Generic Cospark of a Matrix Can Be Computed in Polynomial Time

Sichen Zhong* and Yue Zhao†*
*Department of Applied Mathematics and Statistics, †Department of Electrical and Computer Engineering
Stony Brook University, Stony Brook, NY, 11794, USA
Emails: {sichen.zhong, yue.zhao.2}@stonybrook.edu

***Abstract*—The cospark of a matrix is the cardinality of the sparsest vector in the column space of the matrix. Computing the cospark of a matrix is well known to be an NP hard problem. Given the sparsity pattern (i.e., the locations of the non-zero entries) of a matrix, if the non-zero entries are drawn from independently distributed continuous probability distributions, it is shown that the cospark equals, with probability one, to a particular number we term the *generic cospark* of the matrix. It is proven that, unlike the cospark, the generic cospark of a matrix can be computed in polynomial time. An efficient algorithm that achieves this is offered.**

I. Introduction

The *cospark* of a matrix $A \in \mathbb{R}^{m\times n}, m > n$[1], denoted by *cospark(A)*, is defined to be the *cardinality of the sparsest vector in the column space of* A [1]. In other words, *cospark(A)* is the optimum value of the following l_0-minimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||Ax||_0, \\ \text{subject to} \quad & x \neq 0, \end{aligned} \tag{1}$$

where $||Ax||_0$ is the number of nonzero elements in the vector Ax. It is well known that solving (1) is an NP-hard problem. Indeed, it is equivalent to computing the *spark* of an orthogonal complement of A [1], where the spark of a matrix is defined to be the *smallest number of linearly dependent columns of it* [2]. Specifically, for A with full column rank, we can find a full rank orthogonal complement $A^{\perp} \in \mathbb{R}^{(m-n)\times m}$, and (1) is equivalent to

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||x||_0, \\ \text{subject to} \quad & A^{\perp}x = 0,\ x \neq 0, \end{aligned} \tag{2}$$

where the optimal value of (2) is the spark of $A^{\perp}$, denoted by *spark($A^{\perp}$)*. Computing spark is known to be NP hard [3].

The role of $cospark(A)$ has been studied in decoding under sparse measurement errors where A is the coding matrix [1]. In particular, $\lfloor \frac{cospark(A)-1}{2} \rfloor$ gives the maximum number of errors that an ideal l_0-minimization decoder can tolerate for exact recovery. Closely related to this is the role of $spark(A^{\perp})$ in characterizing the ability to perform compressed sensing [1] [2]. Spark is also related to notions such as mutual coherence [2] [4] and Restricted Isometry Property (RIP) [1] [5] which provide conditions under which sparse recovery can be performed using l-1 relaxation. Last but not least, in addition to its role in the sparse recovery literature, cospark (1) also plays a central role in security problems in cyber-physical systems (see [6] among others).

In this paper, we study the problem of computing the cospark of a matrix. Although it is proven that (1) is an NP-hard problem, we show that the cospark a matrix "generically" has can in fact be computed in polynomial time. Specifically, given the "sparsity pattern", (i.e., the locations of all the non-zero entries of A,) $cospark(A)$ equals, *with probability one*, to a particular number which we term the *generic cospark* of A, if the non-zero entries of A are drawn from independent continuous probability distributions. We develop an efficient algorithm that computes the generic cospark in *polynomial time*. Due to space limitations, some of the proofs are omitted here, and can be found in [7].

[1]We note that the results in this paper can be straightforwardly generalized to complex numbers.

II. Preliminaries

A. Generic Rank of a Matrix

For a matrix $A \in \mathbb{R}^{m\times n}$, we define its *sparsity pattern* as $S = \{(i,j) | A_{ij} \neq 0, 1 \leq i \leq m, 1 \leq j \leq n\}$. Given a sparsity pattern S, we denote A^S to be the set of all matrices with sparsity pattern S over the field $\mathbb{R}$. Since there is a one to one mapping between S and A^S, we use S and A^S interchangeably to denote a sparsity pattern in the remainder of the paper.

The generic rank of a matrix with sparsity pattern S, denoted by $sprank(A^S)$, is defined as follows.

Definition 1 (Generic Rank)**.** *Given S, the generic rank of A^S is $sprank(A^S) \triangleq \sup_{A\in A^S} rank(A)$.*

Clearly, if $sprank(A^S) < n$, the optimal value of (1) is zero. We will thus focus on the case $sprank(A^S) = n$ for the remainder of the paper.

The following lemma states that the generic rank indeed "generically" equals to the rank of a matrix [8].

Lemma 1. *Given S, $rank(A) = sprank(A^S)$ with probability one, if the non-zero entries of A are drawn from independently distributed continuous probability distributions.*

B. Matching Theory Basics

We now introduce some basics from classical matching theory [9] which are necessary for us to introduce the results in the remainder of the paper.

For a bipartite graph $G(X, Y, E)$, a subset of edges $\mathcal{N} \subseteq E$ is a *matching* if all the edges in $\mathcal{N}$ are vertex disjoint. A *max matching* from X onto Y is a matching with the maximum cardinality. A *perfect matching from X onto Y* is a max matching where every vertex in Y is incident to an edge in the matching.

Consider a (not necessarily maximum) matching $\mathcal{N}$. A vertex is called *matched* if it is incident to some edge in $\mathcal{N}$, and *unmatched* otherwise. An *alternating path* with respect to $\mathcal{N}$ is a path which alternates between using edges in $E\backslash\mathcal{N}$ and edges in $\mathcal{N}$, or vice versa. An *augmenting path* w.r.t $\mathcal{N}$ is an alternating path w.r.t. $\mathcal{N}$ which starts and ends at unmatched vertices. With an augmenting path P, it can be easily shown that the symmetric difference[2] $\mathcal{N} \oplus P$ gives a matching with size $|\mathcal{N}| + 1$.

C. Generic Rank as Max Matching

We now introduce an equivalent definition of generic rank via matching theory. A sparsity pattern A^S can be represented as a *bipartite graph* as follows [8]. Let $G(X, Y, E)$ be a bipartite graph whose a) vertices $X = \{1, 2, \ldots, m\}$ correspond to all the row indices of A^S, b) vertices $Y = \{1, 2, \ldots, n\}$ correspond to all the column indices of A^S, and c) edges in $E = S$ correspond to all the non-zero entries of A^S. Accordingly, we also denote the bipartite graph for sparsity pattern S as $G(X, Y, S)$.

The following lemma states the equality between $sprank(A^S)$ and the max matching on $G(X, Y, S)$ [8].

Lemma 2. *Given $G(X, Y, S)$, the generic rank $sprank(A^S)$ equals to the cardinality of the maximum bipartite matching on G.*

Accordingly, finding a max matching on this graph using the Hopcroft-Karp algorithm allows us to find the generic rank with complexity $\mathcal{O}(|S|\sqrt{m+n})$ [10].

III. Generic Cospark

Similarly to the supremum definition of generic rank (cf. Definition 1), given the sparsity pattern of a matrix, we define *generic cospark* as follows.

Definition 2 (Generic Cospark)**.** *Given S, the generic cospark of A^S is $spcospark(A^S) \triangleq \sup_{A \in A^S} cospark(A)$.*

In a spirit similar to the multiple interpretations of generic rank in Section II, we provide a *probabilistic* view and a *matching theory* based view of generic cospark as follows.

A. Cospark Equals to Generic Cospark With Probability One

For any $T \subset [m]$, let A_T and A_T^S represent the matrix A and the set of matrices A^S restricted to the rows T respectively. A class of matrices which has cospark equal to generic cospark are those which satisfy the following property:

[2]The *symmetric difference* of two sets S_1 and S_2 is defined as $S_1 \oplus S_2 = (S_1 \cup S_2) \setminus (S_1 \cap S_2)$.

Lemma 3. *Given any sparsity pattern S so that $sprank(A^S) = n$ for $A^S \subset \mathbb{R}^{m\times n}$, for any $A \in A^S$, if $rank(A_T) = sprank(A_T^S)$, $\forall T \subseteq [m]$, then $cospark(A) = spcospark(A^S)$.*

Proof. Suppose A satisfies the condition in the lemma. Let $x^* = \operatorname{argmin}_{x\neq 0} ||Ax||_0$, and let $U = \{i | a_i x^* = 0\}$, where a_i is the i^{th} row of A. Since $A_U x^* = 0$, $rank(A_U) < n$. Now consider any other matrix $C \in \mathbb{R}^{m\times n}$ with sparsity pattern S. Since $rank(C_U) \leq rank(A_U) = sprank(A_U^S) < n$, $\ker(C_U)$ is also nonempty, meaning there exists a nonzero vector $h \in \mathbb{R}^n$ such that $C_U h = 0$. Because $A_{U^c}x^*$ has no zero entries, we also have $||C_{U^c}h||_0 \leq ||A_{U^c}x^*||_0 = ||Ax^*||_0$. This means $||Ch||_0 = ||C_U h||_0 + ||C_{U^c}h||_0 \leq ||A_{U^c}x^*||_0 = ||Ax^*||_0$. Hence, if $\hat{x} = \operatorname{argmin}_{x\neq 0} ||Cx||_0$, it follows $cospark(C) = ||C\hat{x}||_0 \leq ||Ch||_0 \leq ||Ax^*||_0 = cospark(A)$. Thus, $spcospark(A^S) = \sup_{C\in A^S} cospark(C) \leq cospark(A)$, which implies $cospark(A) = spcospark(A^S)$. ☐

The property $rank(A_T) = sprank(A_T^S), \forall T \subseteq [m]$ is known as the *matching property* of matrix A according to [11].

Now, we have the following theorem showing that the generic cospark indeed "generically" equals to the cospark.

Theorem 1. *Given S, $cospark(A) = spcospark(A^S)$ with probability one, if the non-zero entries of A are drawn from independently distributed continuous probability distributions.*

Proof. If we have a matrix A with sparsity pattern S whose nonzeros are drawn from independent continuous distributions, then every submatrix of rows has rank equaling generic rank w. p. 1 (cf. Lemma 1). This immediately implies $cospark(A) = spcospark(A^S)$ w. p. 1 by Lemma 3. ☐

B. A Matching Theory based Definition of Generic Cospark

Let $G(X, Y, S)$ be the bipartite graph corresponding to $A^S \subseteq \mathbb{R}^{m\times n}$. For a subset of vertices $Z \subseteq X$, we define the induced subgraph $G(Z)$ as the bipartite graph $G(Z, N(Z), \{(i, j) \in S | i \in Z\ , j \in N(Z)\})$, where $N(Z)$ denotes the vertices in Y adjacent to the set Z. In essence, $G(Z)$ is the bipartite graph corresponding to submatrices A_Z^S. We then have the following.

Lemma 4. *Given $G(X, Y, S)$, let $OPT \subset X$ be a largest subset such that the induced subgraph $G(OPT)$ has a max matching of size $n-1$. Then $spcospark(A^S) = m - |OPT|$.*

The intuition behind this matching theory based definition of *$spcospark(A^S)$* is the following. To find the sparsest vector in the image of A, it is equivalent to find a largest set of rows in A, OPT, which span an $n-1$ dimensional subspace. With such a subset OPT, we can find a vector x^* which satisfies $A_{OPT}x^* = 0$, and it is clear that $x^* \in \operatorname{argmin}_{x\neq 0} ||Ax||_0$. Furthermore, based on the equivalence between generic rank and max matching from Lemma 2, we arrive at the matching theory based definition of generic cospark in Lemma 4.

C. An Illustrative Example

We now present an example $\tilde{A}^S$ in Figure 1, which we will refer to throughout the paper for illustrating proof concepts

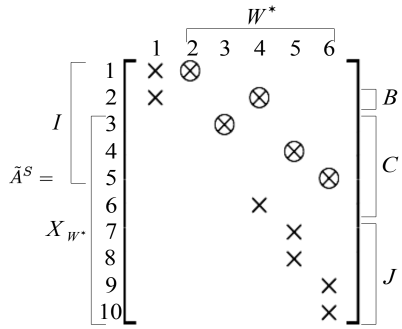

Fig. 1. Entries with $\times$ (either circled or not) represent non-zero entries, while entries with $\otimes$ also represent edges in $\mathcal{M}$. The three entries with $\otimes$ in rows in $\mathcal{C} = \{3, 4, 5, 6\}$ represent edges in set $\mathcal{M}_p$. An output for Algorithm 1 is $X_f = X_{W^*} \cup B = \{2, 3, 4, 5, 6, 7, 8, 9, 10\}$.

and our algorithms. In this example, $X = \{1, 2, ..., 10\}$, $Y = \{1, 2, ..., 6\}$, $\times$ (either circled or not) represents a non-zero entry, and the empty entries are zero. The bipartite graph representation of $\tilde{A}^S$ is given in Figure 2. A set of rows OPT equals $\{1, 2, 3, 4, 5, 7, 8, 9, 10\}$. It can be verified that $sprank\,(G(OPT)) = 5$. Hence, the nullspace of $\tilde{A}_{OPT}$ is nonempty, and there exists a nonzero x such that $\tilde{A}_{OPT}x = 0$. As a result, $spcospark(\tilde{A}^S) = 10-9 = 1$. We note that OPT may (often) not be unique.

IV. Efficient Algorithm for Computing Generic Cospark

In this section, we introduce an efficient algorithm that computes the generic cospark in polynomial time. This algorithm is motivated by Lemma 4.

Given $G(X, Y, S)$, for any size $n-1$ subset of vertices $W \subset Y$, we define $X_W = \{x \in X | N(x) \subseteq W\}$. In other words, X_W is the index set of rows of A^S with a *zero* entry in the remaining coordinate $v = Y \setminus W$. For example, in $\tilde{A}^S$ (cf. Figure 2), with $W = W^* = \{2, 3, 4, 5, 6\} \subset Y$, we have that $X_{W^*} = \{3, 4, 5, 6, 7, 8, 9, 10\}$.

We use X_W as a basis to construct a candidate solution for OPT. The idea is to add a maximal subset of vertices $B \subset X_W^c$ to X_W, such that $\overline{X_W} = X_W \cup B$ has a matching of size $n-1$ onto Y. Specifically, we keep adding vertices $t \in X_W^c$ to B as long as the submatrix corresponding to the index set $X_W \cup B$ has generic rank no greater than $n-1$. In the example of $\tilde{A}^S$ (cf. Figure 2), $\tilde{A}^S_{X_{W^*}}$ has generic rank 4, which is indicated by a max matching $\{(3, 3), (4, 5), (5, 6), (6, 4)\}$. In this example, it can be verified that a) if $B = \{2\}$, then $sprank(\tilde{A}^S_{X_{W^*} \cup B}) = 5$, and b) the entire matrix $\tilde{A}^S$ has a generic rank of 6, and thus B cannot be $\{1, 2\}$.

The following lemma shows adding a vertex to B can only increase the generic rank of $A^S_{X_W \cup B}$ by *at most one*.

Lemma 5. *Given $G(X, Y, S)$, $\forall Z \subset X$ and $u \in X \setminus Z$, $sprank(A^S_{Z \cup \{u\}}) \leq sprank(A^S_Z) + 1$.*

Remark 1. *For a given W, depending on the order we visit the vertices in X_W^c, we could end up with different sets B, possibly of different sizes. However, we will prove that an optimal solution is recovered regardless.*

$\overline{X_W}, \forall W$ are the candidate solutions for OPT, and we obtain the optimal solution by choosing the $\overline{X_W}$ with the largest cardinality, i.e., $X_f = \mathrm{argmax}_{W \subset Y, |W| = n-1} |\overline{X_W}|$. The generic cospark of A^S then equals to $m - |X_f|$. The detailed algorithm is presented in Algorithm 1.

Algorithm 1 Computing Generic Cospark

procedure SPCOSPARK(A^S)
    Initialization: Set $B = \emptyset, t = \emptyset$, and $X_f = \emptyset$
    for all $W \subset Y$ of cardinality $n-1$ **do**
        Scan through all m vertices in X to find X_W and let $T = X_W^c$
        Calculate $sprank(A^S_{X_W})$
        while $sprank(A^S_{X_W \cup B \cup \{t\}}) \leq n-1$ **do**
            Let $B = B \cup t$
            Choose any element t from T, and set $T = T \setminus t$
        end while and let $\overline{X_W} = X_W \cup B$
        if $|X_f| < |\overline{X_W}|$ **then**
            Set $X_f = \overline{X_W}$
        end if
        Set $B = \emptyset$
    end for
    Return X_f, and $spcospark(A^S) = m - |X_f|$.
end procedure

V. Proof of Optimality of Algorithm 1

In this section, we prove that Algorithm 1 indeed solves the generic cospark. It is sufficient to prove that the set X_f returned by Algorithm 1 satisfies the definition of OPT in Lemma 4, i.e., X_f is a subset of vertices of the largest size such that the induced subgraph $G(X_f)$ has a max matching of size $n-1$. Since $G(X_f)$ by construction has a max matching of size $n-1$, *it is sufficient to prove that X_f has the largest size, i.e., $|X_f| = |OPT|$.*

To prove this, let us consider an optimal set $OPT \subset X$. We denote by $\mathcal{M}$ the set of $n-1$ edges in a max matching of $G(OPT)$. We denote by $W^* \subset Y$ the set of $n-1$ vertices in Y incident to edges in $\mathcal{M}$, and denote by $v = Y \setminus W^*$ the remaining vertex in Y. We show that, starting with W^*, Algorithm 1 returns an X_f such that $|X_f| \geq |OPT|$, and hence $|X_f| = |OPT|$. We note that the returned X_f may not be the same as OPT.

As the notations for this section are quite involved, we illustrate them with the example $\tilde{A}^S$ in Figure 2 to help clarify the proof. In this example, an option of OPT is $\{1, 2, 3, 4, 5, 7, 8, 9, 10\}$, and accordingly $W^* = \{2, 3, 4, 5, 6\}$. The thick edges in Figure 2 and the entries with $\otimes$ in Figure 1 represent edges in $\mathcal{M}$.

We first partition OPT into $OPT = \mathcal{I} \cup \mathcal{J}$, $\mathcal{I} \cap \mathcal{J} = \emptyset$, where $\mathcal{I}$ is the set of $n-1$ vertices in OPT incident to edges in $\mathcal{M}$. $\mathcal{J}$ consists of the remaining vertices in OPT not incident to edges in $\mathcal{M}$ (cf. Figure 1 and 2).

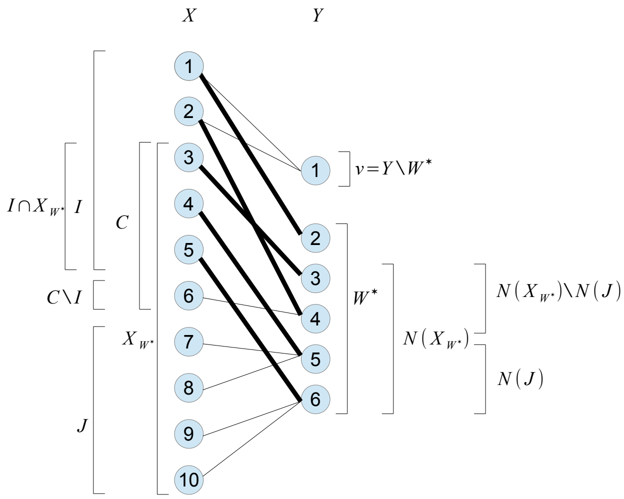

Fig. 2. Bipartite graph representation of $\tilde{A}^S$. The regular black line segments are unmatched edges in the bipartite graph. The thick line segments comprise $\mathcal{M}$, which forms a $n-1$ matching from $\mathcal{I}$ to W^*.

WLOG, we assume $\mathcal{J}$ is nonempty. This is because, if $\mathcal{J}$ is empty, we immediately have $|OPT| = n-1 \le |X_f|$ since each $A_{\overline{X_W}}$ has generic rank $n-1$. We then have the following lemma about X_{W^*} and sets $\mathcal{I}$, $\mathcal{J}$.

Lemma 6. *For any such partition $OPT = \mathcal{I} \cup \mathcal{J}$, we have $\mathcal{J} \subset X_{W^*}$, and $\mathcal{I} \cap X_{W^*}$ is nonempty.*

Accordingly, we can partition $X_{W^*} = \mathcal{C} \cup \mathcal{J}$, $\mathcal{C} \cap \mathcal{J} = \emptyset$, with $\mathcal{C} \triangleq X_{W^*} \setminus \mathcal{J}$. Starting from here, the general idea of proving $|X_f| \ge |OPT|$ is to lower bound

$$|X_f| = |X_{W^*} \cup B| = |\mathcal{C} \cup \mathcal{J} \cup B| = |\mathcal{C}| + |\mathcal{J}| + |B|. \quad (3)$$

We immediately have the following lower bound on $|B|$:

$$|B| \ge (n-1) - sprank(A^S_{X_{W^*}}). \quad (4)$$

This is because a) Algorithm 1 guarantees $sprank(A^S_{W^* \cup B}) = n-1$, and b) every time we add a new vertex t into B, $sprank(A^S_{W^* \cup B})$ increases by at most one (cf. Lemma 5). Since the initial generic cospark is $sprank(A^S_{X_{W^*}})$, we need at least $(n-1) - sprank(A^S_{X_{W^*}})$ vertices added into B to reach $sprank(A^S_{W^* \cup B}) = n-1$.

We next devote the majority of this section to provide a lower bound on $|\mathcal{C}|$.

A. Lower Bounding $|\mathcal{C}|$

The key result we will rely on in this subsection is the following:

Theorem 2. *For the induced bipartite graph $G(X_{W^*})$, there exists a max matching whose edges are not incident to any vertices in $\mathcal{J}$.*

To prove Theorem 2, we use Lemma 6 and start with a partial matching $\mathcal{M}_p \subset \mathcal{M}$ consisting of only edges which are incident to vertices in $\mathcal{I} \cap X_{W^*}$. In other words, $\mathcal{M}_p = \{(i,j) \in \mathcal{M} | i \in \mathcal{I} \cap X_{W^*}\}$. The idea is that we will build a max matching starting from $\mathcal{M}_p$, and this max matching will not touch any vertices in $\mathcal{J}$, thus proving Theorem 2.

The proof of Theorem 2 relies on the following two lemmas.

Lemma 7. *For the induced bipartite graph $G(X_{W^*})$ with partial matching $\mathcal{M}_p$, every vertex in $N(\mathcal{J})$ is incident to some edge in $\mathcal{M}_p$, i.e., already matched.*

Lemma 8. *For the induced bipartite graph $G(X_{W^*})$ with partial matching $\mathcal{M}_p$, there exists no augmenting path starting from any $j \in \mathcal{J}$.*

In the example of $\tilde{A}^S$, $\mathcal{M}_p = \{(3,3),(4,5),(5,6)\}$. We see that every node in $N(\mathcal{J}) = \{5,6\} \subset Y$ is indeed incident to some edge in $\mathcal{M}_p$. Lemma 8 implies that all augmenting paths w. r. t. the partial matching $\mathcal{M}_p$ are from unmatched vertices in $\mathcal{C} \setminus \mathcal{I}$ (where $\mathcal{C} = X_{W^*} \setminus \mathcal{J}$) to unmatched vertices in $N(X_{W^*}) \setminus N(\mathcal{J})$.

A corollary which will prove necessary in proving Theorem 2 is the following:

Corollary 1. *Suppose P is an augmenting path from $c \in \mathcal{C} \setminus \mathcal{I}$ to $u \in N(X_{W^*}) \setminus N(\mathcal{J})$ w. r. t. the matching $\mathcal{M}_p$. Then for any $j \in \mathcal{J}$, there exists no alternating paths w. r. t. $\mathcal{M}_p$ from j to any vertex in P.*

Proof. Let P be an augmenting path from c to u w.r.t. $\mathcal{M}_p$. Suppose there exists an alternating path P'_{jp} from j to a vertex p, where p is the first vertex in P encountered when traversing P'_{jp}. P'_{jp} must have odd number of edges, since p is a matched vertex in P and j is unmatched. Since P'_{jp} is odd, $p \in N(X_{W^*})$. Hence, if $P_{cp} \subset P$ is the restriction of P from c to p, then the alternating path P_{cp} must also have odd length. The total length of P must be odd since P is an augmenting path, which means the length of the alternating path from p to u in P must be even.

Since P_{jp} is an odd alternating path from j to p, and the alternating path from p to u in P is even, then the alternating path from P_{jp} to u is odd. Furthermore, j and u are unmatched, so this path is actually an augmenting path, which immediately contradicts Lemma 8. $\square$

From Corollary 1, any alternating path starting from j w. r. t. $\mathcal{M}_p$ is *vertex disjoint* to any augmenting path P. This implies that a) any alternating path from j w. r. t. $\mathcal{M}_p \oplus P$ remains an alternating path, and b) there remains no augmenting path starting from j w. r. t. $\mathcal{M}_p \oplus P$, i.e., *Lemma 8 continues to hold for $G(X_{W^*})$ with a new matching $\mathcal{M}_p \oplus P$.*

We are now ready to prove Theorem 2.

Proof of Theorem 2. Take $\mathcal{M}_p$ to be an initial matching onto $N(X_{W^*})$. By Lemma 7, all vertices in $N(\mathcal{J})$ are now matched, and Lemma 8 tells us we are left with augmenting paths starting from unmatched vertices in $\mathcal{C} \setminus \mathcal{I}$ to unmatched vertices in $N(X_{W^*}) \setminus N(\mathcal{J})$. If P_1 is one such augmenting path, then $\mathcal{M}_p \oplus P_1$ is a matching with one greater cardinality. By Corollary 1, all alternating paths w.r.t $\mathcal{M}_p$ starting from j are vertex disjoint to P_1, which implies alternating paths starting from j remain unchanged. Furthermore, Corollary 1 tells us $\mathcal{M}_p \oplus P_1$ does not have augmenting paths starting from j. Hence, the only remaining augmenting paths are still from vertices $\mathcal{C} \setminus \mathcal{I}$ to vertices $N(X_{W^*}) \setminus N(\mathcal{J})$. If P_2 is such an augmenting path, we can now repeat the above procedure and compute the matching $\mathcal{M}_p \oplus P_1 \oplus P_2$. Again, alternating paths starting from j remain unchanged, and $\mathcal{M}_p \oplus P_1 \oplus P_2$ contains no augmenting paths starting from j. We can repeat this procedure until all augmenting paths from $\mathcal{C} \setminus \mathcal{I}$ to $N(X_{W^*}) \setminus N(\mathcal{J})$ are eliminated. Since the final matching obtained this way has no augmenting paths, this final matching is optimal, and its edges are incident to no vertices in $\mathcal{J}$. ☐

As a result of Theorem 2, there exists a max matching on $G(X_{W^*})$ that, on the "left hand side" of the graph, only touches vertices in $\mathcal{C} = X_{W^*} \setminus \mathcal{J}$. Since the size of the max matching of $G(X_{W^*})$ equals to $sprank\left(A^S_{X_{W^*}}\right)$ (cf. Lemma 2), we arrive at the following lower bound on $|\mathcal{C}|$:

$$|\mathcal{C}| \geq sprank\left(A^S_{X_{W^*}}\right) \tag{5}$$

B. Proof of the Optimality of Algorithm 1

We now show Algorithm 1 indeed returns the generic cospark as in the following theorem.

Theorem 3. *The output, X_f, of Algorithm 1 satisfies $|X_f| = |OPT|$.*

Proof. By the definition of OPT, $|X_f| \leq |OPT|$. To prove $|X_f| \geq |OPT|$, starting from (3),

$$\begin{aligned}
|X_f| &= |\mathcal{C}| + |\mathcal{J}| + |B| && (6)\\
&\geq sprank(A^S_{X_{W^*}}) + |\mathcal{J}| + |B| && (7)\\
&\geq sprank(A^S_{X_{W^*}}) + |\mathcal{J}| + (n-1) - sprank(A^S_{X_{W^*}}) && (8)\\
&= |\mathcal{J}| + (n-1) = |\mathcal{I}| + |\mathcal{J}| = |OPT| && (9)
\end{aligned}$$

where (7) is from (5), and (8) is from (4). ☐

Remark 2. *The output X_f of Algorithm 1 may not be the same as the original OPT, although $|X_f| = |OPT|$ always. In the example of $\tilde{A}^S$, $X_f = \{2,3,4,5,6,7,8,9,10\}$, whereas the OPT (corresponding to $W^* = \{2,3,4,5,6\}$) at the start of Algorithm 1 equals $\{1,2,3,4,5,7,8,9,10\}$.*

VI. Algorithm Complexity

We now show that Algorithm 1 is efficient, and provide an upper bound on its computational complexity.

Theorem 4. *Given any S, Algorithm 1 computes $spcospark(A^S)$ in $\mathcal{O}(nm(1+|S|))$ time.*

Proof. Observe in the pseudocode above, step 3 is over n iterations. For each iteration, steps 4 to 9 are the most computationally expensive. Step 4 requires a $\mathcal{O}(m)$ scan of the rows of A^S, and step 5 requires us to compute a perfect matching using Hopcroft-Karp algorithm, which can be done in $\mathcal{O}(|S|\sqrt{m+n})$ time.

For the loop in steps 6 to 9, we do not need to recalculate $sprank(A^S_{X_W \cup B})$ every iteration. Given that we know the max matching from the previous iteration, we only need to check if the new vertex t added to B has an augmenting path to an unmatched vertex in Y. Searching for this augmenting path requires us to use breadth first search (BFS) or depth first search (DFS), which can be computed in $\mathcal{O}(|S|)$ time. Since there are $\mathcal{O}(m)$ iterations in the while loop, the total cost of steps 6 to 9 is $\mathcal{O}(m|S|)$.

Hence, for every iteration of step 3, the total cost is $\mathcal{O}(m + |S|\sqrt{m+n} + m|S|) = \mathcal{O}(m(1+|S|))$ since $n \leq m$. It follows immediately our total running time is $\mathcal{O}(nm(1+|S|))$. ☐

From Theorem 4, if A^S is very sparse, the running time of Algorithm 1 is essentially *quadratic*.

Remark 3. *The algorithm's bottleneck is in steps 6-9. For each row t to add, we need to use a BFS. Since we need to add $\mathcal{O}(m)$ such vertices, the total complexity for these steps is $\mathcal{O}(m|S|)$ as in the above proof. To improve this complexity, we would like to detect multiple candidate rows to add to B using a single BFS. Indeed, it can be shown further that steps 6-9 of Algorithm 1 can be improved to $\mathcal{O}(\sqrt{m}|S|)$ based on an idea similar to Hopcroft-Karp matching [10]. This will improve the total running time of Algorithm 1 to $\mathcal{O}(n\sqrt{m}|S|)$.*

VII. Conclusion

Given any sparsity pattern of a matrix, the cospark of the matrix is always upper bounded by the generic cospark, and is equal to the generic cospark with probability one if the nonzero entries of the matrix are drawn from independent continuous probability distributions. We have shown that, although computing the cospark of a matrix is NP hard, the generic cospark can be computed in polynomial time. We have developed an efficient algorithm that achieves this.

References

[1] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[2] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via l-1 minimization," *Proceedings of the National Academy of Sciences*, vol. 100, no. 5, pp. 2197–2202, 2003.

[3] A. M. Tillmann and M. E. Pfetsch, "The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing," *IEEE Transactions on Information Theory*, vol. 60, no. 2, pp. 1248–1259, 2014.

[4] R. Gribonval and M. Nielsen, "Sparse representations in unions of bases," *IEEE Transactions on Information Theory*, vol. 49, no. 12, pp. 3320–3325, 2003.

[5] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[6] Y. Zhao, A. Goldsmith, and H. V. Poor, "Minimum sparsity of unobservable power network attacks," *IEEE Transactions on Automatic Control, to appear*.

[7] S. Zhong and Y. Zhao, "Generic cospark of a matrix can be computed in polynomial time," *arXiv preprint arXiv:1701.08925*, 2017.

[8] K. Reinschke, *Multivariable Control - A Graph-Theoretic Approach*. New York: Springer-Verlag, Lecture Notes in Control and Information Sciences, vol. 108, 1988.

[9] R. Diestel, D. Král, and P. Seymour, "Graph theory," *Oberwolfach Reports*, vol. 13, no. 1, pp. 51–86, 2016.

[10] J. E. Hopcroft and R. M. Karp, "An n^5/2 algorithm for maximum matchings in bipartite graphs," *SIAM Journal on computing*, vol. 2, no. 4, pp. 225–231, 1973.

[11] S. T. McCormick, "A combinatorial approach to some sparse matrix problems." *DTIC Document*, pp. 39–41, 1983.