Counting vertices of integer polytopes defined by facets

Heng Guo[*]
School of Informatics
University of Edinburgh
hguo@inf.ed.ac.uk

Mark Jerrum[†]
School of Mathematical Sciences
Queen Mary, University of London
m.jerrum@qmul.ac.uk

Abstract

We present a number of complexity results concerning the problem of counting vertices of an integral polytope defined by a system of linear inequalities. The focus is on polytopes with small integer vertices, particularly 0/1 polytopes and half-integral polytopes.

1 Introduction

We are interested in counting (or uniformly sampling) vertices of a polytope defined by linear inequalities $Ax \leq b$. In particular we concentrate on the sort of polytopes that arise in the study of computational optimisation problems: integral polytopes (convex hulls of finite subsets of $\mathbb{N}^n$) and especially 0/1 polytopes (convex hulls of subsets of $\{0,1\}^n$). We assume that instances are presented as systems of linear inequalities.

Problem: #Vertices of a 0/1 Polytope.

Instance: An $m \times n$ integer matrix $A \in \mathbb{Z}^{m \times n}$ and a vector $b \in \mathbb{Z}^m$.

Promise: The inequalities $Ax \leq b$ define a 0/1 polytope P.

Output: The number of vertices of P.

It is not clear whether it is possible to decide, in polynomial time, if a given system of linear inequalities defines a 0/1 polytope, or, more generally, an integral polytope. Certainly, it is no easy task. Ding et al. [5] give a proof of co-NP-completeness, but this is for a system of inequalities defining an unbounded polyhedron. In the same paper it is shown, by combining several difficult results, that deciding integrality of the polytope $\{x \in \mathbb{R}^n : Ax \leq b, x \geq \mathbf{0}\}$ can be decided in polynomial time, in the case where A is a 0/1 matrix and $b = \mathbf{1}$. Given the uncertainty surrounding this question, it seems reasonable to add the promise that P is a 0/1 polytope. Note that if we could verify that P is integral in polynomial time, then we could go on verify that it is a 0/1 polytope in polynomial time: we would just need to check that P is contained in the cube $[0,1]^n$, which can be done by linear programming.

In cases of interest, the matrix A and vector b will not be arbitrary, but will have been introduced with a certain goal in mind. The insights that led to their construction can almost certainly be used to provide an elementary proof of integrality of P. Furthermore, except in Section 5, the matrix A will be 'totally unimodular', and hence will necessarily define an integral polytope. Total unimodularity is decidable in polynomial time using Seymour's decomposition theorem for regular matroids [17]: see Truemper [19].

Our first observation is that #VERTICES OF A 0/1 POLYTOPE is hard to solve *exactly*. This follows easily by encoding perfect matchings in bipartite graphs as vertices of a 0/1 polytope. A proof of this easy result can be found at the end of this section.

Proposition 1. *#VERTICES OF A 0/1 POLYTOPE is #**P**-complete.*

In light of this strong negative result, we naturally turn our attention to approximate counting. Before presenting our results, we briefly review the main definitions and concepts used in the study of approximation algorithms for counting problems. The reader is directed to Dyer, Goldberg, Greenhill and Jerrum [6] for precise definitions and a survey of the wider context.

The standard notion of efficient approximation algorithm in the context of counting problems is the *Fully Polynomial Randomised Approximation Scheme*, or FPRAS. This is a randomised algorithm that is required to produce a solution within relative error $1 \pm \varepsilon$, in time polynomial in the instance size and ε^{-1}.[1] The computational complexity of approximate counting problems can be compared through *Approximation-Preserving* (or AP-) *reductions*. These are randomised polynomial-time Turing reductions that preserve (closely enough) the error tolerance. The set of problems that have an FPRAS is closed under AP-reducibility.

Stockmeyer [18] was the first to produce evidence that approximate counting is of lower computational complexity than exact counting. This key insight was refined by Valiant and Vazirani [21, Cor. 3.6], who showed that every function in #**P** can be approximated (in the FPRAS sense) by a polynomial-time Turing machine with an oracle for an **NP**-complete problem. Therefore the strongest negative result we can have for approximate counting is one of **NP**-hardness. An example of such a maximally hard problem is #IS, which asks for the number of independent sets of all sizes in a general graph. It follows that the existence of an FPRAS for #IS would imply **NP** = **RP**. An **NP**-hard problem concerning polytopes is given in the next section.

Many counting problems are known to have an FPRAS and many others to be **NP**-hard to approximate. An interesting empirical observation is that many of the rest are interreducible via approximation-preserving reductions. One member of this equivalence class is #BIS, which asks for the number of independent sets of all sizes in a bipartite graph.

Problem: #BIS.

Instance: A bipartite graph B.

Output: The number of independent sets (of all sizes) in B.

After two decades of fairly intensive study, no FPRAS for #BIS has been discovered, but neither has #BIS been shown to be **NP**-hard to approximate. So showing that a counting problem Π is approximation-preserving interreducible with #BIS can be interpreted as evidence that Π does not admit an FPRAS, though the evidence falls short of a demonstration of **NP**-hardness. An example involving 0/1 polytopes is presented in Section 3.

---

[1]By the standard duplication trick, any polynomial approximation ratio can be amplified into an $(1 \pm \varepsilon)$-approximation in polynomial-time.

Khachiyan [10] showed that approximately counting vertices of general polytopes presented as linear inequalities is **NP**-hard, a result rediscovered by Najt [14]. As we just saw, this is the strongest possible demonstration of intractability. However, the vertices of the polytopes employed in the proof have rational coordinates, which, if rescaled to integers, would be exponential in the dimension of the ambient space. Our aim in this work is to find hard examples in small integers.

Ultimately, we would like to characterise the complexity of computing approximate solutions to #VERTICES OF A 0/1 POLYTOPE. We have not been able to establish the strongest intractability result, which would be a demonstration of **NP**-hardness. The typical approach of showing **NP**-hardness for an approximate counting problem is to reduce from a combinatorial optimisation problem. However, the combinatorial optimisation problem related to a 0/1 polytope is often tractable, which makes showing **NP**-hardness difficult.

Instead, we are able to obtain an **NP**-hardness result by relaxing the allowed problem instances from 0/1 polytopes to 'half-integral' polytopes whose vertices are elements of $\{0, 1, 2\}^n$. (More conventionally, these polytopes are scaled by a factor 2, so as to have vertices in $\{0, \frac{1}{2}, 1\}^n$.) This negative result relates to a natural family of 'perfect 2-matching polytopes' associated with graphs. See Proposition 2.

What can be said about the complexity of #VERTICES OF A 0/1 POLYTOPE itself? The most prominent class of matrices defining integer polytopes are the *totally unimodular matrices*. These are matrices A that have the property that every square submatrix of A has determinant -1, 0 or 1. (In particular, the elements of A take values in $\{-1, 0, 1\}$.) Most 0/1 polytopes arising in combinatorial optimisation arise from totally unimodular matrices. Network matrices and transposes of network matrices are natural subclasses of totally unimodular matrices that will be defined in Section 3. In some sense, these matrix classes are universal in that every totally unimodular matrix can be built from network matrices, transposes of network matrices, and a certain 5×5 matrix.

In Section 3, we show (Theorem 7) that #VERTICES OF A 0/1 POLYTOPE, when restricted to transposes of network matrices, is interreducible with #BIS with respect to approximation-preserving reductions. This locates this special case of the problem as accurately as possible, given the current state of knowledge of the complexity landscape. When restricted to network matrices, #VERTICES OF A 0/1 POLYTOPE appears to be become easier, and we identify a subclass of polytopes (Proposition 9) for which the vertex counting problem is solvable in the FPRAS sense. We leave it as an open question whether there is an FPRAS for #VERTICES OF A 0/1 POLYTOPE when restricted to network matrices more generally.

In the final section we go beyond totally unimodular matrices. Here, there are fewer examples in the literature, but we do note that at least one naturally occurring class of polytopes, arising from 'stable matchings', has been shown to give rise to a vertex counting problem that is interreducible with #BIS. This raises the intriguing possibility that #VERTICES OF A 0/1 POLYTOPE itself is actually equivalent in complexity to #BIS.

The complexity of approximate counting and (almost) uniform sampling are usually closely related, and this is indeed the case here. For simplicity we concentrate throughout on approximate counting, but the corresponding uniform sampling problems have essentially the same complexity. This follows by general considerations [9] from the fact that #VERTICES OF A 0/1 POLYTOPE and all the restrictions of it considered here are self-reducible. We expand on this remark at the end of Section 3.

Mihail and Vazirani [13] conjectured that the simple random walk on the graph of a 0/1 poly-

tope is rapidly mixing, which is still open. Note that this conjecture, if true, does not directly imply an FPRAS for #Vertices of a 0/1 Polytope, because the degree of the vertices can be exponentially large in the dimension of the ambient space. The stationary distribution of the random walk over graphs of 0/1 polytopes can be very different from the uniform distribution.

Although we restrict attention to complexity-theoretic results in this work, it should be noted that several authors have studied heuristic approaches, including Avis and Devroye [1] and Salomone, Vaisman and Kroese [15]. We round off the section with the deferred proof.

Proof of Proposition 1. Membership in #**P** is clear.

To demonstrate hardness, we simply show that #Vertices of a 0/1 Polytope includes counting perfect matchings in a bipartite graph as a special case. Suppose G is a bipartite graph, and denote the vertex and edge sets of G by $V(G)$ and $E(G)$. Introduce variables $\{x_{uv} : uv \in E(G)\}$ in 1-1 correspondence with the m edges of G. Consider the m-dimensional perfect matching polytope $P_{\mathrm{PM}}(G)$ of G, whose vertices are in bijection with the perfect matchings of G. Specifically, for each perfect matching $M \subseteq E(G)$ of G there corresponds a vertex of $P_{\mathrm{PM}}(G)$ given by

$$x_{uv} = \begin{cases} 1, & \text{if } uv \in M; \\ 0, & \text{otherwise,} \end{cases}$$

for all $uv \in E(G)$. In the case that G is bipartite, the polytope $P_{\mathrm{PM}}(G)$ is defined by the following inequalities [16, Thm 18.1]:

$$0 \le x_{uv} \le 1, \quad \text{for all } uv \in E(G), \text{ and}$$
$$\sum_{v \in V(G):uv \in E(G)} x_{uv} = 1, \quad \text{for all } u \in V(G).$$

Thus, counting perfect matchings in a bipartite graph can be reduced to counting vertices of an easily computable and easily described polytope. The result follows from Valiant's classical result that counting perfect matchings in a bipartite graph is #**P**-complete [20]. $\qquad\square$

2 The perfect 2-matching polytope

In this section we see that by going a little beyond 0/1 polytopes we can find counting problems that are **NP**-hard to approximate.

Given a graph G, the *perfect 2-matching polytope* (P2M polytope) is defined by the system of linear inequalities

$$0 \le x_{uv} \le 2, \quad \text{for } uv \in E(G);$$
$$\sum_{v:uv \in E(G)} x_{uv} = 2, \quad \text{for } u \in V(G).$$

Let $Z_{\mathrm{P2M}}(G)$ be the number of vertices of the P2M polytope associated with G. We will show that approximating $Z_{\mathrm{P2M}}(G)$ is **NP**-hard.

Recall that an edge cover of a graph is a set C of edges such that any vertex is incident to some edge $e \in C$. Balinski [2] observed the following characterisation of the vertices of the P2M polytope. (See also Schrijver [16, Cor. 30.2b] together with the observation at the end of [16, §30.3].)

Proposition 2. *The vertices of the P2M polytope correspond to edge covers consisting of a matching M, with $x_e = 2$ for $e \in M$, and vertex-disjoint odd cycles that are also disjoint from M, with $x_e = 1$ for each edge e in any of the odd cycles.*

Theorem 3. *There is no FPRAS for $Z_{\mathrm{P2M}}(G)$ unless $\mathbf{NP} = \mathbf{RP}$.*

Proof. We call edge covers corresponding to the vertices of the P2M polytope, as in Proposition 2, *P2M covers*, and call edge covers of G consisting of vertex-disjoint odd cycles without the matching *odd cycle covers*. We denote the number of odd cycle covers of G by $Z_{\mathrm{OC}}(G)$. By definition, the number of P2M covers is $Z_{\mathrm{P2M}}(G)$.

First we reduce deciding the existence of a Hamiltonian path between two given vertices in a bipartite graph to deciding the existence of an odd cycle cover in a general graph. The former problem is known to be $\mathbf{NP}$-complete [12]. (The given reference treats the Hamilton *cycle* problem, but essentially the same reduction deals with *paths*.) Let G be a bipartite graph with two distinguished vertices s and t. We introduce a new vertex w, and add two edges sw and wt. Call the resulting graph G'. Any odd cycle in G' must include the new vertex w, and so any odd cycle cover of G' must consist of a Hamiltonian path in G from s to t, together with the two edges incident at w. Conversely, any Hamiltonian path in G from s to t can be extended to an odd cycle cover of G' by adding edges sw and wt. Thus, deciding the existence of an odd cycle cover of a graph is $\mathbf{NP}$-complete.

Next we reduce deciding the existence of an odd cycle cover to approximating the number of P2M covers. For the reduction, we use the gadget in Figure 1 to reduce the contribution from the matching (isolated edges). Note that the parameter ℓ will be tuned later.
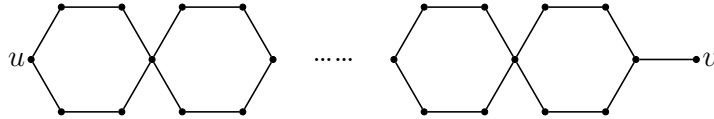

Figure 1: The hexagon gadget, with 2ℓ hexagons in the middle.

Let G be an instance for odd cycle covers. Replace each edge $uv \in E(G)$ with a copy of the gadget, and consider any P2M cover in the resulting graph G'. The possible configurations induced by this cover on the gadget with end points u and v can be partitioned into three types (see Figure 2):

- Type P[ath]. There is a path from u to v together with some isolated edges. Note that there are two choices for the path in each hexagon.

- Type M[atched]. There is no path from u to v, but u and v are both covered (by isolated edges). Note that there are two choices in every other hexagon, starting from the first hexagon (counting from u).

- Type U[nmatched]. There is no path from u to v, and u and v are both uncovered. Note that there are two choices in every other hexagon, starting from the second hexagon (counting from u).

Note that this list is exhaustive.

If there are 2ℓ hexagons in the gadget then there are 4^ℓ configurations of Type P, 2^ℓ of type M and 2^ℓ of type U.
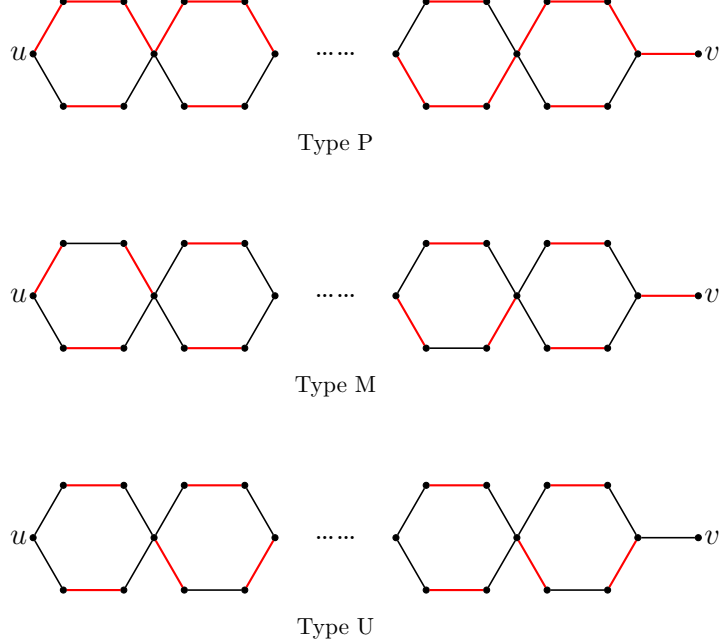
Figure 2: Three types of covers (coloured in red) for the gadget.

Let's return to the P2M cover of G'. The edges in G corresponding to Type P configurations in G' form a collection of disjoint odd cycles in G. (Note that the path in a Type P configuration is of odd length.) Moreover the edges of G corresponding to Type M configurations form a collection of isolated edges, which are disjoint from the odd cycles. Finally, the collection of all odd cycles and isolated edges described above cover all the vertices of G, and hence form a P2M cover of G. Conversely, any P2M cover in G can be lifted to a P2M cover of G' by choosing a Type P configuration in G' for every cycle edge of G, a Type M configuration in G' for every isolated edge in G, and a Type U configuration for every other edge of G. Suppose the P2M cover of G has k cycle edges, and hence $m - k$ other edges, where $m = |E(G)|$. Then the number of P2M covers in G' that correspond to a particular P2M cover in G is $4^{\ell k} 2^{\ell(m-k)} = 2^{\ell m} 2^{\ell k}$. Thus, for sufficiently large ℓ, we expect $\frac{Z_{\mathrm{P2M}}(G')}{2^{\ell m} 2^{\ell n}}$ to be a good approximation to $Z_{\mathrm{OC}}(G)$.

To be more specific, we will choose $\ell \geq m + 2$. The discussion above implies that $Z_{\mathrm{P2M},n}(G') = 2^{\ell m} 2^{\ell n} Z_{\mathrm{OC}}(G)$, where $Z_{\mathrm{P2M},k}(G')$ denotes the number of P2M covers of G' with k edges of the Type P. Note that $Z_{\mathrm{P2M}}(G') = \sum_{k=0}^{n} Z_{\mathrm{P2M},k}(G')$. Moreover, as $Z_{\mathrm{P2M}}(G) \leq 2^m$,

$$\sum_{k=0}^{n-1} Z_{\mathrm{P2M},k}(G') \leq 2^m 2^{\ell m} 2^{\ell(n-1)} \leq \frac{2^{\ell(m+n)}}{4}. \tag{1}$$

Suppose we want to decide whether $Z_{\mathrm{OC}}(G)$ is zero or non-zero. Given an FPRAS for $Z_{\mathrm{P2M}}(G')$, we compute $\widetilde{Z}$ such that $\frac{1}{2} \leq \frac{\widetilde{Z}}{Z_{\mathrm{P2M}}(G')} \leq \frac{3}{2}$ with probability at least $\frac{3}{4}$. There are two cases.

- If $\widetilde{Z} \geq 2^{\ell(m+n)-1}$, then $Z_{\mathrm{P2M}}(G') \geq \frac{2^{\ell(m+n)}}{3} > \frac{2^{\ell(m+n)}}{4}$, which by (1) implies that $Z_{\mathrm{P2M},n}(G') > 0$ and hence $Z_{\mathrm{OC}}(G) \geq 1$.

6

- Otherwise $\widetilde{Z} < 2^{\ell(m+n)-1}$, and thus $Z_{\mathrm{P2M}}(G') < 2 \times 2^{\ell(m+n)-1} = 2^{\ell(m+n)}$. Since $Z_{\mathrm{P2M},n}(G)$ is divisible by $2^{\ell(m+n)}$ it follows that $Z_{\mathrm{P2M},n}(G') = 0$ and $Z_{\mathrm{OC}}(G) = 0$.

With probability at least $\frac{3}{4}$ we correctly decide whether G has an odd-cycle cover. Since we have a polynomial-time randomised algorithm with two-sided error for an **NP**-complete problem, it follows that **NP** $\subseteq$ **BPP**. It is a standard fact that the latter inclusion implies **RP** = **NP**. $\square$

The 'powering' construction (Figure 1) used here is similar to those used in a similar context by Khachiyan [10] and Najt [14], and more generally by Jerrum, Valiant and Vazirani [9]. Note that the proof actually shows a technically stronger result, namely that approximating $Z_{\mathrm{P2M}}(G)$ within (say) a factor 2 is an **NP**-hard problem.

3 Transposes of network matrices

As we noted, one way to specify a 0/1 polytope is by giving a system of linear inequalities $Ax \leq b$ where the matrix A is totally unimodular. Network matrices and their transposes are interesting subclasses of totally unimodular matrices. In some sense, network matrices and their transposes, together with a certain 5×5 matrix, generate all totally unimodular matrices [19]. We begin by defining the class of network matrices.

Suppose we have a directed graph (V, E) and a directed tree (V, T) on the same vertex set. (The orientations of the arcs of the tree are arbitrary, and do not necessarily point towards a specific root vertex. It is convenient to allow parallel arcs in the graph (V, E).) Given an arc $e = uv \in E$ and an arc $t \in T$, let Π be the path from u to v in T. Then define the $|T| \times |E|$ matrix A by

$$A_{te} = \begin{cases} +1, & \text{if } t \text{ occurs in a forward direction in } \Pi; \\ -1, & \text{if } t \text{ occurs in a backward direction in } \Pi; \\ 0, & \text{otherwise.} \end{cases}$$

The matrix A is the *network matrix generated by* (V, E) *and* (V, T). An integer matrix is said to be a network matrix if it is generated in this way.

Proposition 4. *Any network matrix is totally unimodular.*

See [16, Thm 13.20] for a proof. Of course the transpose of a network matrix is also totally unimodular. The following easy lemma will be useful. Amongst other things, it tells us that if we have a system of inequalities defined by network matrix, or the transpose of a network matrix, then we can freely add additional bounds such as $\mathbf{0} \leq x \leq \mathbf{1}$.

Lemma 5. *The property of being a network matrix is preserved under the following operations:*

(a) *duplicating a row or a column;*

(b) *negating a row or a column;*

(c) *extending the matrix with a unit row or column (one which has a 1 in a single location and zeros elsewhere).*

Proof. Let A be a network matrix defined by the directed graph (V, E) and directed tree (V, T). Recall that rows of A correspond to tree arcs $t \in T$ and columns to graph arcs $e \in E$.

(a) To duplicate a row indexed by arc $t = uv \in T$, introduce a new vertex w and set $T :=$ $T \cup \{uw, wv\} \setminus \{uv\}$. To duplicate a column indexed by arc $e \in E$, introduce a new arc e' parallel to e.

(b) To negate a row indexed by t, reverse the direction of arc $t \in T$. To negate a column indexed by e, reverse the direction of arc $e \in E$.

(c) To introduce a new row with a 1 in the column indexed by $e = uv \in E$, introduce a new vertex w and set $E := E \cup \{uw\} \setminus \{uv\}$ and $T := T \cup \{vw\}$. To introduce a new column with a 1 in the row indexed by $t = uv \in T$, introduce a new arc uv to E.

It is easy to check that these actions have the desired effect on the matrix A. □

So now consider a 0/1 polytope P defined by inequalities $Ax \leq b$ where A is the *transpose* of a network matrix. Let us consider the defining equations of the facets of P in terms of (V, E) and (V, T). The arcs in T correspond to variables and those in E to (left-hand sides of) inequalities. Let $uv \in E$ be an arc in the graph. The variables that occur in the corresponding inequality are the ones encountered when tracing out the unique path from u to v in (V, T). The coefficient of a variable is $+1$ or -1 depending on whether the arc in the tree is aligned with or against the direction of the path.

Example 6 (independent sets in a bipartite graph)**.** *Let $B = (U \cup U', F)$ be a bipartite (undirected) graph, where $U = \{u_1, \ldots, u_n\}$ and $U' = \{u'_1, \ldots, u'_m\}$ are the parts of the bipartition of the vertex set. We encode the independent sets in B as vertices of a polytope defined by the transpose of a network matrix. As above, we specify this matrix by giving the graph (V, E) and tree (V, T). Introduce a new vertex r, and let $V = U \cup \{r\} \cup U'$ and $T = \{u_i r : 1 \leq i \leq n\} \cup \{ru'_j : 1 \leq j \leq m\}$. Let the arc set E be obtained from F by simply orienting all edges in F from U to U'. Each arc $u_i r \in T$ (respectively, $ru'_j \in T$) corresponds to a variable x_i (respectively, y_j). Consider the network matrix A defined by (V, E) and (V, T). Each inequality in $A^T(x_1, \ldots, x_n, y_1, \ldots, y_m)^T \leq \mathbf{1}$ is of the form $x_i + y_j \leq 1$ for some $u_i u'_j \in F$. Then we add the inequalities $0 \leq x_i, y_j \leq 1$ for every $i \in [n]$ and $j \in [m]$, which can be done while maintaining the defining matrix to be the transpose of a network matrix, by Lemma 5(c). These inequalities define the independent (or stable) set polytope of the bipartite graph B [16, Thm. 19.7]. This fact also follows easily from total modularity of the system of inequalities, which itself follows from Proposition 4.*

Our goal in this section is to precisely locate the complexity of #VERTICES OF A 0/1 POLYTOPE, when the matrix A is the transpose of a network matrix. For the upper bound (which is the non-trivial direction) we use an approximation-preserving reduction to the following problem.

Problem. #1P1NSAT.

Instance. A CNF Boolean formula φ in which each clause contains at most one positive literal and at most one negative literal.

Output. The number of satisfying assignments of φ.

We know that #1P1NSAT is equivalent to #BIS under polynomial-time approximation-preserving reductions [6, Thm 5].

Theorem 7. *When A is restricted to be the transpose of a network matrix, the problem #VERTICES OF A 0/1 POLYTOPE is equivalent under polynomial-time approximation-preserving reductions to #BIS.*

Proof. We have just seen in Example 6 that #BIS is essentially a special case of #VERTICES OF A 0/1 POLYTOPE, so we just need to describe a polynomial-time approximation-preserving reduction from #VERTICES OF A 0/1 POLYTOPE to #1P1NSAT in the case that A is the transpose of a network matrix. The reduction exploits a construction from Chen, Dyer, Goldberg, Jerrum, Lu, McQuillan and Richerby [4, Lem. 46].

Let the polytope P be an instance of #VERTICES OF A 0/1 POLYTOPE defined by a matrix A and vector b, where A is the transpose of a network matrix. Suppose in turn that A is specified by the directed tree (V, T) and directed graph (V, E) on the common vertex set V. Recall that variables are associated with arcs in T, thus: $\{x_t : t \in T\}$. Choose an arbitrary root $r \in V$ as reference point. The first step is to make a change of variables. Introduce a new set of variables $\{z_v : v \in V\}$, and define $z_r = 0$ and $z_v - z_u = x_t$ for all $t = uv \in T$. Thus

$$z_v = \pm x_{t_1} \pm x_{t_2} \pm \cdots \pm x_{t_\ell},$$

where $(t_1, t_2, \ldots, t_\ell)$ is the path from r to v in the tree, and the sign associated with the ith term is positive if arc t_i is traversed in the forward direction and negative otherwise. Note that the variables $\{x_t : t \in T\}$ determine the variables $\{z_v : v \in V\}$ and vice versa.

Now consider the inequality defined by an arc $e = uv \in E$:

$$\sum_{i=1}^{\ell} \pm x_{t_i} \leq b_e,$$

where $(t_1, t_2, \ldots t_\ell)$ is the path from u to v in T, and the signs are determined by the directions of the arcs. When translated to the new set of variables, this inequality simplifies to $z_v - z_u \leq b_e$. (If w is the lowest common ancestor of u and v in the tree T, then the x-variables corresponding to vertices between r and w appear twice in $z_v - z_u$, but they have opposite signs and hence cancel. Also, the x-variables corresponding to vertices between u and w are negated, reflecting the fact that the path from u to v is being traversed in the direction *towards* the root r.) So it just remains to encode the inequalities

$$0 \leq z_v - z_u \leq 1, \text{ for } uv = e \in T \quad \text{and} \quad z_v - z_u \leq b_e, \text{ for } uv = e \in E \qquad (2)$$

as clauses within an instance φ of #1P1NSAT.

To do this, introduce Boolean variables $\{\zeta_v^i : v \in V \text{ and } -n < i \leq n\}$ and start to build an instance φ on this variable set by introducing clauses

$$\{\zeta_v^{i+1} \Rightarrow \zeta_v^i : v \in V \text{ and } -n < i < n\}.$$

Note that the clause $\zeta_v^{i+1} \Rightarrow \zeta_v^i$ is logically equivalent to $\neg \zeta_v^{i+1} \vee \zeta_v^i$, so has the correct syntactic form. Also note that for each $v \in V$ there are $2n + 1$ consistent assignments to the variables

$\{\zeta_v^i : -n < i \leq n\}$, namely

$$(\zeta_v^{-n+1}, \zeta_v^{-n+2}, \ldots, \zeta_v^{n-1}, \zeta_v^n) = \begin{cases} (0,0,0,\ldots,0,0), \\ (1,0,0,\ldots,0,0), \\ (1,1,0,\ldots,0,0), \\ \quad\vdots \\ (1,1,1,\ldots,1,0), \\ (1,1,1,\ldots,1,1), \end{cases}$$

where we associate false with 0 and true with 1. We use these Boolean assignments to encode integer assignments in the range $\{-n, \ldots, +n\}$ to z_v via the correspondence

$$\zeta_v^i = 1 \iff i \leq z_v, \quad \text{for all } -n < i \leq n.$$

Note that, by construction, for all $v \in V$,

$$|z_v| \leq (\text{length of the path in } T \text{ from } r \text{ to } v) \leq n,$$

so our encoding covers the feasible range of z_v.

It only remains to encode the z-inequalities (2), together with $z_r = 0$, by adding extra clauses to φ. Note that every z-inequality is of the form $z_v - z_u \leq c$, which is equivalent to the collection of clauses $\{\zeta_v^{i+c} \Rightarrow \zeta_u^i : -n < i, i+c \leq n\}$. Finally, the equality $z_r = 0$ is equivalent to the conjunction of clauses

$$\zeta_r^{-n+1}, \ \zeta_r^{-n+2}, \ \ldots, \ \zeta_r^{-1}, \ \zeta_r^0, \ \neg\zeta_r^1, \ \neg\zeta_r^2 \ \ldots, \ \neg\zeta_r^{n-1}, \ \neg\zeta_r^n.$$

This completes the construction of the instance φ of #1P1NSAT. We see that the number of satisfying assignments to φ is equal to the number of feasible solutions to the z-inequalities, which in turn is equal to the number of vertices of the polytope P. The reduction is parsimonious (i.e., preserves the number of solutions) and hence is certainly approximation preserving. $\qquad\square$

We finish the section by noting that counting problems associated with 0/1 polytopes are self-reducible, in the sense that the set of vertices of a 0/1 polytope P can be expressed as the union of the vertices of two lower-dimensional polytopes P_0 and P_1 obtained by intersection with planes of the form $x_n = 0$ and $x_n = 1$. For all the counting problems considered here, the polytopes P_0 and P_1 come from the same class of polytopes as P. This is an easy observation for general 0/1 polytopes, and follows from Lemma 5 for polytopes defined by (transposes of) network matrices. For self-reducible problems, approximate counting and (almost) uniform sampling are related by polynomial-time reductions, as observed by Jerrum, Valiant and Vazirani [9].

4 Network matrices

Now consider the defining equations $Ax \leq b$ of a polytope P when A is a network matrix. Relative to the previous section, the roles of variables and equations are reversed. Variables now correspond to arcs in E and equations to arcs in T. Fix a tree arc $t \in T$ and define

$$F_t^+ = \{e = uv \in E : \text{the path from } u \text{ to } v \text{ passes through } t \text{ is the forward direction}\},$$
$$F_t^- = \{e = uv \in E : \text{the path from } u \text{ to } v \text{ passes through } t \text{ is the backward direction}\},$$

where the paths in question are paths in the tree (V, T). Then the equation defined by the arc t is

$$L_t(x) = \sum_{e \in F_t^+} x_e - \sum_{e \in F_t^-} x_e \le b_t.$$

We are interested in the polytope defined by the system $\{L_t \le b_t : t \in T\}$.

Example 8 (The matching polytope for a bipartite graph)**.** *Given a bipartite graph $B = (U \cup U', F)$ where $|U| = |U'| = n$, we set up the same directed graph (V, E) and tree (V, T) as in Example 6 for #BIS. For each edge $u_i u'_j \in F$ of B we introduce a variable x_{ij} and associate it with the arc $u_i u'_j \in E$.*

Now, the inequality associated with arc $u_i r \in T$ (respectively, $r u'_j \in T$) of the directed tree has the form $\sum_{j:ij \in F} x_{ij} \le c$ (respectively, $\sum_{i:ij \in F} x_{ij} \le c$). The defining inequalities of the matching polytope of B are obtained by setting $c = 1$ for all edges $u_i u_j \in F$, and adding the inequalities $x_{ij} \ge 0$ for all $i, j \in [n]$. To obtain the perfect matchings polytope, we simply include an inequality $\sum_{j:ij \in F} x_{ij} \ge 1$ (respectively $\sum_{i:ij \in F} x_{ij} \ge 1$) complementary to each inequality $\sum_{j:ij \in F} x_{ij} \le 1$ (respectively $\sum_{i:ij \in F} x_{ij} \le 1$). By Lemma 5, the matrix A defining this augmented set of inequalities is still a network matrix. The polytope defined by these inequalities is the (perfect) matching polytope of the graph B [16, Cor. 18.1b]; its vertices correspond to (perfect) matchings in B. This fact also follows easily from total unimodularity of the matrix A.

Now imagine that, for each arc $e = uv \in E$, we route f_e units of flow from u to v in the tree T. The total flow through tree arc $t \in T$ is then

$$\sum_{e \in F_t^+} f_e - \sum_{e \in F_t^-} f_e.$$

So we can think of the vertex counting problem in terms of integer flows in a network defined on $(V, \overleftarrow{E} \cup T)$. Arcs in E are reversed (denoted $\overleftarrow{E}$) and have a lower bound of 0 and an upper bound of 1. Arcs in T have upper bounds defined by the right-hand sides of the inequalities; thus arc $t \in T$ has a upper bound of b_t. The vertices of the corresponding polytope are in bijection with integer flows in the network. Unfortunately, we don't know an FPRAS for integer flows at this level of generality. However, we do have a positive result for a special case.

Proposition 9. *There is an FPRAS for the following problem: Given a network matrix A, together with a promise that $P = \{x : \mathbf{0} \le Ax \le \mathbf{1}\}$ is a 0/1 polytope, return the number of vertices of P.*

Proof. Applying the above translation yields a network in which each arc has capacity 1. In this special case, Jerrum, Sinclair and Vigoda [8, Cor. 8.2] show that the number of integral flows can be obtained by reduction to counting perfect matchings, for which there is an FPRAS. $\square$

There seems to be no strong reason to doubt that counting flows in networks with more general bounds admits an FPRAS. However, it appears that this extension of the known FPRAS for counting perfect matchings would require new ideas. Note that using network matrices we can encode problems such as *b*-matchings and *b*-edge covers in bipartite graphs. A *b-matching* (respectively, *b-edge cover*) of a graph is an edge subset that covers each vertex at most (respectively, at least) *b* times. For these problems, we have efficient approximation algorithms for some *b* but not in general: see Huang, Lu and Zhang [7].

Problem 10. *Is there an FPRAS for #VERTICES OF A 0/1 POLYTOPE, subject only to the restriction that A is a network matrix?*

5 Beyond totally unimodular matrices

Most integral polytopes arising in combinatorial optimisation arise from totally unimodular matrices. In fact, unimodular matrices are the only ones with the property that the polyhedron defined by $Ax \leq b$ is integral for all choices of the integral vector b. However, if we consider A and b together, it can happen that the pair (A, b) defines an integral polytope even when A is not totally unimodular. Since we have not so far discovered any family of 0/1 polytopes whose vertex-counting problem is harder than #BIS, it is tempting to look beyond totally unimodular.

A known class of integer polytopes arise from 'Totally Dual Integral' (TDI) pairs (A, b) [16, §5.17]. A fascinating example is provided by the stable matching polytope of a bipartite graph, which is defined by a natural system of polynomially many inequalities [22]. The defining matrix A is apparently not totally unimodular, but the linear system (A, b) was shown to be TDI by Király and Pap [11]. Intriguingly, Chebolu, Goldberg and Martin [3] have shown that the problem of counting stable matchings (and hence the vertices of the stable matching polytope) is interreducible with #BIS via approximation-preserving reductions. So, again, we do not manage to get beyond #BIS. This raises the question of whether the vertex counting problem considered here is #BIS-easy.

Problem 11. *Is #VERTICES OF A 0/1 POLYTOPE approximation-preserving reducible to #BIS in general?*

Acknowledgement

References

[1] David Avis and Luc Devroye. Estimating the number of vertices of a polyhedron. *Inform. Process. Lett.*, 73(3-4):137–143, 2000. 4

[2] Michel L. Balinski. Integer programming: Methods, uses, computation. *Management Sci.*, 12:253–313, 1965. 4

[3] Prasad Chebolu, Leslie Ann Goldberg, and Russell Martin. The complexity of approximately counting stable matchings. *Theoret. Comput. Sci.*, 437:35–68, 2012. 12

[4] Xi Chen, Martin Dyer, Leslie Ann Goldberg, Mark Jerrum, Pinyan Lu, Colin McQuillan, and David Richerby. The complexity of approximating conservative counting CSPs. *J. Comput. System Sci.*, 81(1):311–329, 2015. 9

[5] Guoli Ding, Li Feng, and Wenan Zang. The complexity of recognizing linear systems with certain integrality properties. *Math. Program.*, 114(2, Ser. A):321–334, 2008. 1

[6] Martin Dyer, Leslie Ann Goldberg, Catherine Greenhill, and Mark Jerrum. The relative complexity of approximate counting problems. *Algorithmica*, 38(3):471–500, 2004. 2, 8

[7] Lingxiao Huang, Pinyan Lu, and Chihao Zhang. Canonical paths for MCMC: from art to science. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 514–527. ACM, New York, 2016. 11

[8] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *J. ACM*, 51(4):671–697, 2004. 11

[9] Mark R. Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoret. Comput. Sci.*, 43(2-3):169–188, 1986. 3, 7, 10

[10] Leonid Khachiyan. Transversal hypergraphs and families of polyhedral cones. In *Advances in convex analysis and global optimization (Pythagorion, 2000)*, volume 54 of *Nonconvex Optim. Appl.*, pages 105–118. Kluwer Acad. Publ., Dordrecht, 2001. 3, 7, 12

[11] Tamás Király and Júlia Pap. Total dual integrality of Rothblum's description of the stable-marriage polyhedron. *Math. Oper. Res.*, 33(2):283–290, 2008. 12

[12] M. S. Krishnamoorthy. An NP-hard problem in bipartite graphs. *SIGACT News*, 7(1):26, January 1975. 5

[13] Milena Mihail and Umesh Vazirani. On the magnification of 0-1 polytopes. Technical Report TR-03-89, Harvard university, 1989. 3

[14] Lorenzo Najt. The intractability of uniformly sampling and counting the vertices of a polytope. 2019. unpublished. 3, 7, 12

[15] Robert Salomone, Radislav Vaisman, and Dirk Kroese. Estimating the number of vertices in convex polytopes. In *4th Annual International Conference on Operations Research and Statistics (ORS 2016)*. Global Science & Technology Forum (GSTF), January 2016. 4

[16] Alexander Schrijver. *Combinatorial optimization. Polyhedra and efficiency. Vol. A*, volume 24 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 2003. Paths, flows, matchings, Chapters 1–38. 4, 7, 8, 11, 12

[17] P. D. Seymour. Decomposition of regular matroids. *J. Combin. Theory Ser. B*, 28(3):305–359, 1980. 2

[18] Larry Stockmeyer. On approximation algorithms for #P. *SIAM J. Comput.*, 14(4):849–861, 1985. 2

[19] Klaus Truemper. A decomposition theory for matroids. V. Testing of matrix total unimodularity. *J. Combin. Theory Ser. B*, 49(2):241–281, 1990. 2, 7

[20] L. G. Valiant. The complexity of computing the permanent. *Theoret. Comput. Sci.*, 8(2):189–201, 1979. 4

[21] L. G. Valiant and V. V. Vazirani. NP is as easy as detecting unique solutions. *Theoret. Comput. Sci.*, 47(1):85–93, 1986. 2

[22] John H. Vande Vate. Linear programming brings marital bliss. *Oper. Res. Lett.*, 8(3):147–153, 1989. 12