Index coding with outerplanar side information

Yossi Berliner * Michael Langberg †

Abstract

We study the Index Coding problem with side information graphs which are *outerplanar*. For general side information graphs, linearly solving the Index Coding problem implies a linear solution to the general (non-multicast) Network Coding problem — a central open problem in the field of network communication. For outerplanar side information graphs, we show that the Index Coding problem can be solved efficiently, and characterize its solution in terms of the *clique cover* size of the information graph at hand.

1 Introduction

The *Index Coding* problem is a fundamental non-multicast network coding problem. The problem has a very clean and elegant structure yet it captures many important aspects of the more general network coding problem. The Index Coding problem was introduced by Birk and Kol [4] in the context of data dissemination to clients with local caches. An instance of the Index Coding problem includes a sender node r, a set $C = \{c_1, \ldots, c_n\}$ of wireless clients and a set $P = \{p_1, p_2, \ldots, p_n\}$ of n independent messages that belong to some alphabet Σ. The message p_i needs to be delivered to client c_i. Each client c_i is assumed to hold certain *side information* $\Gamma_i \subseteq P$. It is common to specify the side information by a graph $G = (V, E)$ with vertex set $V = \{1, \ldots, n\}$ in which vertex i corresponds to client c_i and edge $(i, j) \in E$ iff $j \in \Gamma_i$.

In each *round* of communication the sender can transmit a single symbol of Σ (i.e., a single message). We assume that all symbols transmitted by the sender are received by all clients without error. The j'th round of communication is specified by an encoding function $g_j : \Sigma^n \to \Sigma$. The objective is to find a set of encoding functions $\Phi = \{g_i\}_{i=1}^{\ell}$ that will allow client c_i to decode the messages p_i it requires while minimizing the number of transmissions $\ell = |\Phi|$. Client $c_i \in C$ can decode packet p_i if there exists a decoding function $\gamma_i : \Sigma^{\ell} \times \Sigma^{|\Gamma_i|} \to \Sigma$ that allows c_i to obtain p_i from the ℓ characters transmitted by the sender and the $|\Gamma_i|$ characters of side information. The minimum value of ℓ is defined to be the *round complexity* of the index coding instance at hand (in cases in which the alphabet Σ is specified we denote the round complexity as ℓ_{Σ}).

In this work we focus on information graphs G which are *undirected* (namely $i \in \Gamma_j$ iff $j \in \Gamma_i$), and on the case in which the encoding functions are linear (or actually *scalar* linear by common terminology). In this case we also take Σ to be a finite field. We note that one can extend the definition of the Index Coding problem to directed side information graphs, to encoding functions which are not scalar linear but rather *vector linear* (via time sharing) or non-linear, and to clients c_i which require not a single message but multiple ones. We touch on these extensions in the conclusion of this work.

There exist beautiful connections between combinatorial properties of the undirected side information graph G and the optimal solution to the corresponding Index Coding problem. In [3] it is shown that the problem of finding (scalar) linear solutions to the Index Coding problem is equivalent to the problem of minimizing the rank of a certain matrix with "don't care" entries. The latter problem, referred to as the `MinRank` problem (denoted by $\texttt{MR}_{\Sigma}$), has been investigated by Haemers and Peeters [7, 12] in which it is shown that the optimal solution value of

*The Open University of Israel, Raanana, 43107, Israel. email: yossi_berliner@radwin.com

†The Open University of Israel, Raanana, 43107, Israel. email: mikel@openu.ac.il

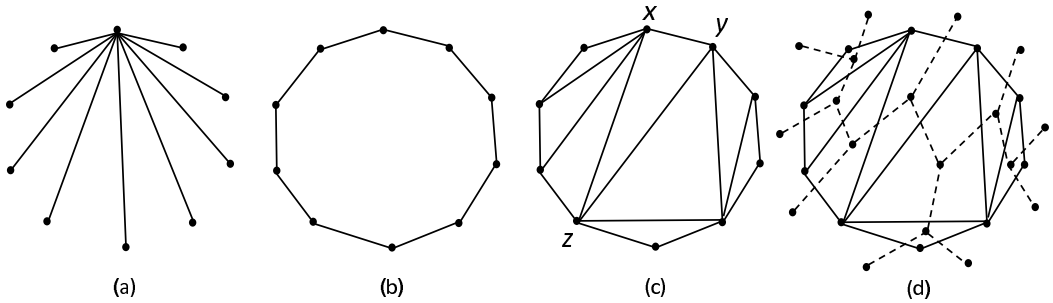

Figure 1: Examples of outerplanar graphs. (a) The "star" graph. (b) The "ring" graph. (c) A general outerplanar graph. The vertices x, y and z illustrate the proof of Claim 4.3. (d) The outerplanar graph G presented in (c) with its corresponding tree representation $\bar{G}$ (with dotted edges).

$\mathtt{MR}_\Sigma(G)$ is "sandwiched" between the maximum sized independent set in G (denoted by $\alpha(G)$) and the minimum sized clique cover of G (denoted by $\mathtt{CC}(G)$). Here, an independent set in G is a subset of vertices that do not share any edges, a clique in G is a subset of vertices for which every pair share an edge, and a clique cover of G is a collection of cliques in G such that every vertex in G appears in at least a single clique in the collection. Namely, for an field Σ and an instance to the index coding problem defined by an undirected graph G, the scalar linear round complexity $\ell_\Sigma^{(lin)}$ is exactly $\mathtt{MR}_\Sigma(G)$ which in turn satisfies $\mathtt{CC}(G) \geq \mathtt{MR}_\Sigma(G) \geq \alpha(G)$.

The authors of [3] study the Index Coding problem in the setting of (scalar) linear encoding functions. Lubetzky et al. [11] show that non-linear codes can significantly outperform linear codes for certain families of problem instances. El Rouayheb et al. [6] show that any instance of the more general network coding problem can be efficiently reduced to an instance of the Index Coding problem (under the assumption of linear encodings). This reduction implies that efficiently finding optimal linear solutions for the Index coding problem implies an efficient solution to the general (non-multicast) network coding problem — the latter being a central open problem in network communication. In the scalar linear setting, this implies that it is NP-hard to solve the index coding problem (via the hardness results of [10] on scalar linear network coding). Such hardness results were also shown independently in [12]. The hardness of finding approximate solutions for the Index Coding problem has been studied in [9].

1.1 Our contribution

As it is NP-hard to efficiently find scalar linear solutions to the Index Coding problem for arbitrary side information graphs G, in this work we address the following natural question: *On which families of side information graphs can Index Coding be solved efficiently?*

Our work takes a modest step in better understanding this question. Namely, in our study, we investigate "simple nature" side information graphs G that are so-called *outerplanar* — i.e., graphs G that can be drawn in the plane such that (a) all vertices of G lie on the boundary of a circle, and (b) representing edges of G by straight lines between their corresponding vertices, no two edges of G intersect. Some examples of outerplanar graphs are given in Figure 1.

We show that on outerplaner side informations graphs G, the Index Coding problem can be solved efficiently and can be characterized by the clique cover number of G. The study of more complicated side information graphs, such as k-outerplanar graphs[1] or planar graphs, is left open in this work. The main contribution of this work can

[1] A k-outerplanar graph can be embedded in the plane with no edge intersections in such a way that k consecutive removals of the outer

be summarized by the following theorem.

Theorem 1.1. *Let Σ be any finite field. The optimal scalar linear solution to the Index Coding problem over Σ with outer planar side information graphs G can be found efficiently (i.e., in time which is polynomial in $|G|$). In this case, the scalar linear round complexity $\ell_{\Sigma}^{(lin)} = \mathtt{MR}_{\Sigma}(G)$ of the optimal scalar linear index coding solution is equal to the clique cover size $\mathtt{CC}(G)$ of G.*

Another natural question that motivated our work addresses the relationship between the index coding round complexity $\ell_{\Sigma}^{lin} = \mathtt{MR}_{\Sigma}(G)$ and its combinatorial upper and lower bounds — the clique cover size $\mathtt{CC}(G)$ and the independent set size $\alpha(G)$. It is not hard to find graphs G for which $\ell_{\Sigma}^{lin}(G)$ is strictly greater than $\alpha(G)$, for example the five cycle C_5 satisfies $\ell_{\Sigma}^{lin}(C_5) = 3$ while $\alpha(C_5) = 2$. However, it is significantly more difficult to find graphs G for which $\mathtt{CC}(G)$ is strictly greater than $\ell_{\Sigma}^{lin}(G)$.

Roughly speaking, the index coding solutions that correspond to clique covers of G are extremely simple in nature. Specifically, for each clique C in the cover one defines an encoding function g which equals the sum of messages p_i corresponding the vertices in the clique C. Hence asking whether for a certain G it holds that $\mathtt{CC}(G) > \ell_{\Sigma}^{lin}(G)$ is equivalent to asking whether for a certain side information graph G there is an index coding solution which is better than the trivial one. Such graph (of rather complicated nature) are known to exist, e.g., [7, 12]. However, in an attempt to construct simpler families of graphs G for which $\mathtt{CC}(G) > \ell_{\Sigma}^{lin}(G)$, it is natural to ask: *For which family of side information graphs G is it the case that $\ell_{\Sigma}^{lin}(G) = \mathtt{CC}(G)$?* This work takes a small step in better understanding this question.

1.2 Proof techniques

Many NP-hard graph problems become easier to solve on outerplanar graphs. One general technique for coping with outerplanarity (and other simple graph structures) is that of Baker [2]. Baker's approach is based on a decomposition of the given graph into a tree structure that supports a dynamic programming approach to the problem at hand. Based on this paradigm, Baker [2] presents efficient algorithms for solving several NP complete problems on outerplanar graphs. These include for example the maximum independent set (and minimum vertex cover), minimum dominating set, and minimum edge-dominating set problems. Baker's technique generalize to k-outerplanar graphs. Baker shows that efficient solutions to k-outerplanar graphs yield high quality approximation algorithms[2] for planar graphs.

The clique cover problem on planar graph is known to be NP-complete [5, 8]. To the best of our knowledge this problem has not been addressed in the context of outerplanar or k-outerplanar graphs. In this work, we extend the ideas of Baker to show that one can solve clique cover on outerplanar graphs efficiently. Our extension involves a delicate analysis of the dynamic programming resulting from the tree structure suggested in [2], and does not follow directly from [2].

We then turn to tie the clique cover size of a given outerplanar graph to its `MinRank` (or equivalently, its scalar linear index coding round complexity $\ell_{\Sigma}^{(lin)}$). As mentioned above, the `MinRank` of any graph is at most its clique cover. In this work we show that for outerplanar graphs the clique cover size and `MinRank` value are actually equal. Our proof follows the dynamic nature of Baker's algorithm. We start by showing how to apply Baker's paradigm to the problem of computing the `MinRank` of a given graph G. This proves that the `MinRank` problem on outerplanar graphs can be solved efficiently. As in the case of clique cover, applying the work of [2] to the `MinRank` problem involves a delicate analysis that does not follow directly from [2].

To tie the clique cover number with the `MinRank` of G, we show that throughout the execution of Baker's algorithm - no matter what intermediate objective is being considered (the clique cover size or the `MinRank`) the

face of the embedding result in the empty graph.

[2] An r approximation algorithm is one which returns a solution whose value is within an r multiplicative factor of the optimal solution value.

value returned in the intermediate step is identical. This implies that the $\texttt{MinRank}$ of G equals its clique cover in the outerplanar scenario.

Our proof techniques hold for outerplanar graphs when one computes $\texttt{MinRank}$ over any finite Σ. It is natural to ask what happens in the k-outerplanar case. Such generalizations are left open in this work and seemingly cannot be addressed by the current proof techniques.

1.3 Structure

Our paper is structured as follows. In Section 2, we present some definitions and notation together with some preliminary properties of $\texttt{MR}_\Sigma$ and $\texttt{CC}$ that we use throughout the paper. In Section 3, we present a rough overview of Baker's algorithmic paradigm [2] for outerplanar graphs and specify the main points that need to be addressed in order to apply the algorithm to the clique cover and $\texttt{MinRank}$ problems. In Section 4 we address the points specified in Section 3 for the $\texttt{MinRank}$ problem. In Section 5 we address the points specified in Section 3 for the clique cover problem. In Section 6 we show the connection between clique cover and $\texttt{MinRank}$.

2 Preliminaries

Definition 2.1 ($\texttt{MR}_\Sigma(G)$)**.** *Let Σ be a finite field. We say that a matrix $A = a_{ij}$ fits an undirected graph G if for all i and j: $a_{ii} = 1$, and $a_{ij} = 0$ whenever (i, j) is not an edge of G.* $\texttt{MR}_\Sigma(G) \equiv \min\{\textbf{rank}_\Sigma(A) \mid A \text{ fits } G\}$

Definition 2.2 ($\texttt{CC}(G)$)**.** *A clique in an undirected graph G is a subset of vertices for which every pair share an edge. A clique cover of G is a collection of cliques in G such that every vertex in G appears in at least a single clique in the collection.* $\texttt{CC}(G)$ *is the size of the minimum clique cover in G.*

As described in the Introduction, finding the optimal (scalar) linear solution to an index coding instance with undirected side information graph G is equivalent to determining the matrix A for which $\texttt{MinRank}(G) = \textbf{rank}(A)$, e.g., [3]. Thus, in the remainder of our presentation we will only consider the $\texttt{MinRank}$ problem. For a graph G, vertices x and y and an edge e, we denote by $G + \{x\}$, $G - \{x\}$, $G + e$, $G - e$, $G + (x, y)$, and $G - (x, y)$ the new graph obtained by adding or removing the vertex x, the edge e, or the edge (x, y) to G respectively. We now present several *equivalent* properties of $\texttt{CC}(G)$ and $\texttt{MR}_\Sigma(G)$ that are used later in our proof.

2.1 Properties of Clique-Cover

Property 2.1. *Given a graph G and a node v, $\texttt{CC}(G) \leq \texttt{CC}(G + \{v\}) \leq \texttt{CC}(G) + 1$ (Adding a node to a graph can increase the clique-cover by at most 1).*

Proof. The clique which contains the single node v can be added to the original clique cover of G. Thus resulting in new clique cover with size $\texttt{CC}(G) + 1$. For the other direction, any clique cover of $G + \{v\}$ induces a clique cover of the same size in G. □

Property 2.2. *Given a graph G which contains the nodes $\{x, y\}$, and doesn't contain the edge $e(x, y)$, $\texttt{CC}(G) \geq \texttt{CC}(G + e) \geq \texttt{CC}(G) - 1$ (Adding an edge to a graph can decrease the clique-cover by at most 1).*

Proof. Consider a graph G, which includes nodes x, y and does not include the edge (x, y). Suppose that adding an edge (x, y) to the graph can decrease the clique-cover by at least 2 $\Rightarrow \texttt{CC}(G + (x, y)) \leq \texttt{CC}(G) - 2$. Removing the node y (and as a result the edge (x, y)) from $(G + (x, y))$ causes a new graph with clique-cover that equals at most $(\texttt{CC}(G) - 2)$. Thus, adding the node y again to that graph and the edges of G adjacent to y, can increase the clique-cover at most by 1 (according to the first property above) $\Rightarrow \texttt{CC}(G) \leq \texttt{CC}(G) - 1$, a contradiction. The first inequality follows easily by the definition of a clique cover. □

Property 2.3. *Let G_1, G_2 be 2 graphs with one common node x (such that there are no edges between $G_1 - \{x\}$ and $G_2 - \{x\}$). If the following values are known:* $\texttt{CC}(G_1)$, $\texttt{CC}(G_2)$, $\texttt{CC}(G_1 - \{x\})$, $\texttt{CC}(G_2 - \{x\})$, *then the clique cover of the union of G_1 and G_2 is known and equal to the following:*

- *If* $\texttt{CC}(G_1) = \texttt{CC}(G_1 - \{x\})$ *or* $\texttt{CC}(G_2) = \texttt{CC}(G_2 - \{x\})$, *then* $\texttt{CC}(G_1 \cup G_2) = \texttt{CC}(G_1 - \{x\}) + \texttt{CC}(G_2 - \{x\})$.
- *If* $\texttt{CC}(G_1) = \texttt{CC}(G_1 - \{x\}) + 1$ *and* $\texttt{CC}(G_2) = \texttt{CC}(G_2 - \{x\}) + 1$, *then* $\texttt{CC}(G_1 \cup G_2) = \texttt{CC}(G_1 - \{x\}) + \texttt{CC}(G_2 - \{x\}) + 1$.

Proof. First consider the minimum size clique cover of the union of the disjoint graphs $(G_1 - \{x\})$ and $(G_2 - \{x\})$. It holds that the minimum size clique cover of the disjoint union equals the sum of the (disjoint) minimum size clique covers (this follows directly from the definition of a clique cover). Now, we will add the node x to the graph. By the properties above, the clique-cover size of the new graph can be equal to the clique-cover size of the union or it may increase by 1. We study two cases:

- If $\texttt{CC}(G_1) = \texttt{CC}(G_1 - \{x\})$ or $\texttt{CC}(G_2) = \texttt{CC}(G_2 - \{x\})$, then adding x to the union of the disjoint graphs doesn't increase the clique cover because x belongs to one of the cliques in the optimal cover of G_1 or G_2.
- If $\texttt{CC}(G_1) = \texttt{CC}(G_1 - \{x\}) + 1$ and $\texttt{CC}(G_2) = \texttt{CC}(G_2 - \{x\}) + 1$, we show that adding x to the union of the disjoint graphs increases the clique cover size by 1 (in this case $\{x\}$ will be the clique added to the clique cover). Assume by contradiction that $\texttt{CC}(G_1 \cup G_2) = \texttt{CC}(G_1 - \{x\}) + \texttt{CC}(G_2 - \{x\})$. Consider the minimum clique cover on $G_1 \cup G_2$. The clique in this cover that contains x can't contain additional nodes from both the graphs G_1 and G_2 as they are disjoint graphs (excluding x). Assume w.o.l.g. that x belongs to a clique with nodes from G_1. Thus, in the clique cover, the number of cliques that cover G_1 is exactly $\texttt{CC}(G_1 \cup G_2) - \texttt{CC}(G_2 - \{x\}) = \texttt{CC}(G_1 - \{x\})$. This implies that $\texttt{CC}(G_1 - \{x\}) = \texttt{CC}(G_1)$, in contradiction to the assumption.

□

2.2 Properties of `MinRank`

Property 2.4. *Given a graph G and a node v,* $\texttt{MR}_\Sigma(G) \leq \texttt{MR}_\Sigma(G + \{v\}) \leq \texttt{MR}_\Sigma(G) + 1$ *(Adding a node to a graph can increase the* `MinRank` *by at most 1).*

Proof. Consider the matrix M that realizes $\texttt{MR}_\Sigma(G)$. Adding a new row and column corresponding to the new vertex v, in which all new entries are of value 0 except the new diagonal entry $m_{v,v}$ that is of value 1, we get that $\texttt{MR}_\Sigma(G + \{v\}) \leq \texttt{MR}_\Sigma(G) + 1$. For the lower bound notice that any matrix that fits $G + \{v\}$ has a corresponding restriction (of lower or equal rank) that also fits G. □

Property 2.5. *Given a graph G which contains the nodes $\{x, y\}$, and doesn't contain the edge $e(x, y)$,* $\texttt{MR}_\Sigma(G) \geq \texttt{MR}_\Sigma(G + e) \geq \texttt{MR}_\Sigma(G) - 1$ *(Adding an edge to a graph can decrease the* `MinRank` *by at most 1).*

Proof. Consider a graph G, which includes nodes x, y and does not include the edge (x, y). Suppose that adding an edge (x, y) to the graph can decrease the `MinRank` by at least 2. Namely, that $\texttt{MR}_\Sigma(G + (x, y)) \leq \texttt{MR}_\Sigma(G) - 2$. Removing the node y (and as a result the edge (x, y)) from $(G + (x, y))$ causes a new graph with `MinRank` that equals at most $(\texttt{MR}_\Sigma(G) - 2)$ (as one can take any matrix that fits G and turn it into one that fits $G - \{y\}$ by removing the row and column that correspond to y). Thus, adding the node y again to that graph and the edges of G adjacent to y, can increase the `MinRank` at most by 1 (according to the first property above). We conclude that $\texttt{MR}_\Sigma(G) \leq \texttt{MR}_\Sigma(G) - 1$, a contradiction. For the upper bound in the assertion notice that any matrix that fits G also fits $G + e$. □

Property 2.6. *Let G_1, G_2 be 2 graphs with one common node x (such that there are no edges between $G_1 - \{x\}$ and $G_2 - \{x\}$). If the following values are known: $\mathtt{MR}_\Sigma(G_1)$, $\mathtt{MR}_\Sigma(G_2)$, $\mathtt{MR}_\Sigma(G_1 - \{x\})$, $\mathtt{MR}_\Sigma(G_2 - \{x\})$, then the* `MinRank` *of the union of G_1 and G_2 is known and equal to the following:*

- *If $\mathtt{MR}_\Sigma(G_1) = \mathtt{MR}_\Sigma(G_1 - \{x\})$ or $\mathtt{MR}_\Sigma(G_2) = \mathtt{MR}_\Sigma(G_2 - \{x\})$, then $\mathtt{MR}_\Sigma(G_1 \cup G_2) = \mathtt{MR}_\Sigma(G_1 - \{x\}) + \mathtt{MR}_\Sigma(G_2 - \{x\})$.*
- *If $\mathtt{MR}_\Sigma(G_1) = \mathtt{MR}_\Sigma(G_1 - \{x\}) + 1$ and $\mathtt{MR}_\Sigma(G_2) = \mathtt{MR}_\Sigma(G_2 - \{x\}) + 1$, then $\mathtt{MR}_\Sigma(G_1 \cup G_2) = \mathtt{MR}_\Sigma(G_1 - \{x\}) + \mathtt{MR}_\Sigma(G_2 - \{x\}) + 1$.*

Proof. Let M be a matrix which fits $G_1 \cup G_2$. Assume M has the following structure:

	G_1'	G_2'	x
G_1'	G_{1l}'	0	
G_2'	0	G_{2r}'	
x	x_l	x_r	1

In the above description we use the following notation. G_1' represents the subgraph $(G_1 - \{x\})$. G_2' represents the subgraph $(G_2 - \{x\})$. The row and column labels appear to the left or above the double line, while the matrix entries appear to the right and below the double line. Each row (and column) of M corresponds to a vertex v in $G_1 \cup G_2$. An entry m_{uv} in M corresponds to the vertices u and v in $G_1 \cup G_2$. For the row vector $(v_1, \ldots, v_n)$ corresponding to v we denote its entries corresponding to G_1' by v_{left} or v_l, its entries corresponding to G_2' by v_{right} or v_r and its entries corresponding to a vertex w by v_w. For example, for the vertex x we have that $x_x = 1$. Also for a vertex $v \in G_2'$ we have that $v_l = 0$ as there are no edges between G_2' and G_1'. The submatrix G_{1l}' (G_{2r}') consists of the vectors v_l (v_r) for $v \in G_1'$ ($v \in G_2'$). Finally, for any vertex v we abuse notation and refer to the row vector corresponding to v by the same notation: v. The same goes for subsets of vertices A.

Define M_1 and M_2 as matrices that fit G_1 and G_2 correspondingly, and have the minimum **rank** among all such matrices. By definition, $\mathtt{MR}_\Sigma(G_1) = \textbf{rank}(M_1)$ and $\mathtt{MR}_\Sigma(G_2) = \textbf{rank}(M_2)$. That is, M_1 and M_2 can be expressed as follows:

	G_1'	x
G_1'	H_1'	
x		1

	G_2'	x
G_2'	H_2'	
x		1

The submatrix H_1' (H_2') consists of the vectors corresponding to $v \in G_1'$ ($v \in G_2'$). We consider two cases:

- If $\mathtt{MR}_\Sigma(G_1) = \mathtt{MR}_\Sigma(G_1 - \{x\}) = \textbf{rank}(M_1)$, we first claim that the rows corresponding to G_1' in M_1 span a vector space of dimension $\textbf{rank}(M_1)$, and the vector corresponding to x in M_1 is spanned by the rows

corresponding to G'_1. Denote the latter vector by x_1. The above follows since the submatrix H'_1 fits G'_1 and has rank at most that of the the rows corresponding to G'_1. However, there is no matrix that fits G'_1 of rank less than $\texttt{MR}_\Sigma(G_1 - \{x\})$.

Now consider the matrix M above. In what follows we will suggest values for the entries of M that will yield rank equal to $\texttt{MR}_\Sigma(G_1 - \{x\}) + \texttt{MR}_\Sigma(G_2 - \{x\})$. Namely, we set G'_{1l} to be equal to H'_1, we set the x'th entry of the rows of G'_1 in M to be equal to their corresponding entries in M_1, we set G'_{2r} to be H'_2, the (row) vector x_{lx} (the entries of the vector x in M corresponding to vertices in $G'_1 \cup \{x\} = G_1$) to be the vector x_1, and x_r to be 0. As we show above, the vector x_{lx} is spanned by the rows of M corresponding to G'_1. Thus x is spanned by the rows corresponding to $G'_1 \cup G'_2$. We conclude that the resulting matrix M has rank $\textbf{rank}(M_1) + \textbf{rank}(H'_2) = \texttt{MR}_\Sigma(G'_1) + \texttt{MR}_\Sigma(G'_2)$. This shows that $\texttt{MR}_\Sigma(G) \leq \texttt{MR}_\Sigma(G'_1) + \texttt{MR}_\Sigma(G'_2)$. To obtain equality, notice that as G'_1 and G'_2 are disjoint graphs, it is not hard to verify (from the definition of `MinRank`) that these rows span a vector space of dimension at least $\texttt{MR}_\Sigma(G'_1) + \texttt{MR}_\Sigma(G'_2)$. The same proof can be shown for the case that $\texttt{MR}_\Sigma(G_2) = \texttt{MR}_\Sigma(G_2 - \{x\})$.

- $\texttt{MR}_\Sigma(G_1) = \texttt{MR}_\Sigma(G_1 - \{x\}) + 1$ and $\texttt{MR}_\Sigma(G_2) = \texttt{MR}_\Sigma(G_2 - \{x\}) + 1$. Assume by contradiction that the optimal matrix M satisfies $\textbf{rank}(M) = \texttt{MR}_\Sigma(G_1 \cup G_2) = \texttt{MR}_\Sigma(G_1 - \{x\}) + \texttt{MR}_\Sigma(G_2 - \{x\}) = \texttt{MR}_\Sigma(G'_1) + \texttt{MR}_\Sigma(G'_2)$. This implies that $x \in \textbf{span}(G'_1 \cup G'_2)$, as otherwise either $\textbf{rank}(G_{1l}) < \texttt{MR}_\Sigma(G'_1)$ or $\textbf{rank}(G_{2r}) < \texttt{MR}_\Sigma(G'_2)$ which is a contradiction to the facts that G'_{1l} fits G'_1 and G'_{2r} fits G'_2.

  Consider the rows of M that participate in the linear combination that yields the vector x. If these rows are included in G'_1, then it follows that the subvector x_{lx} (the left coordinates of the vector x including the coordinate corresponding to x) is in $\textbf{span}(G'_1)$ which implies that $\texttt{MR}_\Sigma(G_1) = \texttt{MR}_\Sigma(G_1 - \{x\})$, a contradiction. To see the contradiction, construct M_1 by setting the rows corresponding to G'_1 in M_1 to be equal to the corresponding entries in the rows corresponding to G'_1 in M, and the vector corresponding to x in M_1 to be equal to x_{lx}. A similar analysis can be done for the case that the rows of M that participate in the linear combination that yields the vector x are included in G'_2.

  If the the rows of M that participate in the linear combination that yields the vector x combine vectors from G'_1 and G'_2, one may consider the partial linear combination from G'_1 and G'_2 separately. Let x_1 be the linear combination resulting from the rows in G'_1 and x_2 be the linear combination corresponding to the rows in G'_2. Namely $x = x_1 + x_2$. Let the coordinate in x_i corresponding to the vertex x be a_i. As the entry $x_x = 1$, we have that $a_1 + a_2 = 1$. Thus, it cannot be the case that both a_1 and a_2 are 0. Assume w.l.o.g. that $a_1 \neq 0$. Also assume w.l.o.g. that $a_1 = 1$ (otherwise the entries of M_1 to be constructed shortly can be scaled accordingly). Now construct M_1 by setting the rows corresponding to G'_1 in M_1 to be equal to the corresponding entries in the rows corresponding to G'_1 in M, and the vector corresponding to x in M_1 to be equal to the corresponding coordinates in a revised version of x_1 in which the entry corresponding to the vertex x in x_1 is changed to a_1. It is not hard to verify that the modified version of x_1 is spanned by the rows corresponding to G'_1 in M_1 and thus $\texttt{MR}_\Sigma(G_1) = \texttt{MR}_\Sigma(G_1 - \{x\})$, a contradiction.

□

3 Overview of Baker's algorithm [2]

Our work is based on Baker's algorithmic paradigm [2]. In what follows we give a brief (and rough) overview of the main ideas that govern the algorithm of [2]. We will identify the major points that need to be addressed in order to apply the paradigm at hand to the case of the `MinRank` and clique cover problems.

Given an outerplanar graph G, the algorithm of Baker [2] has two major steps. In the first step, a tree representation $\bar{G}$ of G is constructed. Every node in $\bar{G}$ corresponds to a subgraph of G, where the root of $\bar{G}$ corresponds to G itself, each leaf in $\bar{G}$ corresponds to an edge, and internal nodes in $\bar{G}$ correspond to the subgraph of G induced

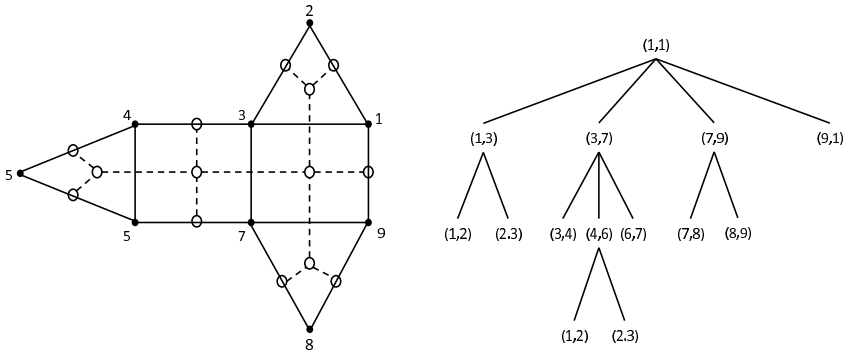

Figure 2: An outerplanar graph and its corresponding tree representation. Taken from [2].

by their children in $\bar{G}$. The construction of $\bar{G}$ from G is very simple in nature, and the tree $\bar{G}$ tightly resembles the standard notion of the *dual* to an (outer)planar graph. See Figure 1(d) and Figure 2.

In slightly more detail: $\bar{G}$ is constructed as follows (in this presentation we suppose there are no cutpoints in G, i.e., a vertex whose deletion disconnects the graph). Place a vertex in each interior face and on each exterior edge, and draw an edge from each vertex representing a face f to each vertex representing either an adjacent face (i.e., a face sharing an edge with f) or an exterior edge of f. (This tree is closely related to the dual of the graph; however, the dual would lack vertices for exterior edges and would have an additional vertex for the exterior face.) An example (taken from [2]) is shown in Figure 2.

The planar embedding induces a cyclic ordering on the edges of each vertex in the tree. Choosing a face vertex v as the root and choosing which child of v is to be its leftmost child determine the parent and ordering of children for every other vertex of $\bar{G}$. Label the vertices of $\bar{G}$ recursively, as follows: Label each leaf of the tree with the oriented exterior edge it represents. Label each face vertex with the first and last nodes in its children's labels. If a face vertex is labeled (x, y), the leaves of its subtree represent a directed walk of exterior edges in a counterclockwise direction from x to y. For the root, $x = y$ and the directed walk covers all the exterior edges. For any other face vertex v, $x \neq y$, and (x, y) is an interior edge shared by the face represented by v and the face represented by its parent in the tree. We define $G(x, y)$ to be the subgraph corresponding to the subtree of $\bar{G}$ rooted at tree-node (x, y). Namely, $G(x, y)$ contains all edges corresponding to leaves in it's subtree with the addition of the edge (x, y).

For example, in Figure 2, the leaves of the node labeled $(3, 7)$ represent a walk along nodes $3, 4, 5, 6, 7$. The leaves of the root $(1, 1)$ represent a counterclockwise walk around the exterior edges beginning and ending at node 1. The vertex labeled $(1, 3)$ represents the face containing nodes $1 - 3$, its parent represents the face containing nodes $1, 3, 7, 9$, and $(1, 3)$ is the interior edge shared by these faces. We refer the interested reader to Baker's original work [2] for a clear and full presentation of the tree structure $\bar{G}$.

Once the tree $\bar{G}$ is given, the objective in [2] is to dynamically compute the objective function `Obj` at hand (e.g., `MinRank`) in a *bottom up* manner from the leaves to the root. Namely, for each vertex (x, y) of $\bar{G}$, based on the algorithm of [2], we will define a table for this vertex which contains 4 values:

- The solution to the objective function at hand for the subgraph $G(x, y)$ including the nodes x and y.
- The solution to the objective function at hand for the subgraph $G(x, y)$ including the node x and not y.
- The solution to the objective function at hand for the subgraph $G(x, y)$ including the node y and not x.
- The solution to the objective function at hand for the subgraph $G(x, y)$ excluding the nodes x and y

Procedure `table`(v)

if v is a level 1 leaf corresponding to an edge with label (x, y)
  then
    return a table representing the edge (x, y);
  else * v is a face vertex*
    begin
    T = table(u), where u is the leftmost child of v;
    for each other child u of v from left to right
      T = **merge** (T, table(u));
    return (**adjust**(T));
    end

Figure 3: The algorithm `table` from [2].

The table for a leaf of the tree representing an edge (x, y) specifies (in our case in which `Obj` is either the `MinRank` or clique cover) that `Obj` for the subgraph considered is 1 if exactly one endpoint of (x, y) or both of them are in the subgraph, and 0 if neither x nor y are in the subgraph. The table for every other vertex will be computed recursively by merging the tables of its children according to the algorithm of [2] given in Figure 3.

There are two major operations that need to be addressed in the above procedure. The **merge** operation takes as input the intermediate table T and the table of a tree node u and returns a "merged" table of the two. More specifically, let v be a vertex which represents a face vertex (x, y) with the following children: (x, a), (a, b), ... (i, z), (z, w), ... (j, y). Assume that we are considering the child (z, w) of (x, y) in $\bar{G}$. Let $G(x, z)$ be the subgraph of G that includes all the edges that are in the union of the subgraphs $G(x, a) \cup G(a, b) \cup ... \cup G(i, z)$. The current table T includes information for the subgraph $G(x, z)$. Namely (by induction) assume the current table T has a value for each *bit pair* representing x and z. By the term "bit pair" we refer to each possibility for the existence of x and z in the subgraph $G(x, z)$, meaning that the table T has 4 solutions to the objective function at hand, each of a subgraph $G(x, z)$ (solution for the subgraph containing x and z, solution for the subgraph containing x and not containing z, solution for the subgraph containing z and not containing x, solution for the subgraph without both x and z.) The child u has label (z, w) for some w, and table(u) has a value for each bit pair representing z and w. The goal of procedure merge is to construct an updated table T with a solution of the objective function on $G(x, w) \cup G(w, z)$ for every bit pair representing x and w.

The **adjust** operation takes as input a table T consisting of the merge of all the children of v, and returns a table v corresponding to v. Specifically, let v be a vertex which represents a face vertex (x, y) with the following children: (x, a), (a, b), ... (i, z), (z, w), ... (j, y). After merging all of the children, we get a table for the subgraph $G(x, y)$, which includes the clique cover of the subgraph $G(x, y)$ for each bitpair of (x, y), but does not include the edge (x, y) if such an edge exists. The goal of procedure adjust is to solve the objective function at hand after adding this edge.

To apply Baker's algorithm to an objective function `Obj` of our choice, we must show how to implement the subroutines **merge** and **adjust**. It is not hard to verify that to prove that one can implement the **merge** operation, it suffices to present an algorithm that takes as input two induced subgraphs G_1 and G_2 of G that intersect at a single vertex x and the solutions `Obj`(G_1), `Obj`$(G_1 - \{x\})$, `Obj`(G_2) and `Obj`$(G_2 - \{x\})$ and returns a solution for `Obj`$(G_1 \cup G_2)$. Here, as the graphs G_1 and G_2 are induced subgraphs of G notice that there are no edges between $G_1 - \{x\}$ and $G_2 - \{x\}$. Similarly, for the **adjust** operation, it suffices to present an algorithm which takes as input an outerplanar graph G that includes vertices x and y but does not include the edge (x, y) on the outer face of G, the solutions `Obj`$(G - \{x\})$, `Obj`$(G - \{y\})$, `Obj`$(G - \{x, y\})$, and `Obj`(G) and returns a solution for `Obj`$(G + e)$. The implementation of these tasks for the clique cover and `MinRank` objective functions are in

cases highly non-trivial, and to the best of our knowledge have not been addressed in the past. In the upcoming sections we will address the task of implementing these subroutines efficiently.

4 The MinRank objective function

4.1 Merge

In the **merge** operation one takes two subgraphs that are *almost* disjoint for which the optimal MinRank is known, and returns the MinRank of their union (including the corresponding matrix A of minimum rank). If the graphs were disjoint then the MinRank of the union is just the union of the corresponding MinRanks, however, as the subgraphs share a vertex, the MinRank of their union might be smaller according to the claim below.

Claim 4.1. *Let G_1 and G_2 be induced subgraphs of G that have a single vertex x in common. If $\mathtt{MR}_\Sigma(G_1) = \mathtt{MR}_\Sigma(G_1 - \{x\})$ or $\mathtt{MR}_\Sigma(G_2) = \mathtt{MR}_\Sigma(G_2 - \{x\})$, then $\mathtt{MR}_\Sigma(G_1 \cup G_2) = \mathtt{MR}_\Sigma(G_1 - \{x\}) + \mathtt{MR}_\Sigma(G_2 - \{x\})$. Otherwise, $\mathtt{MR}_\Sigma(G_1 \cup G_2) = \mathtt{MR}_\Sigma(G_1 - \{x\}) + \mathtt{MR}_\Sigma(G_2 - \{x\}) + 1$. Moreover, in both cases the corresponding matrix A of minimum rank that fits $G_1 \cup G_2$ can be obtained efficiently from the matrices corresponding to the* MinRank *of $G_1 - \{x\}$ and $G_2 - \{x\}$.*

Proof. G_1 and G_2 intersect at a single node, thus, using Property 2.6, we can find the $\mathtt{MR}_\Sigma$ of $G_1 \cup G_2$. □

4.2 Adjust

In the "adjust" operation one takes an outerplanar graph G with vertices x and y but without the edge (x, y) that lies on the outer face of G; and computes the MinRank of $G + e$ based on the MinRank of the graph $G - \{x\}$, the graph $G - \{y\}$, the graph $G - \{x, y\}$ and the graph G. As we have shown, it holds that $\mathtt{MR}_\Sigma(G + e)$ either equals $\mathtt{MR}_\Sigma(G)$ or is smaller and equals $\mathtt{MR}_\Sigma(G) - 1$, however the correct answer depends strongly on the the values of the MinRank in the subgraphs of G that do not include the vertices x or y. The following two claims summarize the adjust operation when applied to the MinRank objective function. The first claim covers almost all possible settings except one, and can be proven relatively straightforward from the basic properties of MinRank combined with Property 2.6 above. The second claim addresses the last setting, and is more challenging.

Claim 4.2. *Let G be an outerplanar graph that includes vertices x and y but does not include the edge $e = (x, y)$ that sits on the outer face of G. Then the value of $\mathtt{MR}_\Sigma(G + e)$ is determined by the following table which expresses the possible input values of $\mathtt{MR}_\Sigma(G - \{x, y\})$, $\mathtt{MR}_\Sigma(G - \{x\})$, and $\mathtt{MR}_\Sigma(G - \{y\})$ as a function of $\lambda = \mathtt{MR}_\Sigma(G)$; and the resulting value of $\mathtt{MR}_\Sigma(G + e)$ as a function of $\lambda = \mathtt{MR}_\Sigma(G)$:*

$\mathtt{MR}_\Sigma(G - \{x, y\})$	$\mathtt{MR}_\Sigma(G - \{x\})$	$\mathtt{MR}_\Sigma(G - \{y\})$	$\mathtt{MR}_\Sigma(G + e)$
λ			λ
$\lambda - 2$			$\lambda - 1$
$\lambda - 1$	λ	$\lambda - 1$	λ
$\lambda - 1$	$\lambda - 1$	λ	λ

Moreover, the minimum rank matrix A corresponding to $\mathtt{MR}_\Sigma(G + e)$ can be computed efficiently using the matrices corresponding to the MinRank *of the subgraphs above.*

Proof. We consider the different cases stated in the assertion:

1. $\mathtt{MR}_\Sigma(G - \{x, y\}) = \mathtt{MR}_\Sigma(G)$. As $e = (x, y)$, it holds that $\mathtt{MR}_\Sigma(G + e) \geq \mathtt{MR}_\Sigma(G - \{x, y\})$ (as any matrix that fits $G + e$ has a minor which fits $G - \{x, y\}$). Thus, adding the nodes $\{x, y\}$ and the edge (x, y) can't decrease the MinRank of the former graph $\mathtt{MR}_\Sigma(G - \{x, y\})$ which is equal to $\mathtt{MR}_\Sigma(G)$.

2. $\texttt{MR}_\Sigma(G - \{x, y\}) = \texttt{MR}_\Sigma(G) - 2$. Adding the nodes $\{x, y\}$ plus the edge (x, y) to $G - \{x, y\}$ increases its MinRank by at most 1 (one can just append to a matrix that fits $G - \{x, y\}$ the row vector that is 1 on coordinates x and y and 0 otherwise). Thus we get $\texttt{MR}_\Sigma(G + e) \leq (\texttt{MR}_\Sigma(G) - 2) + 1 = \texttt{MR}_\Sigma(G) - 1$. The value $\texttt{MR}_\Sigma(G + e)$ can't be less than or equal to $\texttt{MR}_\Sigma(G) - 2$ because of Property 2.5 (adding an edge to a graph can decrease the MinRank by 1 at most).

3. $\texttt{MR}_\Sigma(G - \{x\}) = \texttt{MR}_\Sigma(G)$. As in the first case it holds that $\texttt{MR}_\Sigma(G + e) \geq \texttt{MR}_\Sigma(G - \{x\})$ (as any matrix that fits $G + e$ has a minor which fits $G - \{x\}$).

4. Same as 3.

□

Claim 4.3. *Let G be an outerplanar graph that includes vertices x and y but does not include the edge $e = (x, y)$ that sits on the outer face of G. Then if $\texttt{MR}_\Sigma(G - \{x, y\}) = \texttt{MR}_\Sigma(G - \{x\}) = \texttt{MR}_\Sigma(G - \{y\}) = \texttt{MR}_\Sigma(G) - 1$ the* MinRank *of $G + e$ is determined by the following cases. (a) If there are no vertices z in $G + e$ such that x, y, z from a triangle (a clique) in $G + e$ then $\texttt{MR}_\Sigma(G + e) = \texttt{MR}_\Sigma(G)$. (b) If there exists a vertex z in $G + e$ such that x, y, z from a triangle (a clique) in $G + e$ then $\texttt{MR}_\Sigma(G + e)$ depends on $\texttt{MR}_\Sigma(G - \{x, y, z\})$. Namely, if $\texttt{MR}_\Sigma(G - \{x, y, z\}) = \texttt{MR}_\Sigma(G) - 2$ then $\texttt{MR}_\Sigma(G + e) = \texttt{MR}_\Sigma(G) - 1$, otherwise $\texttt{MR}_\Sigma(G + e) = \texttt{MR}_\Sigma(G)$. Moreover, the minimum rank matrix corresponding to $\texttt{MR}_\Sigma(G + e)$ can be computed efficiently using the matrices corresponding to the* MinRank *of the subgraphs above.*

When using Claim 4.3 in our algorithm we use the fact that in case (b), when the algorithm of Baker [2] computes the MinRank (and corresponding matrix A) of G the MinRanks (and corresponding matrices) of $G - \{x, z\}$, $G - \{y, z\}$ and thus also $G - \{x, y, z\}$ are known. Claim 4.3 is the most challenging claim proven in this work. Examples in which the MinRank of G after adding the edge (x, y) are equal to $\texttt{MR}_\Sigma(G)$ or $\texttt{MR}_\Sigma(G) - 1$ are given below.

- If $x - z - y$ is a path, then adding the edge (x, y) closes a triangle and the MinRank is decreased by 1.
- If $x - z - u - v - y$ is a path, then adding the edge (x, y) doesn't decrease the MinRank.

Proof. We consider the following cases:

Case 1: There is a node $'z'$ such that x, y, z close a triangle. According to Baker's method, (x, z) and (z, y) are children of the node (x, y), meaning that before finding the value of $\texttt{MR}_\Sigma(G)$ the procedure "merge" was performed with the nodes (x, z) and (z, y). The following values were known in this process: $\texttt{MR}_\Sigma(G(x, z))$, $\texttt{MR}_\Sigma(G(x, z) - \{x, z\})$, $\texttt{MR}_\Sigma(G(z, y))$, and $\texttt{MR}_\Sigma(G(z, y) - \{z, y\})$. Thus, $\texttt{MR}_\Sigma(G) - \{x, y, z\})$ is known and equals $\texttt{MR}_\Sigma(G(x, z) - \{x, z\}) + \texttt{MR}_\Sigma(G(z, y) - \{z, y\})$ (the subgraphs $G(x, z)$ and $G(z, y)$ excluding the nodes x, z, y are disjoint graphs because of the outerplanarity constraint, see, e.g., Figure 1.

We consider two cases: If $\texttt{MR}_\Sigma(G - \{x, y, z\}) = \texttt{MR}_\Sigma(G) - 2$, we claim that $\texttt{MR}_\Sigma(G + e) = \texttt{MR}_\Sigma(G) - 1$. This follows since, given a matrix A that fits $G - \{x, y, z\}$ one can construct one that fits $G + e$ by expanding A in a natural way (adding rows/columns corresponding to x, y, z), in which the only new entries that are non-zero are those corresponding to a pair in $\{x, y, z\}$ (which can be set to equal 1). Thus, $\texttt{MR}_\Sigma(G + (x, y)) = \texttt{MR}_\Sigma(G) - 1$ (the MinRank is decreased by 1 after adding the edge (x,y)).

For the second case, if $\texttt{MR}_\Sigma(G - \{x, y, z\}) = \texttt{MR}_\Sigma(G) - 1$, we now show that adding the edge (x, y) can't decrease the MinRank, thus $\texttt{MR}_\Sigma(G + (x, y)) = \texttt{MR}_\Sigma(G)$.

Claim 4.4. *If $\texttt{MR}_\Sigma(G - \{x, y, z\}) = \texttt{MR}_\Sigma(G) - 1$, then $\texttt{MR}_\Sigma(G + (x, y)) = \texttt{MR}_\Sigma(G)$.*

Proof. We start by noting that the removal of the three vertices $\{x, y, z\}$ in G disconnects G into 2 disjoint components. See, e.g., Figure 1. We denote these components by A_1 and A_2. As $G + e$ is outerplanar, and the edge (x, y) is on the outer face of $G + e$, we can assume w.l.o.g. that vertices in A_1 are not connected by an edge to vertices in A_2. Moreover, vertices in A_1 are not connected by an edge to y, while vertices in A_2 are not connected by an edge to x.

Define M as a matrix which fits $G+(x, y)$, and has the minimum **rank** among all such matrices. By definition, $\texttt{MR}_\Sigma(G+(x, y)) = \min\{\textbf{rank}(M) \mid M \text{ fits } (G+(x, y))\}$. Assume in contradiction that after adding the edge (x, y), $\texttt{MR}_\Sigma(G)$ is decreased by 1, meaning that $\textbf{rank}(M) = \texttt{MR}_\Sigma(G + (x, y)) = \texttt{MR}_\Sigma(G) - 1$. We will see that by changing a few entries in M we obtain a new matrix M', with the same **rank** as M, which fits G (the original graph without (x, y)), implying that $\texttt{MR}_\Sigma(G) \leq \textbf{rank}(M') = \textbf{rank}(M) = \texttt{MR}_\Sigma(G) - 1$, a contradiction. We will thus conclude that $\texttt{MR}_\Sigma(G + (x, y)) = \texttt{MR}_\Sigma(G)$. Consider the matrix M:

	A_1	x	y	z	A_2
A_1	A_{1l}		0		0
x	x_l	1	x_y	*	0
y	0	y_x	1	*	y_r
z	z_l	*	*	1	z_r
A_2	0	0			A_{2r}

In the above description we use the following notation. A_1 represents the subgraph corresponding to the face vertex (x, z) in G. A_2 represents the subgraph corresponding to the face vertex (z, y) in G. An entry m_{uv} in M corresponds to the vertices labeled by u and v in G. Each row of M corresponds to a vertex v in G. For the row vector $(v_1 \dots v_n)$ corresponding to v we denote its entries corresponding to A_1 by v_{left} or v_l, its entries corresponding to A_2 by v_{right} or v_r and its entries corresponding to a vertex w by v_w. For example, for the vertex x we have that $x_x = 1$. Also for a vertex $v \in A_2$ we have that $v_l = 0$ as there are no edges between A_2 and A_1. Finally, for any vertex v we abuse notation and refer to the row vector corresponding to v by the same notation: v. Similarly for a set of vertices A we refer to the row vectors corresponding to A in M by A.

In the description of M above we have specified entries that must be 0, entries that must be 1, and some entries of interest that can be either 0 or 1 (denoted by *). For each matrix which fits the graph G (the graph that doesn't include the edge (x, y)), it is known that $x_y = 0$ and $y_x = 0$, because there is no edge between x and y. Consider the matrix M above which fits $G + (x, y)$. If $x_y = 0$ and $y_x = 0$, then M also fits G, implying that $\texttt{MR}_\Sigma(G) \leq \textbf{rank}(M) = \texttt{MR}_\Sigma(G) - 1$, a contradiction. Thus, one of the values x_y or y_x must equal 1. Recall that we are assuming that $\texttt{MR}_\Sigma(G-\{x, y, z\}) = \texttt{MR}_\Sigma(G)-1$, meaning that $\textbf{rank}(M) = \texttt{MR}_\Sigma(G)-1 = \texttt{MR}_\Sigma(G-\{x, y, z\}) = \texttt{MR}_\Sigma(A_1 \cup A_2)$. Consider the rows of M corresponding to A_1 and A_2. We now claim that these rows span a vector space of dimension $\texttt{MR}_\Sigma(G) - 1$ and thus span M. Indeed, if the rows of A_1 and A_2 spanned a vector space of lower dimension, then one could construct a matrix that fits the subgraph induced by $A_1 \cup A_2$ with **rank** lower than $\texttt{MR}_\Sigma(G) - 1$. However, as stated above, $\texttt{MR}_\Sigma(A_1 \cup A_2) = \texttt{MR}_\Sigma(G) - 1$, a contradiction.

We conclude that the row vector x (corresponding to the vertex x) is in $\textbf{span}(A_1 \cup A_2)$. The same holds for the vectors y and z. Consider the sub vector x_l. Potentially, x_l could be spanned by the left coordinates in the vectors of A_1 (denoted as A_{1l}) and the left coordinates in the vectors of A_2 (denoted as A_{2l}). However, A_{2l} is all zero (as there are no edges between vertices of A_2 and those of A_1). Thus, $x_l \in \textbf{span}(A_{1l})$. Moreover, it also holds that the vector (x_l, x_x) (i.e, the entries of x corresponding to vertices $(A_1 \cup x)$) is spanned by the coordinates of A_1 corresponding to the vertices $(A_1 \cup x)$.

We now suggest to change the vector x to x' such that $x' \in \textbf{span}(A_1)$. This can be done by zeroing out the

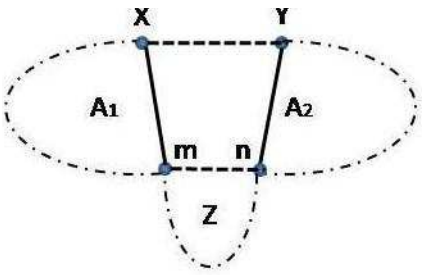

Figure 4: The partition of G into 4 graphs in the proof of Claim 4.4 (case 2).

values of x in the coordinate y and by changing the value of x in coordinate z to according to the following rule. By the discussion above, let the coefficients $\{\alpha_i\}$ satisfy : $(x_l, x_x) = \sum_{a_i \in A_1} \alpha_i (a_{il}, a_{ix})$. Now set $x' = \sum_{a_i \in A_1} \alpha_i a_i$. Notice that $x'_y = 0$. In a similar (and symmetric) way we can change the vector y to $y' \in$ **span**(A_2) such that $y'_x = 0$. The resulting matrix M' still has rank at most $\mathtt{MR}_\Sigma(G) - 1 =$ **rank**$(A_1 \cup A_2)$, as all new and old row vectors are spanned by $A_1 \cup A_2$. Moreover, M' fits G. Thus, $\mathtt{MR}_\Sigma(G) \leq$ **rank**$(M') = \mathtt{MR}_\Sigma(G) - 1$, a contradiction. □

Case 2: There is no node $'z'$ such that x, y, z closes a triangle. In Figure 4 we present a decomposition of G into 4 graphs: A_1, A_2, Z, $\{x, y, m, n\}$. As described previously, the graph G (denoted by $G(x, y)$ earlier) corresponds to the subgraph of the face vertex which is labeled (x, y) in the tree construction of Baker. We denote by m the neighbor of x (on the corresponding face) which is furthest away from x in a counterclockwise direction. We denote by n the neighbor of y which is furthest away from y (on the corresponding face) in a clockwise direction. Removing the vertices x, y, m and n from G, we get the 4 disconnected components that appear in Figure 4.

Define M as a matrix which fits $G+(x, y)$, and has the minimum **rank** among all such matrices. By definition, $\mathtt{MR}_\Sigma(G+(x, y)) = \min\{$**rank**$(M) \mid M$ fits $(G+(x, y))\}$. Assume in contradiction that after adding the edge (x, y), $\mathtt{MR}_\Sigma(G)$ is decreased by 1, meaning that **rank**$(M) = \mathtt{MR}_\Sigma(G+(x, y)) = \mathtt{MR}_\Sigma(G) - 1$. We will see that by changing a few entries in M we obtain a new matrix M', with the same **rank** as M, which fits G (the original graph without (x, y)), implying that $\mathtt{MR}_\Sigma(G) \leq$ **rank**$(M') =$ **rank**$(M) = \mathtt{MR}_\Sigma(G) - 1$, a contradiction. We will thus conclude that $\mathtt{MR}_\Sigma(G + (x, y)) = \mathtt{MR}_\Sigma(G)$. Consider the matrix M (we use a similar representation to that given in the analysis of the previous case).

	A_1	x	m	Z	n	y	A_2
A_1				0	0	0	0
x	x_l	1	*	0	0	x_y	0
m	m_l	*	1			0	0
Z	0	0				0	0
n	0	0			1	*	n_r
y	0	y_x	0	0	*	1	y_r
A_2	0	0	0	0			

For each matrix which fits the graph G (the graph that doesn't include the edge (x, y)), it is known that $x_y = 0$ and $y_x = 0$, because there is no edge between x and y. Consider the matrix M above which fits $G + (x, y)$. If

$x_y = 0$ and $y_x = 0$, then M also fits G, implying that $\mathtt{MR}_\Sigma(G) \leq \textbf{rank}(M) = \mathtt{MR}_\Sigma(G) - 1$, a contradiction. Thus, one of the values x_y or y_x must equal 1. Recall that we are asuuming that $\mathtt{MR}_\Sigma(G - \{x, y\}) = \mathtt{MR}_\Sigma(G) - 1$, meaning that $\mathtt{MR}_\Sigma(G) - 1 = \textbf{rank}(M) = \textbf{rank}(G - \{x, y\}) = \mathtt{MR}_\Sigma(A_1 \cup A_2 \cup Z \cup m \cup n)$ (here, "$A_1 \cup A_2 \cup Z \cup m \cup n$" refers to the subgraph induced on these vertices of G). Consider the rows of M corresponding to A_1 , A_2, Z, m, n. We now claim that these rows span a vector space of dimension $\mathtt{MR}_\Sigma(G) - 1$ and thus span M. Indeed, if the rows of A_1 , A_2, Z, m, n spanned a vector space of lower dimension, then one could construct a matrix that fits the subgraph induced by $A_1 \cup A_2 \cup Z \cup m \cup n$ with **rank** lower than $\mathtt{MR}_\Sigma(G) - 1$. However, as stated above, $\mathtt{MR}_\Sigma(A_1 \cup A_2 \cup Z \cup m \cup n) = \mathtt{MR}_\Sigma(G) - 1$, a contradiction.

As before, for a vertex x we abuse notation and refer to x as the corresponding row vector in M. Similarly for a set of vertices A we refer to the row vectors corresponding to A in M by A. We conclude that the row vector x (corresponding to the vertex x) is in $\textbf{span}(A_1 \cup A_2 \cup Z \cup m \cup n)$. The same holds for the vector y.

The following cases must be considered:

- Both m and n are in $\textbf{span}(A_1 \cup A_2 \cup Z)$, meaning that $\textbf{rank}(A_1 \cup A_2 \cup Z) = \mathtt{MR}_\Sigma(G) - 1$. In this case, the row vector x is in $\textbf{span}(A_1 \cup A_2 \cup Z)$. Consider the sub vector x_l. Potentially, x_l could be spanned by the left coordinates in the vectors of A_1 (denoted as A_{1l}), the left coordinates in the vectors of A_2 (denoted as A_{2l}), and the left coordinates in the vectors of Z (denoted as Z_l). However, A_{2l} is all zero (as there are no edges between vertices of A_2 and those of A_1), and Z_l is all zero (as there are no edges between vertices of Z and those of A_1). Thus, $x_l \in \textbf{span}(A_{1l})$. Moreover, it also holds that the vector (x_l, x_x) (i.e, the entries of x corresponding to vertices $(A_1 \cup x)$) is spanned by the coordinates of A_1 corresponding to the vertices $(A_1 \cup x)$.

  We now suggest to change the vector x to x' such that $x' \in \textbf{span}(A_1)$. This can be done by zeroing out the values of x in the coordinate y and by changing the value of x in coordinate m according to the following rule. By the discussion above, let the coefficients $\{\alpha_i\}$ satisfy : $(x_l, x_x) = \sum_{a_i \in A_1} \alpha_i (a_{il}, a_{ix})$. Now set $x' = \sum_{a_i \in A_1} \alpha_i a_i$.

  Similarly, we change the vector y to $y' \in \textbf{span}(A_2)$ with $y'_x = 0$. The resulting matrix M' still has rank at most $\mathtt{MR}_\Sigma(G) - 1 = \textbf{rank}(A_1 \cup A_2 \cup Z)$, as all new and old row vectors are spanned by $A_1 \cup A_2 \cup Z$. Moreover, M' fits G. Thus, $\mathtt{MR}_\Sigma(G) \leq \textbf{rank}(M') = \mathtt{MR}_\Sigma(G) - 1$, a contradiction.

- n is in $\textbf{span}(A_1 \cup A_2 \cup Z)$, and m not. We first suggest to change the vector m to m' by zeroing out the values of m in the coordinates Z and n. Now consider the vector x, or more specifically the sub vector x_l. Potentially, x_l could be spanned by the left coordinates in the vectors of A_1 (denoted as A_{1l}), the left coordinates in the vectors of A_2 (denoted as A_{2l}), the left coordinates in the vectors of Z (denoted as Z_l), and the left coordinates in the vector m (denoted as m_l). However, A_{2l} is all zero (as there are no edges between vertices of A_2 and those of A_1), and Z_l is all zero (as there are no edges between vertices of Z and those of A_1). Thus, $x_l \in \textbf{span}(A_{1l} \cup m_l) = \textbf{span}(A_{1l} \cup m'_l)$. Moreover, it also holds that the vector (x_l, x_x) (i.e, the entries of x corresponding to vertices $(A_1 \cup x)$) is spanned by the coordinates of A_1 and m' corresponding to the vertices $(A_1 \cup x)$.

  We now suggest to change the vector x to x' such that $x' \in \textbf{span}(A_1 \cup m')$. This can be done by zeroing out the values of x in the coordinate y and by changing the value of x in coordinate m according to the following rule. By the discussion above, let the coefficients $\{\alpha_i\}$ satisfy : $(x_l, x_x) = \sum_{a_i \in A_1 \cup \{m'\}} \alpha_i (a_{il}, a_{ix})$. Now set $x' = \sum_{a_i \in A_1 \cup \{m'\}} \alpha_i a_i$. Notice that as m' is 0 is coordinates corresponding to Z, n, y and A_2 the same holds for so is x'.

  We now address the vector y as we did in the former case (recall that n is in $\textbf{span}(A_1 \cup A_2 \cup Z)$). Namely, we change the vector y to $y' \in \textbf{span}(A_2)$ with $y_x = 0$. The matrix M' with the new rows m', x' and y' still has rank at most $\textbf{rank}(A_1 \cup A_2 \cup Z \cup \{m'\}) \leq \textbf{rank}(A_1 \cup A_2 \cup Z \cup \{m\}) = \mathtt{MR}_\Sigma(G) - 1$. This

follows as all new and old row vectors are spanned by $A_1 \cup A_2 \cup Z \cup \{m'\}$. Moreover, M' fits G. Thus, $\mathtt{MR}_\Sigma(G) \leq \textbf{rank}(M') = \mathtt{MR}_\Sigma(G) - 1$, a contradiction.

- n is in **span**$(A_1 \cup A_2 \cup Z)$ and m not. Similar to the former case.
- both m and n are not in **span**$(A_1 \cup A_2 \cup Z)$. We need to consider the following cases:
  - m is not in **span**$(A_1 \cup A_2 \cup Z \cup n)$ and n is not in **span**$(A_1 \cup A_2 \cup Z \cup m)$. We suggest to change the vector m to m' by zeroing out the values of m in the coordinates Z and n. We also suggest to change the vector n to n' by zeroing out the values of n in the coordinates Z and m. Now we can change the vector x to x' such that $x' \in$ **span**$(A_1 \cup m')$. This can be done by zeroing out the values of x in the coordinate y and by changing the value of x in coordinate m as done before. Similarly, we change the vector y to $y' \in$ **span**$(A_2 \cup n')$. The resulting matrix M' (with m', n', x' and y') has rank at most **rank**$(A_1 \cup A_2 \cup Z \cup \{m'\} \cup \{n'\}) \leq \mathtt{MR}_\Sigma(G) - 1$, as all new and old row vectors are spanned by $A_1 \cup A_2 \cup Z \cup m' \cup n'$. Moreover, M' fits G. Thus, $\mathtt{MR}_\Sigma(G) \leq \textbf{rank}(M') = \mathtt{MR}_\Sigma(G) - 1$, a contradiction.
  - n is in **span**$(A_1 \cup A_2 \cup Z \cup m)$. As $n \notin$ **span**$(A_1 \cup A_2 \cup Z)$ it follows that $n = \alpha_m m + \sum_{a_i \in A_1 \cup A_2 \cup Z} \alpha_i a_i$, where $a_m \neq 0$. Consider the sub vector $n_l = \alpha_m m_l + \sum_{a_i \in A_1 \cup A_2 \cup Z} \alpha_i a_{il}$. A_{2l} and Z_l are all zero, meaning that $n_l = \alpha_m m_l + \sum_{a_i \in A_1} \alpha_i a_{il}$. Moreover, n_l must equal 0 as there are no edges between vertices of A_1 and n. Thus, $0 = \alpha_m m_l + \sum_{a_i \in A_1} \alpha_i a_{il}$ where $\alpha_m \neq 0$. This implies that m_l is in **span**(A_{1l}). Consider the sub vector x_l. Potentially, x_l could be spanned by A_{1l}, A_{2l}, Z_l, m_l and n_l. A_{2l} , Z_l, and n_l are all zero, meaning that $x_l \in$ **span**$(A_{1l} \cup m_l)$. As we know that $m_l \in$ **span**(A_{1l}), we conclude that $x_l \in$ **span**(A_{1l}). Now we can change the vector x to x' such that $x' \in$ **span**(A_1) exactly as done before.

    To address y, notice that in this case we have that (y_l, y_y) is spanned by the corresponding coordinates of A_2. Thus as done before we change the vector y to $y' \in$ **span**(A_2) with $y'_x = 0$. The resulting matrix M' (with x' and y') still has rank at most $\mathtt{MR}_\Sigma(G) - 1 =$ **rank**$(A_1 \cup A_2 \cup Z \cup m)$, as all new and old row vectors are spanned by $A_1 \cup A_2 \cup Z \cup m$. Moreover, M' fits G. Thus, $\mathtt{MR}_\Sigma(G) \leq \textbf{rank}(M') = \mathtt{MR}_\Sigma(G) - 1$, a contradiction.
  - m is in **span**$(A_1 \cup A_2 \cup Z \cup n)$.Similar to the former case.

□

5 The clique cover objective function

As in Section 4 one may prove properties for the clique cover objective on the routines "merge" and "adjust" in the algorithm of [2]. The claims corresponding to the clique cover objective function are exactly those presented in Section 4 when the objective `MinRank` (or $\mathtt{MR}_\Sigma(G)$) is replaced by the term "clique cover" (or $\mathtt{CC}(G)$). We note that the first and second claims we discuss for the clique cover objective follow relatively straightforwardly from the basic properties of clique covers in undirected graphs, while the third claim (analogous to Claim 4.3) is more challenging to prove.

Claim 5.1. *Let G_1 and G_2 be induced subgraphs of G that have a single vertex x in common. If $\mathtt{CC}(G_1) = \mathtt{CC}(G_1 - \{x\})$ or $\mathtt{CC}(G_2) = \mathtt{CC}(G_2 - \{x\})$, then $\mathtt{CC}(G_1 \cup G_2) = \mathtt{CC}(G_1 - \{x\}) + \mathtt{CC}(G_2 - \{x\})$. Otherwise, $\mathtt{CC}(G_1 \cup G_2) = \mathtt{CC}(G_1 - \{x\}) + \mathtt{CC}(G_2 - \{x\}) + 1$. Moreover, in both cases the optimal clique cover of $G_1 \cup G_2$ can be obtained efficiently from those of $G_1 - \{x\}$ and $G_2 - \{x\}$.*

Proof. G_1 and G_2 intersect at a single node, thus, using Property 2.3, we can find the clique cover of $G_1 \cup G_2$. □

Claim 5.2. *Let G be an outerplanar graph that includes vertices x and y but does not include the edge $e = (x, y)$ that sits on the outer face of G. Then the value of $\texttt{CC}(G + e)$ is determined by the following table which expresses the possible input values of $\texttt{CC}(G - \{x, y\})$, $\texttt{CC}(G - \{x\})$, and $\texttt{CC}(G - \{y\})$ as a function of $\lambda = \texttt{CC}(G)$; and the resulting value of $\texttt{CC}(G + e)$ as a function of $\lambda = \texttt{CC}(G)$:*

$\texttt{CC}(G - \{x, y\})$	$\texttt{CC}(G - \{x\})$	$\texttt{CC}(G - \{y\})$	$\texttt{CC}(G + e)$
λ			λ
$\lambda - 2$			$\lambda - 1$
$\lambda - 1$	λ	$\lambda - 1$	λ
$\lambda - 1$	$\lambda - 1$	λ	λ

Moreover, the optimal clique cover of $G + e$ can be computed in linear time using the optimal clique covers of the subgraphs above.

Proof. We consider the following cases of the assertion:

1. $CC(G - \{x, y\})$ equals to $CC(G)$. As $e = (x, y)$, it holds that $\texttt{CC}(G + e) \geq \texttt{CC}(G - \{x, y\})$ (as any clique cover for $G + e$ is one for $G - \{x, y\}$).
2. $CC(G - \{x, y\}) = CC(G) - 2$. Adding the nodes $\{x, y\}$ plus the edge (x, y) increases $CC(G)$ by 1, because the new clique cover consists of the cliques of $CC(G - \{x, y\})$ with the addition of the clique (x, y). The new clique cover is $(CC(G) - 2) + 1 = CC(G) - 1$. The new clique cover (after adding the edge (x, y)) can't be less than or equal to $CC(G) - 2$ because of Property 2.2 (adding an edge to a graph can decrease the clique-cover by 1 at most).
3. $CC(G - \{x\}) = CC(G)$. As in the first case, it holds that $\texttt{CC}(G + e) \geq \texttt{CC}(G - \{x\})$ (as any clique cover for $G + e$ is one for $G - \{x\}$).
4. Same as 3.

□

Claim 5.3. *Let G be an outerplanar graph that includes vertices x and y but does not include the edge $e = (x, y)$ that sits on the outer face of G. Then if $\texttt{CC}(G - \{x, y\}) = \texttt{CC}(G - \{x\}) = \texttt{CC}(G - \{y\}) = \texttt{CC}(G) - 1$ the optimal clique cover of $G + e$ is determined by the following cases. (a) If there are no vertices z in $G + e$ such that x, y, z from a triangle (a clique) in $G + e$ then $\texttt{CC}(G + e) = \texttt{CC}(G)$. (b) If there exists a vertex z in $G + e$ such that x, y, z from a triangle (a clique) in $G + e$ then $\texttt{CC}(G + e)$ depends on $\texttt{CC}(G - \{x, y, z\})$. Namely, if $\texttt{CC}(G - \{x, y, z\}) = \texttt{CC}(G) - 2$ then $\texttt{CC}(G + e) = \texttt{CC}(G) - 1$, otherwise $\texttt{CC}(G + e) = \texttt{CC}(G)$. Moreover, the optimal clique cover of $G + e$ can be computed in linear time using the optimal clique covers of the subgraphs above.*

In the claim above we use the fact that in case (b), when the algorithm of Baker [2] computes the clique cover of $\texttt{CC}(G)$ the optimal clique covers of $G - \{x, z\}$, $G - \{y, z\}$ and thus also $G - \{x, y, z\}$ are known. We note that the clique cover of G after adding the edge (x, y) can be equal to $\texttt{CC}(G)$ or $\texttt{CC}(G) - 1$, for example:

- If $x - z - y$ is a path, then adding the edge (x, y) closes a triangle and the clique cover is decreased by 1.
- If $x - z - u - v - y$ is a path, then adding the edge (x, y) doesn't decrease the clique cover.

Proof. We consider the following cases:

Case 1: There is no node $'z'$ such that x, y, z close a clique. Consider the clique cover of G of size $\texttt{CC}(G)$. Assume that adding the edge (x, y) decreases $\texttt{CC}(G)$ by 1. The minimum clique cover of $G + e$ must use the edge e and contain the clique $\{x, y\}$, otherwise, the new edge (x, y) doesn't have any influence and it can't change the

value of CC. Here we use the fact that any clique containing $\{x, y\}$ must be of size 2 at most. Thus, the new graph without the clique $\{x, y\}$ contains $\texttt{CC}(G) - 2$ cliques, in contradiction to the assumption that $\texttt{CC}(G - \{x, y\}) = \texttt{CC}(G) - 1$.

Case 2: There is a node $'z'$ such that x, y, z close a triangle. According to Baker's method, (x, z) and (z, y) are childs of the node (x, y), meaning that before finding the value of $\texttt{CC}(G(x, y))$ the procedure "merge" was performed with (x, z) and (z, y). The following values were known in this process: $\texttt{CC}(G(x, z))$, $\texttt{CC}(G(x, z) - \{x, z\})$, $\texttt{CC}(G(z, y))$, $\texttt{CC}(G(z, y) - \{z, y\})$. We conclude that the clique cover of G without nodes x, y, z is known and equals $\texttt{CC}(G(x, z) - \{x, z\}) + \texttt{CC}(G(z, y) - \{z, y\})$ (the subgraphs $G(x, z)$ and $G(y, z)$ excluding the nodes x, z, y are disjoint graphs because of the outerplanarity constraint). If this clique cover equals $\texttt{CC}(G) - 2$, then adding the clique $\{x, y, z\}$ causes the clique cover of $G + e$ to be of size $\texttt{CC}(G) - 1$ (the clique cover is decreased by 1). If this clique cover equals $\texttt{CC}(G) - 1$, then adding the edge (x, y) can't decrease the clique cover size. This follows as the new clique cover must contain a clique including the set $\{x, y\}$ (otherwise the added edge e doesn't have any influence), and can be assumed without loss of generality to contain $\{x, y, z\}$. Thus, the clique cover of the remaining graph (without $\{x, y, z\}$) equals $\texttt{CC}(G) - 2$ in contradiction to our assumption. □

6 Proof of Theorem 1.1

We now prove Theorem 1.1 given in the Introduction and show that $\texttt{MR}_\Sigma G = \texttt{CC}(G)$ in outerplanar graphs (and that both objective functions can be computed efficiently). In order to find the MinRank or the clique cover for outerplanar graphs, we processed a tree $\bar{G}$ (defined in Section 3) that represents the structure of G. We then used Baker's algorithm [2] to calculate for each tree vertex and corresponding subgraph G_1 of G the solution to the corresponding objective function. These computations were based on the "merge" and "adjust" operations studied in Sections 4 and 5. As the analysis for the MinRank and minimum clique cover given in Sections 4 and 5 are analogous, it follows by induction on the execution of Baker's algorithm that for any intermediate vertex in the tree $\bar{G}$ corresponding to a subgraph G_1 it holds that $\texttt{MR}_\Sigma(G_1) = \texttt{CC}(G_1)$. In particular, this also holds for G itself.

For the base case, consider a *level 1* vertex in the tree, i.e. a leaf v representing an edge (x, y). The table of v specifies that the size of the MinRank or the clique cover is 1 if exactly one endpoint of (x, y), or both of them are in the corresponding subgraph, and 0 if neither x nor y are in the subgraph. Thus, for a leaf v, its table values are equal for both the clique cover and MinRank problems. Assume that the statement is true for level $j < k$ vertices in the tree $\bar{G}$. The table for a level k vertex is calculated according to tables of level j vertices, using the procedures "merge" and "adjust" analyzed in Sections 4 and 5. As the claims states in Sections 4 and 5 are completely equivalent (given the change in the objective function), the inductive assertion follows. All in all, for any level k vertex, the calculation of table(v) depends only on the tables of level $j < k$ vertices. By assumption, these tables are equal for MinRank and clique cover and thus procedures "merge" and "adjust" give the same results for the same input. We conclude that MinRank and clique cover are equal for a level k vertex as well. This suffices to conclude our proof. The efficiency of our calculations follow directly from our constructive proofs.

7 Conclusions

The results of this work focus on information graphs G which are undirected and on the case in which the encoding functions are (scalar) linear. The problem of efficiently computing the index coding round complexity for encoding functions which are not scalar linear but rather *vector linear* or non-linear is left open in this work. The connection between the clique cover of G and the scalar linear index coding round complexity holds also for directed side information graphs as well. However, in the directed case it is not hard to find examples in which these two differ (e.g., any directed cycle). Finally, the side information graph model does not suffice to represent the index coding problem in which clients c_i require not a single message but multiple ones. In this case one should introduce a hypergraph side information model (e.g., [1]), which does not fit into the framework discussed in this work.

References

[1] N. Alon, E. Lubetzky, U. Stav, A. Weinstein, and A. Hassidim. Broadcasting with side information. *In Proceedings of 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 823–832, 2008.

[2] B. S. Baker. Approximation Algorithms for NP-complete Problems on Planar Graphs. *J. ACM*, 41(1):153–180, 1994.

[3] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol. Index Coding with Side Information. *In Proceedings of 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 197–206, 2006.

[4] Yitzhak Birk and Tomer Kol. Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients. *IEEE Trans. Inform. Theory*, 52(6):2825–2830, 2006. An earlier version appeared in INFOCOM 1998.

[5] M. E. Dyer and A. M. Frieze. Planar 3DM is NP-Complete. *Journal of Algorithms*, 7:174–184, 1986.

[6] S. El Rouayheb, A. Sprintson, and C. Georghiades. On the relation between the index coding and the network coding problems. In *IEEE International Symposium on Information Theory (ISIT)*, pages 1823–1827, July 2008.

[7] W. H. Haemers. On Some Problems of Lovász Concerning the Shannon Capacity of a Graph. *IEEE Transactions on Information Theory,*, 25(2):231–232, 1979.

[8] H. B. Hunt III, M. V. Marathe, V. Radhakrishnan, and R. E. Stearns. The complexity of planar counting problems. *SIAM J. Comput.*, 27:1142–1167, August 1998.

[9] M. Langberg and A. Sprintson. On the hardness of approximating the network coding capacity. In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pages 315–319, July 2008.

[10] A. Rasala Lehman. *Network Coding*. Ph.d. thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2005.

[11] E. Lubetzky and U. Stav. Non-linear Index Coding Outperforming the Linear Optimum. *In Proceedings of 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 161–168, 2007.

[12] R. Peeters. Orthogonal Representations Over Finite Fields and the Chromatic Number of Graphs. *Combinatorica*, 16(3):417–431, 1996.