. This is easy to do by using the following result (Sherman-Morrison-Woodbury).

$$Z(J)^{-1}_{S_\omega,S_\omega} = Z^{-1}_{S_\omega,S_\omega} + Z^{-1}_{S_\omega,J}(I - Z_{J,J}Z^{-1}_{J,J})^{-1}Z_{J,J}Z^{-1}_{J,S_\omega}$$

Which can be computed in $\mathcal{O}(r^\omega)$ time since each of the matrices individually can be computed in $\mathcal{O}(r^\omega)$ time from *Lemma 2*.

The running time follows from this because it is doing m/r iterations each one taking time $\mathcal{O}(r^\omega)$. The whole algorithm takes $\mathcal{O}(mr^{\omega-1})$.

4 Reducing Matroid Parity to Parity Basis

Recall that in *Lecture 17*, we defined a term Y for a matroid M with a pairing $/b_i, c_i/_{1\leq i\leq m}$ as follows

$$Y = \sum_{i=1}^{m} x_i(b_i \wedge c_i)$$

Claim 4. $Y = MTM^T$

This can be easily verified term by term by using TM^T from *Lemma 2*.

Also recal that we proved that the size of parity of M is equal to the rank of Y. We also computed it in (mr^2) time. But given *Claim 4*, we can compute it in only $\mathcal{O}(mr^{\omega-1})$ (*Lemma 2*).

Let $k = rank(Y)$. There exists a $P \subseteq \lfloor r \rfloor$ such that $Y_{P,P}$ is full rank, so the linear matroid M, restricted to rows from P has a parity basis (*Lecture 17*). This completes the reduction of Matroid Parity to Parity Basis in time $\mathcal{O}(mr^{\omega-1})$ as the columns forming a parity basis in $M_{P,*}$ form a maximum parity in M.

References

[1] Ho Yee Cheung, *Algebraic algorithms in combinatorial optimization*,Master Thesis,The Chinese University of Hong Kong, 2011.