SUPPLEMENTARY MATERIAL

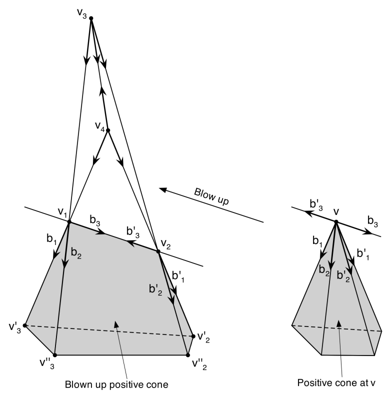

Figure 6: A singular (non-generic) vertex v is blown-up into a virtual polytope with vertices v_1, v_2, v_3, v_4. Notice the base at each virtual vertex contains only 2 directions along the actual positive sign cone.

We have so far restricted our attention to matrices A that are TUM. This was useful to guarantee that Q_{12} has entries in $\{-1, 0. +1\}$ and that the pivoting operation actually works to pass from one vertex to the next. However, as a corollary to lemma **??** we get that the difference between two neighbor vertices is an integer, therefore a multiple of a column of Q_2. Since by varying y we can get all possible positive vertices (since A_1 is unimodular), we get that Q_2 has the sam form before and after pivoting:

Corollary 0.1. *If A is unimodular, $Q'_2 = Pivot(Q_2, i, j)$ has the same form that Q_2, that is, $Q'_2 = \begin{pmatrix} Q'_{12} \\ I_{n-m} \end{pmatrix}$.*

This means that our algorithms work for A unimodular as well.

The following proposition is rather obvious and its proof is left to the reader.

Proposition 0.2. *If A is totally unimodular, then the matrix A' obtained from A by any one of the following operations is still totally unimodular.*

1. *Permuting rows and columns.*
2. *Removing a row or column from A.*
3. *Adding to A one more row or column containing only 0's except one 1.*
4. *Adding to A one more row or column already in A.*
5. *Multiplying a row or column by -1.*

Following Schrijver (1998) we call a $m \times m$ submatrix of a full rank integer $m \times n$ matrix, $n \geq m$, a basis if it has full rank.

Theorem 0.3 (Theorem 19.5 in Schrijver (1998))**.** *Let A be an integral matrix. The following two assertions are equivalent.*

1. *For every basis A_1 of A, the matrix $A_1^{-1}A$ is integral*
2. *For every basis A_1 of A, the matrix $A_1^{-1}A$ is totally unimodular*

Lemma 2.2 becomes a corollary of this theorem.

Proof. (of Lemma 2.2) A is totally unimodular, therefore every basis A_1 is unimodular, therefore A_1^{-1} is integral and so is $A_1^{-1}A = (I, A_1^{-1}A_2)$. Therefore $(I, A_1^{-1}A_2)$ is totally unimodular. Therefore, $Q_{12} = -A_1^{-1}A_2$ is totally unimodular (a consequence of prop 0.2). Therefore $Q_2 = \binom{Q_{12}}{I}$ is totally unimodular. ☐

Proof. (of Proposition 2.3) The pivoting operation of algorithm 1 is easily seen to be a combination of operations from proposition 0.2 and the pivoting operation defined in Schrijver (1998) by:

$$\begin{pmatrix} \epsilon & c \\ b & D \end{pmatrix} \rightarrow \begin{pmatrix} \epsilon & -\epsilon c \\ \epsilon b & D - \epsilon bc \end{pmatrix}$$

The latter operation is proved in Schrijver (1998) to preserve TUM. ☐

Proof. (of Lemma 2.4) Any solution has the form $\nu^{(0)} + Q_2 w$ with w a non-negative $n - m$ vector. Let $J = \{j : Q_2[i, j] \neq 0\}$ and assume $Q_2[i, j] < 0$ for all $j \in J$. Any non-zero scalar $Q_2[i,]w$ would be negative, so $y'[i] + Q_2[i,]w$ would again be negative and so would never be the coordinate of a solution. Therefore the columns b_j, $j \in J$ are unecessary to express any solution in the form $\binom{y'}{0} + Q_2 w$. The solution polytope has dimension $n - m$, and therefore contains at least $n - m + 1$ linearly independent solutions. One solution might be $\binom{y'}{0}$ (if $\nu^{(0)}$ is actually a vertex), but there are at least $n - m$ other linearly independent solutions. However, the remaining columns $Q_2[, k]$, $k \notin J$, being less than $n - m$ in number, cannot express linearly independent $n - m$ solutions; a contradiction. ☐

Proof. (of Proposition 2.6) Both in the queue vq and the one-skeleton data structure sk, vertices are accompanied by their base-matrix (or set of base-matrices if it is a non-generic vertex) and their corresponding permutations. We place our first vertex in the queue, then each vertex v popped out of the queue is appended to the one-skeleton data structure and checked for its neighbors. So vq contains vertices with neighbors unchecked and sk contains vertices with neighbors checked. If a neighbor v' is not already in vq, we check if it is already in sk so far (if it is already in vq, we ignore it and look at the next one). If it is in sk, we update the list N of neighbors of both v and v' in sk. If it is not in sk, then it is a new vertex and we add it to the queue. This should be clear that this exhausts all vertices and their neighboring relations. For the running time, the number of iterations of the while loop on line 3 is equal to the number of vertices v. For each iteration, the dominating operation is the pivot operation (which occurs only when a vertex is appended to the queue). Checking in line 11 and 13 that a vertex neighbor (that is, a vertex candidate) is not in the queue and not in the skeleton can be done in $O(1)$ time with a hash table sufficiently large. $\square$