Algorithms for Generating Minimal Blockers of Perfect Matchings in Bipartite Graphs and Related Problems⋆

Endre Boros, Khaled Elbassioni, and Vladimir Gurvich

RUTCOR, Rutgers University, 640 Bartholomew Road, Piscataway NJ 08854-8003;
{boros,elbassio,gurvich}@rutcor.rutgers.edu

Abstract. A minimal blocker in a bipartite graph G is a minimal set of edges the removal of which leaves no perfect matching in G. We give a polynomial delay algorithm for finding all minimal blockers of a given bipartite graph. Equivalently, this gives a polynomial delay algorithm for listing the anti-vertices of the perfect matching polytope $P(G) = \{x \in \mathbb{R}^E \mid Hx = \mathbf{e}, \quad x \geq 0\}$, where H is the incidence matrix of G. We also give similar generation algorithms for other related problems, including d-factors in bipartite graphs, and perfect 2-matchings in general graphs.

1   Introduction

Let $G = (A, B, E)$ be a bipartite graph with bipartition $A \cup B$ and edge set E. For subsets $A' \subseteq A$ and $B' \subseteq B$, denote by (A', B') the subgraph of G induced by these sets. A perfect matching in G is a subgraph in which every vertex in $A \cup B$ has degree exactly one, or equivalently a subset of $|A| = |B|$ pairwise disjoint edges of G. Throughout the paper, we shall always assume that a graph with no vertices has a perfect matching. A blocker in G is a subset of edges $X \subseteq E$ such that the graph $G(A, B, E \setminus X)$ does not have any perfect matching. A blocker X is minimal if, for every edge $e \in X$, the set $X \setminus \{e\}$ is not a blocker. Denote respectively by $\mathcal{M}(G)$ and $\mathcal{B}(G)$ the families of perfect matchings and minimal blockers for G. Note that in any bipartite graph G, the set $\mathcal{B}(G)$ is the family of minimal transversals to the family $\mathcal{M}(G)$, i.e. $\mathcal{B}(G)$ is the family of minimal edge sets containing an edge from every perfect matching in G. Clearly, the above definitions imply that if $|A| \neq |B|$, then $\mathcal{B}(G) = \{\emptyset\}$.

The main problem we consider in this paper is the following.

GEN$(\mathcal{B}(G))$**:** *Given a bipartite graph G, enumerate all minimal blockers of G.*

The analogous problem GEN$(\mathcal{M}(G))$ of enumerating perfect matchings for bipartite graphs was considered in [9,20].

In such generation problems the output size may be exponential in the input size, therefore the efficiency of an algorithm is evaluated by measuring the time it needs to find a *next* element, see e.g., [14]. More precisely, we say that the elements of a set $\mathcal{S}$ represented in some implicit way by the input I are generated with *polynomial delay*, if the generation algorithm produces all elements of $\mathcal{S}$ in some order such that the computational time between any two outputs is limited by a polynomial expression of the input size. The generation is said to be *incrementally polynomial*, if the time needed to find the $k+1$st element, after the generation of the first k elements, depends polynomially not only on the input size but also on k.

The problems of generating perfect matchings and minimal blockers are special cases of the more general open problems of generating vertices and anti-vertices of a polytope given by its linear description. Indeed, let H be the vertex-edge incidence matrix of $G = (A, B, E)$, and consider the polytope $P(G) = \{x \in \mathbb{R}^E \mid Hx = \mathbf{e}, \quad x \geq 0\}$, where $\mathbf{e} \in \mathbb{R}^{|A|+|B|}$ is the vector of all ones. Then the perfect matchings of G are in one-to-one correspondence with the vertices of $P(G)$, which in turn, are in one-to-one correspondence with the minimal collections of columns of H containing the vector $\mathbf{e}$ in their conic hull. On the other hand, the minimal blockers of G are in one-to-one correspondence with the anti-vertices of $P(G)$, i.e. the maximal collections of columns of H, not containing $\mathbf{e}$ in their conic hull. Both corresponding generating problems are open in general, see [2] for details. In the special case, when H is the incidence matrix of a bipartite graph G, the problem of generating the vertices of $P(G)$ can be solved with polynomial delay [9,20]. In this note, we obtain an analogous result for anti-vertices of $P(G)$.

Theorem 1. *Problem $GEN(\mathcal{B}(G))$ can be solved with polynomial delay.*

For non-bipartite graphs G, it is well-know that the vertices of $P(G)$ are half-integral [16] (i.e. the components of each vertex are in $\{0, 1, 1/2\}$), and that they correspond to the basic 2-matchings of G, i.e. subsets of edges that cover the vertices with vertex-disjoint edges and vertex-disjoint odd cycles. Polynomial delay algorithms exist for listing the vertices of 0/1-polytopes, simple and simplicial polytopes [1,5], but the status of the problem for general polytopes is still open. We show here that for the special case of the polytope $P(G)$ of a graph G, the problem can be solved in incremental polynomial time.

Theorem 2. *All basic perfect 2-matchings of a graph G can be generated in incremental polynomial time.*

The proof of Theorem 1 is based on a nice characterization of minimal blockers, which may be of independent interest. The method used for enumeration is the supergraph method which will be outlined in Section 3.1. A special case of this method, when applicable, can provide enumeration algorithms with stronger properties, such as enumeration in lexicographic ordering, or enumeration with polynomial space. This special case will be described in Section 3.2 and will be used in Section 5 to solve two other related problems of enumerating d-factors

in bipartite graphs, and enumerating perfect 2-matchings in general graphs. The latter result will be used to prove Theorem 2.

2   Properties of Minimal Blockers

2.1   The Neighborhood Function

Let $G = (A, B, E)$ be a bipartite graph. For a subset $X \subseteq A \cup B$, denote by $\Gamma(X) = \Gamma_G(X) = \{v \in (A \cup B) \setminus X \mid \{v, x\} \in E$ for some $x \in X\}$ the set of neighbors of X. It is known (see, e.g., [15]), and also easy to verify, that set-function $|\Gamma(\cdot)|$ is submodular, i.e. $|\Gamma(X \cup Y)| + |\Gamma(X \cap Y)| \leq |\Gamma(X)| + |\Gamma(Y)|$ holds for all subsets $X, Y \subseteq A \cup B$. A simple but useful observation about submodular set-functions is the following (see, e.g., [17]):

Proposition 1. *Let V be a finite set and $f : 2^V \mapsto \mathbb{R}$ be a submodular set-function. Then the family of sets X minimizing $f(X)$ forms a sub-lattice of the Boolean lattice 2^V. If $f(X)$ is polynomial-time computable for every $X \subseteq V$, then the unique minimum and maximum of this sub-lattice can be computed in polynomial time.*

2.2   Matchable Graphs

Matchable graphs, a special class of bipartite graphs with a *positive surplus* (see Section 1.3 in [16] for definitions and properties), were introduced in [12]. They play an important role in the characterization of minimal blockers.

Definition 1 ([12]). *A bipartite graph $G = (A, B, E)$ with $|A| = |B| + 1$ is said to be matchable if the graph $G \setminus v$ has a perfect matching for every vertex $v \in A$.*

The smallest matchable graph is one in which $|A| = 1$ and $|B| = 0$. The following is a useful equivalent characterization of matchable graphs.

Proposition 2 ([12]). *For a bipartite graph $G = (A, B, E)$ with $|A| = |B| + 1$, the following statements are equivalent:*

(M1) *G is matchable;*
(M2) *G has a matchable spanning tree, that is, a spanning tree T in which every node $v \in B$ has degree 2.*

It can be shown that for bipartite graphs $G = (A, B, E)$ with $|A| = |B| + 1$ these properties are further equivalent with

(M3) *G contains a unique independent set of size $|A|$, namely the set A.*

Given a bipartite graph G, let us say that a matchable subgraph of G is *maximal*, if it is not properly contained in any other matchable subgraph of G. The following property can be derived from a result by Dulmage and Mendelsohn, or more generally from the Gallai-Edmonds Structure Theorem (see Section 3.2 in [16] for details and references).

Proposition 3. *Let $G = (A, B, E)$ be a matchable graph with $|A| = |B| + 1$, and let $a \in A$ and $b \in B$ be given vertices. Then the graph $G' = G - \{a, b\}$ is the union of a maximal matchable subgraph $G_1 = (A_1, B_1)$ of G' and a (possibly empty) subgraph $G_2 = (A_2, B_2)$ which has a perfect matching, such that the graph (A_1, B_2) is empty. Furthermore, this decomposition of G' into G_1 and G_2 is unique, and can be found in polynomial time.*

The next proposition acts, in a sense, as the opposite to Proposition 3, and can be derived in a similar way: Given a decomposition of a matchable graph into a matchable subgraph and a subgraph with a perfect matching, we can reconstruct in a *unique* way the original matchable graph.

Proposition 4. *Let $G = (A, B, E)$ be a bipartite graph and $G_1 = (A_1, B_1)$ and $G_2 = (A_2, B_2)$ be two vertex-disjoint subgraphs of G such that G_1 is matchable, G_2 has a perfect matching, and $A = A_1 \cup A_2$ and $B = B_1 \cup B_2$. Then there is a unique extension of G_1 into a maximal matchable subgraph $G'_1 = (A'_1, B'_1)$ of G such that the graph $G'_2 \stackrel{\text{def}}{=} (A_1 \cup A_2 \setminus A'_1, B_1 \cup B_2 \setminus B'_1)$ has a perfect matching. Such an extension is the unique possible decomposition of G into a maximal matchable graph plus a graph with a perfect matching, and can be computed in polynomial time.*

As a simple application of Proposition 4, we obtain the following corollary.

Corollary 1. *Let $G = (A, B, E)$ be a bipartite graph and $G' = (A', B')$ be a matchable subgraph of G. Given two vertices $a \in A \setminus A'$ and $b \in B \setminus B'$ such that there is an edge $\{b, a'\}$ for some $a' \in A'$, there exists a unique decomposition of $G' = (A' \cup \{a\}, B' \cup \{b\})$ into a maximal matchable graph plus a graph with a perfect matching.*

The next property, which can be derived analogously to Proposition 3, implies that matchable graphs are "continuous" in the sense that we can reach from a given matchable graph to any matchable graph containing it by repeatedly appending pairs of vertices, one at a time, always remaining within the class of matchable graphs.

Proposition 5. *Let $T = (A, B, E)$ and $T' = (A', B', E')$ be two matchable spanning trees such that $|A| = |B| + 1$, $|A'| = |B'| + 1$, $A \subseteq A'$ and $B \subseteq B'$. Then (i) there exists a vertex $b \in B' \setminus B$ such that $|\Gamma_{T'}(\{b\}) \cap A| \geq 1$, (ii) for any such vertex b, there exists a vertex $a \in A' \setminus A$, such that the graph $G[a, b] \stackrel{\text{def}}{=} (A \cup \{a\}, B \cup \{b\}, E \cup E')$ is matchable.*

The above properties readily imply the following corollary.

Corollary 2. *Let $G = (A, B, E)$ and $G' = (A', B', E')$ be two matchable graphs such that $|A| = |B| + 1$, $|A'| = |B'| + 1$, $A \subset A'$, and $B \subset B'$. Then there exists a pair of vertices $a \in A' \setminus A$ and $b \in B' \setminus B$ such that the graph $(A \cup \{a\}, B \cup \{b\}, E \cup E')$ is matchable.*

To arrive to a useful characterization of minimal blockers, we need one more definition.

Definition 2. *Let $G = (A, B, E)$ be a bipartite graph having a perfect matching. A matchable split of G, denoted by $[(A_1, B_1), (A_2, B_2), (A_3, B_3)]$, is a partition of the vertex set of G into sets $A_1, A_2, A_3 \subseteq A$ and $B_1, B_2, B_3 \subseteq B$ such that*

(B1) $|A_1| = |B_1| + 1$, $|A_2| = |B_2| - 1$, $|A_3| = |B_3|$,
(B2) *the graph $G_1 = (A_1, B_1)$ is matchable,*
(B3) *the graph $G_2 = (A_2, B_2)$ is matchable,*
(B4) *the graph $G_3 = (A_3, B_3)$ has a perfect matching, and*
(B5) *the graphs (A_1, B_3) and (A_3, B_2) are empty.*

Denote by $\mathcal{S}(G)$ the family of all matchable splits of G. We are now ready to state our characterization for minimal blockers in bipartite graphs.

Lemma 1. *Let $G = (A, B, E)$ be a bipartite graph in which there exists a perfect matching. Then a subset of edges $X \subseteq E$ is a minimal blocker if and only if there is a matchable split $[(A_1, B_1), (A_2, B_2), (A_3, B_3)]$ of G, such that $X = \{\{a, b\} \in E : a \in A_1, b \in B_2\}$ is the set of all edges between A_1 and B_2 in G. This correspondence between minimal blockers and matchable splits is one-to-one. Given one representation we can compute the other in polynomial time.*

3   Enumeration Algorithms

Given a (bipartite) graph $G = (V, E)$, consider the problem of listing all sub-graphs of G, or correspondingly, the family $\mathcal{F}_\pi \subseteq 2^E$ of all minimal subsets of E, satisfying some monotone property $\pi : 2^E \mapsto \{0, 1\}$. For instance, if $\pi(X)$ is the property that the subgraph with edge set $X \subseteq E$ has a perfect matching, then $\mathcal{F}_\pi$ is the family of perfect matchings of G. Enumeration algorithms for listing subgraphs satisfying a number of monotone properties are well known. For instance, it is known [19] that the problems of listing all minimal cuts or all spanning trees of an undirected graph $G = (V, E)$ can be solved with delay $O(|E|)$ per generated cut or spanning tree. It is also known (see e.g., [7,10,18]) that all minimal (s, t)-cuts or (s, t)-paths, can be listed with delay $O(|E|)$ per cut or path. Furthermore, polynomial delay algorithms also exist for listing perfect matchings, maximal matchings, maximum matchings in bipartite graphs, and maximal matchings in general graphs, see e.g. [9,20,21].

In the next subsections we give an overview of two commonly used techniques for solving such enumeration problems.

3.1   The Supergraph Approach

This technique works by building and traversing a directed graph $\mathcal{G} = (\mathcal{F}_\pi, \mathcal{E})$, defined somehow on the set of elements of the family to be generated $\mathcal{F}_\pi \subseteq 2^E$. The arcs of $\mathcal{G}$ are defined by a *polynomial-time computable* neighborhood function $\mathcal{N} : \mathcal{F}_\pi \mapsto 2^{\mathcal{F}_\pi}$ that defines, for any $X \in \mathcal{F}_\pi$ the set of its outgoing neighbors $\mathcal{N}(X)$ in $\mathcal{G}$. A special vertex $X_0 \in \mathcal{F}_\pi$ is identified from which all other vertices of $\mathcal{G}$ are reachable. The method works by performing a (breadth- or depth-first search) traversal on the nodes of $\mathcal{G}$, starting from X_0. The following is a basic fact about this approach:

(S1) If $\mathcal{G}$ is strongly connected and $|\mathcal{N}(X)| \leq p(|\pi|)$ for every $X \in \mathcal{F}_\pi$, where $p(|\pi|)$ is a polynomial that depends only on the size of the description of π, then the supergraph approach gives us a polynomial delay algorithm for enumerating $\mathcal{F}_\pi$, starting from an arbitrary set $X_0 \in \mathcal{F}_\pi$.

Let us consider as an example the *generation of minima of submodular functions*: Let $f : 2^V \mapsto \mathbb{R}$ be a submodular function, $\alpha = \min\{f(X) \; : \; X \subseteq V\}$, and $\pi(X)$ be the property that $X \subseteq V$ contains a minimizer of f. By Proposition 1, the set $\mathcal{F}_\pi$ forms a sub-lattice $\mathcal{L}$ of 2^V, and the smallest element X_0 of this lattice can be computed in polynomial time. For $X \in \mathcal{F}_\pi$, define $\mathcal{N}(X)$ to be the set of subsets $Y \in \mathcal{L}$ that immediately succeed X in the lattice order. The elements of $\mathcal{N}(X)$ can be computed by finding, for each $v \in V \setminus X$, the smallest cardinality minimizer $X' = \mathrm{argmin}\{f(Y) \; : \; Y \supseteq X \cup \{v\}\}$ and checking if $f(X') = \alpha$. Then $|\mathcal{N}(X)| \leq |V|$, for all $X \in \mathcal{F}_\pi$. Note also that the definition of $\mathcal{N}(\cdot)$ implies that the supergraph is strongly connected. Thus (S1) implies that all the elements of $\mathcal{F}_\pi$ can be enumerated with polynomial delay.

3.2   The Flashlight Approach

This can be regarded as an important special case of the supergraph method. Assume that we have fixed some order on the elements of E. Let X_0 be the lexicographically smallest element in $\mathcal{F}_\pi$. For any $X \in \mathcal{F}_\pi$, let $N(X)$ consist of a single element, namely, the element next to X in lexicographic ordering. Thus the supergraph $\mathcal{G}$ in this case is a Hamiltonian path on the elements of $\mathcal{F}_\pi$ (see [19] for general background on backtracking algorithms). The following is a sufficient condition for the operator $\mathcal{N}(\cdot)$ (and also for the element X_0) to be computable in polynomial time:

(F1) For any two disjoint subsets S_1, S_2 of E, we can check in polynomial time $p(|\pi|)$ if there is an element $X \in \mathcal{F}_\pi$, such that $X \supseteq S_1$ and $X \cap S_2 = \emptyset$.

The traversal of $\mathcal{G}$, in this case, can be organized in a backtracking tree of depth $|E|$, whose leaves are the elements of $\mathcal{F}_\pi$, as follows. Each node of the tree is identified with two disjoint subsets $S_1, S_2 \subseteq E$, and have at most two children. At the root of the tree, we have $S_1 = S_2 = \emptyset$. The left child of any node (S_1, S_2) of the tree at level i is $(S_1 \cup \{i\}, S_2)$ provided that there is an $X \in \mathcal{F}_\pi$, such that $X \supseteq S_1 \cup \{i\}$ and $X \cap S_2 = \emptyset$. The right child of any node (S_1, S_2) of the tree at level i is $(S_1, S_2 \cup \{i\})$ provided that there is an $X \in \mathcal{F}_\pi$, such that $X \supseteq S_1$ and $X \cap (S_2 \cup \{i\}) = \emptyset$. Furthermore, we may restrict our attention to subsets S_1 satisfying certain properties. More precisely, let $\mathcal{F}'_\pi \subseteq 2^E$ be a family of sets, containing $\mathcal{F}_\pi$, such that we can test in polynomial time if a set X belongs to it, and such that for every $X \in \mathcal{F}_\pi$, there is a set $S \in \mathcal{F}'_\pi$ contained in X. Then the following is a weaker requirement than that of (F1):

(F2) For any two disjoint subsets $S_1 \in \mathcal{F}'_\pi$ and $S_2 \subseteq E$, and given element $i \in E \setminus (S_1 \cup S_2)$ such that $S_1 \cup \{i\} \in \mathcal{F}'_\pi$, we can check in polynomial time $p(|\pi|)$ if there is an element $X \in \mathcal{F}_\pi$, such that $X \supseteq S_1 \cup \{i\}$ and $X \cap S_2 = \emptyset$.

This way, under assumption (F2), we get a polynomial delay, polynomial space algorithm for enumerating the elements of $\mathcal{F}_\pi$ in lexicographic order.

As an example, let us consider the *generation of maximal matchable subgraphs*: Let $G = (A, B, E)$ be a bipartite graph. For subsets $A' \subseteq A$ and $B' \subseteq B$, let $\pi(A', B')$ be the property that the graph (A', B') contains a maximal matchable subgraph. To generate the family of maximal matchable graphs $\mathcal{F}_\pi$, we introduce an artificial element $*$, and consider the ground set $E' = \{(a, b) \ : \ a \in A, \ b \in B \cup \{*\}\}$. Any $X \subseteq E'$ represents an induced subgraph $G(X) = (A', B')$ of G where $A' = \{a \ : \ (a, b) \in X\}$ and $B' = \{b \ : \ (a, b) \in X, \ b \neq *\}$. Furthermore, let us say that the elements of $X \subseteq E'$ are disjoint if for every pair of elements $(a, b), (a', b') \in X$, we have $a \neq a'$ and $b \neq b'$. Let $\mathcal{F}'_\pi = \{X \subseteq E' \ : \ $ the elements of X are disjoint, one of them contains *, and $G(X)$ is matchable$\}$. Given two disjoint sets of pairs $S_1, S_2 \subseteq E'$, such that $S_1 \in \mathcal{F}'_\pi$, and a pair $(a, b) \in E' \setminus (S_1 \cup S_2)$, the check in (F2) can be performed in polynomial time. Indeed, by Corollary 2, all what we need to check is that the graph on $S_1 \cup \{a, b\}$ is matchable.

4   Polynomial Delay Generation of Minimal Blockers

In this section, we use the supergraph approach to show that minimal blockers can be enumerated with polynomial delay. Using Lemma 1, we may equivalently consider the generation of matchable splits. We start by defining the neighborhood function used for constructing the supergraph $\mathcal{G}_\mathcal{S}$ of matchable splits.

4.1   The Supergraph

Let $G = (A, B, E)$ be a bipartite graph that has a perfect matching. Given a matchable split $X = [G_1 = (A_1, B_1), G_2 = (A_2, B_2), G_3 = (A_3, B_3)] \in \mathcal{S}(G)$, the set of out-going neighbors of X in $\mathcal{G}_\mathcal{S}$ are defined as follows. For each edge $\{a, b\} \in E$ connecting a vertex $a \in A_2$ to a vertex $b \in B_2$, such that b is also connected to some vertex $a' \in A_1$, there is a unique neighbor $X' = \mathcal{N}(X, a, b)$ of X, obtained by the following procedure:

1. Let $G''_1 = (A_1 \cup \{a\}, B_1 \cup \{b\})$. Because of the edges $\{a, b\}, \{a', b\} \in E$, the graph G''_1 is matchable, see Corollary 1.

2. Delete the vertices a, b from G_2. This splits the graph $G_2 - \{a, b\}$ in a unique way into a matchable graph $G'_2 = (A'_2, B'_2)$ and a graph with a perfect matching $G''_2 = (A''_2, B''_2)$ such that there are no edges in E between A''_2 and B''_2, see Proposition 3.

3. Let $G'''_3 = G_3 \cup G''_2$. Then the graph G'''_3 has a perfect matching. Using Proposition 4, extend G'''_3 in the union $G''_1 \cup G'''_3 = (A''_3, B''_3)$ into a maximal matchable graph $G'_1 = (A'_1, B'_1)$ such that the graph $G'_3 = (A'_3, B'_3) = (A''_3 \setminus A'_1, B''_3 \setminus B'_1)$ has a perfect matching.

4. Let $X' = [(A'_1, B'_1), (A'_2, B'_2), (A'_3, B'_3)]$, and note that $X' \in \mathcal{S}(G)$.

Similarly, for each edge $\{a, b\} \in E$ connecting a vertex $a \in A_1$ to a vertex $b \in B_1$, such that a is also connected to some vertex $b' \in B_2$, there is a unique neighbor X' of X, obtained by a procedure similar to the above.

4.2 Strong Connectivity

For the purpose of generating minimal blockers in a bipartite graph $G = (A, B, E)$, we may assume without loss of generality that

(A1) The graph G is connected.
(A2) Every edge in G appears in a perfect matching.

It is easy to see that both properties can be checked in polynomial time (see *elementary bipartite graphs* in [16]). Indeed, if G has a number of connected components, then the set of minimal blockers of G is the union of the sets of minimal blockers in the different components, computed individually for each component. Furthermore, all the edges in G that do not appear in any perfect matching can be removed (in polynomial time), since they do not appear in any minimal blocker. In this section, we prove the following.

Lemma 2. *Under the assumptions (A1) and (A2), the supergraph $\mathcal{G}_S$ is strongly connected.*

Clearly Lemma 2 implies Theorem 1, according to **(S1)**. Call the set of edges connected to a given vertex $v \in A \cup B$ a *v-star*, and it by denote by v^*. We start with the following lemma.

Lemma 3. *(i) Under the assumption (A1) and (A2), every star is a minimal blocker. (ii) The matchable split corresponding to an a-star, $a \in A$, is $[(\{a\}, \emptyset), (A \setminus \{a\}, B), (\emptyset, \emptyset)]$. (iii) The matchable split corresponding to a b-star, $b \in B$, is $[(A, B \setminus \{b\}), (\emptyset, \{b\}), (\emptyset, \emptyset)]$.*

Given two matchable splits $X = [(A_1, B_1), (A_2, B_2), (A_3, B_3)]$ and $X' = [(A'_1, B'_1), (A'_2, B'_2), (A'_3, B'_3)]$, let us say that X and X' are *perfectly nested* if

(N1) $A_1 \subseteq A'_1$ and $B_1 \subseteq B'_1$,
(N2) $A_2 \supseteq A'_2$ and $B_2 \supseteq B'_2$, and
(N3) each of the graphs $(A_2 \cap A'_1, B_2 \cap B'_1)$, $(A_3 \cap A'_1, B_3 \cap B'_1)$, $(A_2 \cap A'_3, B_2 \cap B'_3)$, and $(A_3 \cap A'_3, B_3 \cap B'_3)$, has a perfect matching.

The strong connectivity of $\mathcal{G}_S$ is a consequence of the following fact.

Lemma 4. *There is a path in $\mathcal{G}_S$ between any pair of perfectly nested matchable splits.*

Proof. Let $X = [(A_1, B_1), (A_2, B_2), (A_3, B_3)]$, $X' = [(A'_1, B'_1), (A'_2, B'_2), (A'_3, B'_3)]$ be two matchable splits, satisfying (N1), (N2) and (N3) above. We Show that there is a path in $\mathcal{G}_S$ from X to X'. A path in the opposite direction can be found similarly. Fix a matching M in the graph $(A'_1 \cap A_2, B'_1 \cap B_2)$. By Proposition 5, there is a vertex $b \in B'_1 \setminus B_1$, such that b is connected by an edge $\{a, b\}$ to some vertex $a \in A_1$. Note that $b \notin B_3 \cap B'_1$ since the graph (A_1, B_3) is empty. Thus $b \in B_2 \cap B'_1$. Let $a' \in A'_1 \cap A_2$ be the vertex matched by M to b. Now consider the neighbor $X'' = \mathcal{N}(X, a', b) = [(A''_1, B''_1), (A''_2, B''_2), (A''_3, B''_3)]$ of X in $\mathcal{G}_S$. We claim that X'' is perfectly nested with respect to X'. To see this

observe, by Proposition 2, that the graph $(A_2, B_2) - \{a', b\}$ is decomposed, in a unique way, into a matchable graph $G_2'' = (A_2'', B_2'')$ and a graph with a perfect matching $G_2''' = (A_2''', B_2''')$. On the other hand, the graph (A_2', B_2') is matchable, while the graph $(A_2 \cap A_1' \setminus \{a'\}, B_2 \cap B_1' \setminus \{b\}) \cup (A_2 \cap A_3', B_2 \cap B_3')$ has a perfect matching, and therefore by Proposition 3, the union of these two graphs (which is $(A_2, B_2) - \{a', b\}$) decomposes in a unique way into a matchable graph F_1 containing (A_2', B_2') and a graph with a perfect matching F_2. The uniqueness of the decomposition implies that $F_1 = G_2''$ and $F_2 = G_2'''$, i.e. $A_2'' \supseteq A_2'$ and $B_2'' \supseteq B_2'$. Note that the graph $(A_2'' \cap A_1', B_2'' \cap B_1')$ still has a perfect matching. Note also that the graph (A_1'', B_1'') is obtained by extending the matchable graph $(A_1 \cup \{a'\}, B_1 \cup \{b\})$ with pairs from the graph $(A_2''', B_2''') \cup (A_1' \cap A_3, B_1' \cap B_3)$, which has a perfect matching. Such an extension must stay within the graph (A_1', B_1'), i.e. $A_1'' \subseteq A_1'$ and $B_1'' \subseteq B_1'$, since there are no edges between A_1' and $B_2'' \cap B_3'$. Finally, since the extension leaves a perfect matching in the graph $(A_2'' \setminus A_1'', B_2'' \setminus B_1'') \cup (A_1' \cap A_3 \setminus A_1'', B_1' \cap B_3 \setminus B_1'')$, we conclude that X'' and X' are perfectly nested. This way, we obtained a neighbor X'' of X that is closer to X' in the sense that $|A_1''| > |A_1|$. This implies that there is a path in $\mathcal{G}_{\mathcal{S}}$ from X to X' of length at most $|A_1' \setminus A_1|$. □

Proof of Lemma 2. Let $X = [(A_1, B_1), (A_2, B_2), (A_3, B_3)]$ be a matchable split. Let a be any vertex in A_1, then the matchability of A_1 implies that the graph $(A_1 \setminus \{a\}, B_1)$ has a perfect matching. Thus, with respect to a^* and X, the conditions (N1)-(N3) hold, implying that they are perfectly nested. Lemma 4 hence implies that there are paths in $\mathcal{G}_{\mathcal{S}}$ from a^* to X and from X to a^*. Similarly we can argue that there are paths in $\mathcal{G}_{\mathcal{S}}$ between X and b^* for any $b \in B_2$. In particular, there are paths in $\mathcal{G}_{\mathcal{S}}$ between a^* and b^* for any $a \in A$ and $b \in B$. The lemma follows. □

5 Some Generalizations and Related Problems

5.1 d-Factors

Let $G = (A, B, E)$ be a bipartite graph, and $d : A \cup B \mapsto \{0, 1 \ldots, |A| + |B|\}$ be a non-negative function assigning integer weights to the vertices of G. We shall assume in what follows that, for each vertex $v \in A \cup B$, the degree of v in G is at least $d(v)$. A *d-factor* of G is a subgraph (A, B, X) of G in which each vertex $v \in A \cup B$ has degree $d(v)$ (see Section 10 in [16]), i.e., perfect matchings are the 1-factors. Note that d-factors of G are in one-to-one correspondence with the vertices of the polytope $P(G) = \{x \in \mathbb{R}^E \mid Hx = d, \ 0 \leq x \leq 1\}$, where H is the incidence matrix of G. In particular, checking if a bipartite graph has a d-factor can be done in polynomial time by solving a *capacitated transportation problem* with edge capacities equal to 1, see, e.g., [6].

Since the d-factors are the vertices of a 0/1-polytope, they can be computed with polynomial delay [5]. We present below a more efficient procedure.

Theorem 3. *For any given bipartite graph $G = (A, B, E)$, and any non-negative integer function $d : A \cup B \mapsto \{0, 1 \ldots, |A| + |B|\}$, after computing the first d-factor, all other d-factors of G can be enumerated with delay $O(|E|)$.*

Proof. We use the flashlight method. Let $\mathcal{M}_d(G)$ be the set of d-factors of G. It is enough to check that condition (F1) is satisfied. Given $S_1, S_2 \subseteq E$, we can check in polynomial time whether there is an $X \in \mathcal{M}_d(G)$ such that $X \supseteq S_1$ and $X \cap S_2 = \emptyset$ by checking the existence of a d'-factor in the graph $(A, B, E \setminus S_2)$, where $d'(v) = d(v) - 1$ if $v \in A \cup B$ incident to some edge in S_1 and $d'(v) = d(v)$ for all other vertices $v \in A \cup B$.

A more efficient procedure can be obtained as a straightforward generalization of the one in [9]. It is a slight modification of the flashlight method that avoids checking the existence of a d-factor at each node in the backtracking tree. Given a d-factor X in G, it is possible to find another one, if it exists, by the following procedure. Orient all the edges in X from A to B and all the other edges in $E \setminus X$ from B to A. Then it is not difficult to see that the resulting directed graph G' has a directed cycle if and only if G contains another d-factor $X' \in \mathcal{M}_d(G)$. Such an X' can be found by finding a directed cycle C in G', and taking the symmetric difference between the edges of C and X.

Now, the backtracking algorithm proceeds as follows. Each node w of the backtracking tree is identified with a weight function d_w, a d_w-factor X_w, and a bipartite graph $G_w = (A, B, E_w)$. At the root r of the tree, we compute a d-factor X in G, and let $d_w \equiv d$ and $G_w = G$. At any node w of the tree, we check if there is another d_w-factor X'_w using the above procedure, and if there is none, we define w to be a leaf. Otherwise, we let e_w be an arbitrary edge in $X_w \setminus X'_w$. This defines the two children u and z of w by: $d_u(v) = d_w(v) - 1$ for $v \in e_w$ and $d_u(v) = d_w(v)$ for all other vertices $v \in A \cup B$, $X_u = X_w$, $G_u = G_w$, $d_z \equiv d_w$, $X_z = X'_w$, and finally, $G_z = G_w - e_w$. The d-factors computed at the nodes of the backtracking tree are distinct and they form the complete set of d-factors of G. □

5.2   2-Matchings in General Graphs

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. Let H be the incidence matrix of G. As stated in the introduction, if G is a bipartite graph then the perfect matchings of G correspond to the vertices of the polytope $P(G) = \{x \in \mathbb{R}^E \mid Hx = \mathbf{e}, \;\; x \geq 0\}$. In general graphs the vertices of $P(G)$ are in one-to-one correspondence with the *basic* perfect 2-*matchings* of G, i.e. subsets of edges that form a cover of the vertices with vertex-disjoint edges and vertex-disjoint odd cycles (see [16]). A (not necessarily basic) perfect 2-matching of G is a subset of edges that covers the vertices of G with vertex-disjoint edges and vertex-disjoint (even or odd) cycles. Denote respectively by $\mathcal{M}_2(G)$ and $\mathcal{M}'_2(G)$ the families of perfect 2-matchings and basic perfect 2-matchings of a graph G. We show below that, the family $\mathcal{M}_2(G)$ can be enumerated with polynomial delay, and the family $\mathcal{M}'_2(G)$ can be enumerated in incremental polynomial time. Theorem 2 will follow from the following two lemmas.

Lemma 5. *All perfect 2-matchings of a graph G can be generated with polynomial delay.*

Proof. We use the flashlight method with a slight modification. For $X \subseteq E$, let $\pi(X)$ be the property that the graph (V, X) has a perfect 2-matching. Then $\mathcal{F}_\pi = \mathcal{M}_2(G)$. Define $\mathcal{F}'_\pi = \{X \subseteq E \ : \ $ the graph (V, X) is a vertex-disjoint union of some cycles, some disjoint edges, and possibly one path which maybe of length one$\}$. Clearly, any $X \in \mathcal{M}_2(G)$ contains some set $X' \in \mathcal{F}'_\pi$. Given $S_1 \subseteq \mathcal{F}'_\pi$, $S_2 \subseteq E$, we modify the basic approach described in Section 3.2 in two ways. First, when we consider a new edge $e \in E \setminus (S_1 \cup S_2)$ to be added to S_1, we first try an edge incident to the path in S_1, if one exists. If no such path exits, then any edge $e \in E \setminus (S_1 \cup S_2)$ can be chosen and defined to be a path of length one in $S_1 \cup \{e\}$. Second, when we backtrack, for the first time, on an edge e defining a path of length one in S_1, we redefine S_1 by considering e as a single edge rather than a path of length one. Now it remains to check that (F2) is satisfied. Given $S_1 \subseteq \mathcal{F}'_\pi$, $S_2 \subseteq E$, and an edge $e \in E \setminus (S_1 \cup S_2)$, chosen as above, such that $S_1 \cup \{e\} \in \mathcal{F}'_\pi$, we can check in polynomial time whether there is an $X \in \mathcal{M}_2(G)$ such that $X \supseteq S_1 \cup \{e\}$ and $X \cap S_2 = \emptyset$ in the following way. First, we delete all edges in S_2 from G. Second, we construct an auxiliary bipartite graph G^b as follows (see [16]). For every vertex v of G we define two vertices v' and v'' in G^b, and for every edge $\{u, v\}$ in G we define two edges $\{u'v''\}$ and $\{u'', v'\}$ in G^b. Third, we orient all the edges in $S_1 \cup \{e\}$ in such a way that all cycles and the path (if it exists) become directed. Single edges in $S_1 \cup \{e\}$ get double orientations. Finally, we mark edges in G^b corresponding to the oriented arcs in X: for an arc (u, v) in X, we mark the corresponding edge $\{u', v''\}$ in G^b. It is easy to see that there is an $X \in \mathcal{M}_2(G)$ such that $X \supseteq S_1 \cup \{e\}$ and $X \cap S_2 = \emptyset$ if and only if there is a perfect matching in G^b extending the marked edges. $\square$

Lemma 6. *Let $\mathcal{M}_2(G)$ and $\mathcal{M}'_2(G)$ be respectively the families of perfect 2-matchings and basic perfect 2-matchings of a graph $G = (V, E)$. Then*

$$|\mathcal{M}_2(G)| \leq |\mathcal{M}'_2(G)| + \binom{|\mathcal{M}'_2(G)|}{2}.$$

Proof of Theorem 2. By generating all perfect 2-matchings of G and discarding the non-basic ones, we can get generate all basic perfect 2-matchings. By Lemma 6, the total time for this generation is polynomial in $|V|$, $|E|$, and $|\mathcal{M}'_2(G)|$. Since the problem is self-reducible, we can convert this output polynomial generation algorithm to an incremental one, see [3,4,8] for more details. $\square$

References

1. D. Avis, K. Fukuda, A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra, *Symposium on Computational Geometry* 1991, North Conway, NH, pp. 98–104.

2. E. Boros, K. Elbassioni, V. Gurvich and L. Khachiyan, On enumerating minimal dicuts and strongly connected subgraphs, *Integer Programming and Combinatorial Optimization, 10th International IPCO Conference*, (D. Bienstock and G. Nemhauser, eds.) Lecture Notes in Computer Science 3064 (2004) pp. 152–162.

3. E. Boros, K. Elbassioni, V. Gurvich and L. Khachiyan, Algorithms for enumerating circuits in matroids, in *Proc. 14th Annual International Symposium on Algorithms and Computation (ISAAC 2003)*, *LNCS* 2906, pp. 485–494, Kyoto, Japan, 2003.

4. E. Boros, K. Elbassioni, V. Gurvich and L. Khachiyan, Generating maximal independent sets for hypergraphs with bounded edge-intersections, *6th Latin American Theoretical Informatics Conference (LATIN 2004)*, (Martin Farach-Colton, ed.) Lecture Notes in Computer Science 2461, pp. 424–435.

5. M. Bussieck and M. Lübbecke, The vertex set of a 0/1-polytope is strongly p-enumerable, *Computational Geometry: Theory and Applications* 11(2), pp. 103–109, 1998.

6. W. Cook, W. Cunningham, W. Pulleyblank and A. Schrijver, *Combinatorial Optimization*, John Wiley and Sons, New York, Chapter 4, pp. 91-126.

7. N. Curet, J. DeVinney and M. Gaston, An efficient network flow code for finding all minimum cost s-t cutsets, *Computers and Operations Research* **29** (2002), pp. 205–219.

8. T. Eiter, G. Gottlob, and K. Makino, New results on monotone dualization and generating hypergraph transversals, *SIAM J. Computing*, 32 (2003) 514–537.

9. K. Fukuda and T. Matsui, Finding all minimum cost perfect matchings in bipartite graphs, Networks 22, 1992.

10. D. Gusfield and D. Naor, Extracting maximum information about sets of minimum cuts, *Algorithmica* **10** (1993) pp. 64–89.

11. M. Hall, *Combinatorial theory* (Blaisdell Publ. Co., Waltham, 1967).

12. P.L. Hammer and I.E. Zverovich, Constructing of a maximum stable set with k-extensions, RUTCOR Research Report RRR 5-2002, Rutgers University, http://rutcor.rutgers.edu/˜rrr/2002.html.

13. S. Iwata, L. Fleischer, S. Fujishige: A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions, *STOC 2000*, pp. 97-106.

14. E. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, Generating all maximal independent sets: NP-hardness and polynomial-time algorithms, *SIAM J. Computing*, 9 (1980) pp. 558–565.

15. L. Lovász, Submodular functions and convexity. In: *Mathematical Programming: The State of the Art*, Bonn 1982, pp. 235–257, (Springer Verlag, 1983).

16. L. Lovász and M. D. Plummer, Matching theory, North-Holland, Amsterdam, 1986.

17. K. Murota, Matrices and Matroids for Systems Analysis, (Algorithms and Combinatorics, 20), Springer, 1999.

18. J. S. Provan and D. R. Shier, A paradigm for listing (s, t) cuts in graphs, *Algorithmica* **15**, (1996), pp. 351–372.

19. R. C. Read and R. E. Tarjan, Bounds on backtrack algorithms for listing cycles, paths, and spanning trees, *Networks* 5 (1975) 237–252.

20. T. Uno, Algorithms for enumerating all perfect, maximum and maximal matchings in bipartite graphs, *8th International Symposium on Algorithms and Computation, (ISAAC 1997)*, Lecture Notes in Computer Science 1350, pp. 92–101.

21. T. Uno, An Algorithm for Enumerating All Maximal Matchings of a Graph, IPSJ SIGNotes Algorithms Abstract No. 086-007, 2002.