ON THE COMPLEXITY OF CERTAIN COMPLETION PROBLEMS

by

S. B. Horton*, T. Easton† and R. Gary Parker†

Abstract

Many combinatorial problems that are efficiently solvable are made more difficult by the imposition of a partial solution. For example, while the celebrated four-color theorem guarantees that every planar graph can be four colored, if some of the vertices are colored as part of the instance, deciding if the remaining vertices can be properly colored with a total of four colors is $\mathcal{NP}$-complete. Interestingly, this phenomenon can also occur when the unconstrained problem is very easy or even trivial. In this regard, this paper gives some recent results for completion problems related to linear arrangement and latin square construction.

*Department of Mathematical Sciences, United States Military Academy. West Point, NY 10996. This research is supported in part by a grant from the Faculty Development and Research Fund, Office of the Dean, United States Military Academy.
† School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA. 30332-0205.

1 INTRODUCTION

1.1 Completion Problems

Let Π be a well-solved problem in the usual sense ($i.e., \Pi \in \mathcal{P}$) and let the set of instances be given by D_Π. Now, consider a modification to Π created by allowing an instance $I \in D_\Pi$ to be "restricted" in the sense that some subset of variables in *potential* candidate solutions to I are fixed. Then the sort of question that interests us here asks if there is an admissible completion for I subject to the stated, fixed conditions. Note that we include optimization problems in the obvious way where completions, if relevant, are evaluated against some threshold value that is part of I.

It is easy to create *completion problems* that fit within the framework posed above. Hereafter denoted by Π^F, some of these are natural, some less so. In addition, some cases are not so interesting while others can lead to results that are surprising. In this paper, we consider four examples of the latter variety.

1.2 Organization

The paper is organized in the following way. In the next section, we give some complexity results for certain completion problems. In the first subsection, we prove some complexity results related to the *optimal linear arrangement* problem. Following, we prove some similar results for a related graph labeling problem, the *bandwidth* problem. We then describe some outcomes regarding the complexity of completing *latin squares.* After which, various graph coloring completion problems are considered. We conclude the paper with a summary and directions for further research.

2 SOME COMPLEXITY RESULTS

2.1 Optimal Linear Arrangement

Given a simple, finite graph $G = (V, E)$ of order n, the OPTIMAL LINEAR ARRANGEMENT problem (OLA) seeks a vertex labeling $f : V \to \{1, 2, \ldots, n\}$ such that $\sum_{(u,v)\in E} |f(u) - f(v)|$ is minimum over all such labelings. For ease, let us denote the value of an admissible labeling of a graph G by $L(G)$. Optimal labelings are denoted by f^* and their values by L^*. OLA is well-known to be NP-hard in general but solvable on trees following work reported in Shiloach (1979) and more recently, in Chung (1984). The problem is also solved on the class of *outerplanar* graphs; planar graphs without subgraphs homeomorphic to K_4 or $K_{2,3}$ (Frederickson, *et al.* (1988)).

An interesting modification to OLA results if we assume that some (possibly empty) subset $\overline{V} \subseteq V$ is pre-labeled from integers in $\alpha \subseteq \{1, 2, \ldots, n\}$ and the notion now is to label the remaining vertices in $V \backslash \overline{V}$ with the other, "unused" labels and to do so in an optimal way overall, given the fixed labeling initially imposed. The decision version of the problem is simply to determine if there exists a labeling $f^*(G)$ satisfying the constraints of the pre-labeled vertices such that $L^*(G) \leq k$. Calling this version PARTIAL ARRANGEMENT (OLA^F), it is not at all clear what its status is, even for graph classes where it is trivial to solve OLA. For example, if G is a simple path, P_n we do not know how to solve the partial arrangement version, but neither do we have an $\mathcal{NP}$-hardness outcome. On the other hand, if we are allowed to further restrict and/or relax instances, we can produce some results.

For a solvable case of OLA^F for arbitrary graphs, suppose we take G to be any graph on n vertices and further, let us assume that G has no edges between vertices in $V \backslash \overline{V}$. Then, for such instances, OLA^F is easily solved as a weighted bipartite matching problem. We simply form a (complete) bipartite graph $G = (A, B, E)$ where $A = V \backslash \overline{V}$ (the unlabeled vertices) and $B = \{1, 2, \ldots, n\} \backslash \alpha$ (the set of available labels). For $i \in A$ and $j \in B$, let w_{ij} be the weight on edge (i, j) which we define to be the arrangement cost of assigning label j to vertex i.

Alternately, with different restrictions we can produce negative results. First we consider the case where G is a forest of paths.

Theorem 1: *Given a graph G that is the disjoint union of simple paths, OLA^F is $\mathcal{NP}$-complete.*

Proof: The problem is clearly in $\mathcal{NP}$; given a labeling it is easy to test if its value satisfies the threshold, k.

Our reduction is from the 3-PARTITION problem, the statement of which appears below:

Given a set A of $3m$ elements, and integer $B \in Z^+$, and an integer size $s(a) \in Z^+$ for each $a \in A$ such that $\frac{B}{4} < s(a) < \frac{B}{2}$ and where $\sum_{a \in A} s(a) = mB$, can A be partitioned into m disjoint sets $A_1, A_2, \ldots, A_m$ such that for $1 \leq i \leq m$, $\sum_{a \in A_i} s(a) = B$?

From an instance of 3-PARTITION we create an instance of OLA^F as follows. Let $k = 2m(B-1)$. The instance graph G consists of a disjoint union of $3m+1$ paths. The first $3m$ paths correspond to the elements of A and have length $s(a_j)$ for each respective $a_j \in A$. The last path has length $m+1$, and each vertex in this path has a fixed label. The ith vertex in this path has label $1 + i(B+1)$ for $i = 0, 1, \ldots, m$. An illustration of the construction is shown in Figure 1; the vertices in $\overline{V}$ are embedded across the top of the figure with their respective fixed labels indicated.

($\Rightarrow$) Suppose there exists a suitable partition relative to the instance for 3-PARTITION. Among the integers from 1 to $1 + m(B+1)$ inclusive, there are

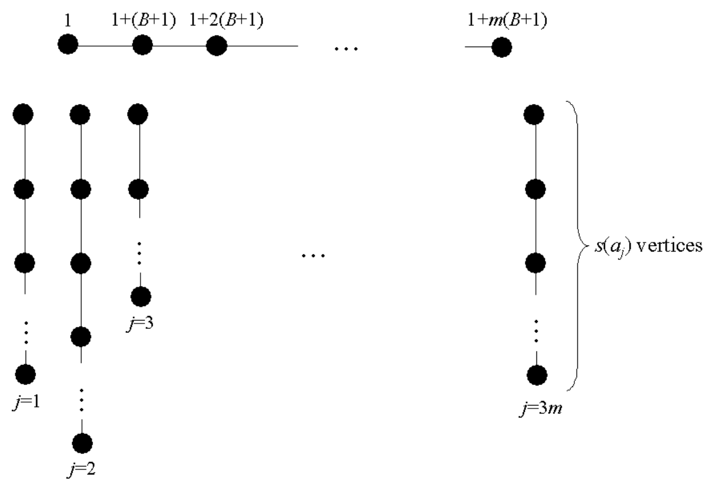

Figure 1: Reduction Mapping

m groups of B consecutive integers that are not used as fixed labels. Now, we use a set of B consecutive integers to label the B vertices of G corresponding to some A_i. We label these three paths in the obvious optimal way, incurring a cost of $\sum_{a \in A_i}(s(a) - 1)$. We do this for each of the m different A_i's. This yields a total labeling having a cost of

$$\sum_{a \in A}(s(a) - 1) + m(B + 1) = mB - 3m + m(B + 1) = 2m(B - 1)$$

so $L^*(G) \leq k$ as required.

($\Leftarrow$) Now suppose G can be labeled so that $L^*(G) \leq k$. We will show that this is possible only when there is a suitable partition that solves 3-PARTITION. First observe that

$$L^*(G) \geq m(B + 1) + \sum_{a \in A}(s(a) - 1) = m(B + 1) + mB - 3m = k$$

so we can assume $L^*(G) = k$. First note that the path that is partially labeled is actually completely labeled, and its cost is $m(B + 1)$. Further, note that every other path in G corresponding to $a \in A$ must be labeled using $s(a)$ consecutive integers if a value $L^*(G) = k$ is to be achieved. But since the partial labeling leaves “blocks” of B consecutive integers, it is clear that the condition $L^*(G) = k$ can only occur if there is a suitable partition relative to the instance of 3-PARTITION. □

To demonstrate the reduction, consider an instance of 3-PARTITION with $m = 3$, $B = 19$, and $A = \{5, 5, 5, 5, 6, 7, 7, 8, 9\}$. This is a “yes” instance since we can take $A_1 = \{5, 5, 9\}$, $A_2 = \{5, 7, 7\}$, and $A_3 = \{5, 6, 8\}$. For each i, the subgraph corresponding to A_i is labeled with one of the available sets of consecutive integers. The labeled graph that results is shown in Figure 2. The horizontally embedded path contains the fixed labels (1,21,41, and 61) and the vertically embedded paths correspond to the elements of A. The graph in the figure is completely labeled to satisfy $L^*(G) = k = 2m(B - 1) = 108$ according to the strategy described above.

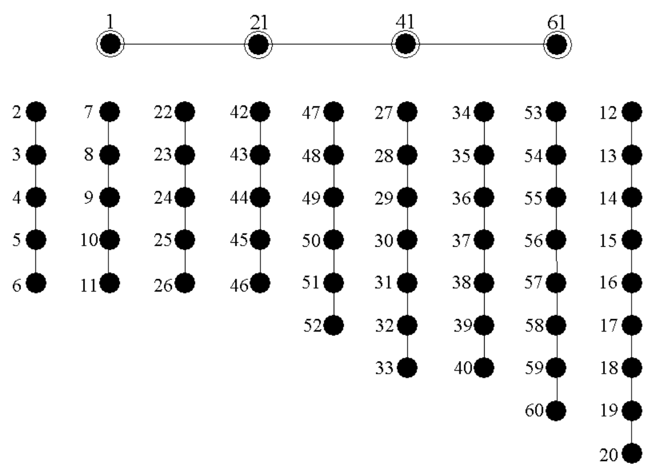

Figure 2: OLA Reduction Mapping Example

The following corollaries are immediate.

Corollary 2: *Let G satisfy the conditions of Theorem 1. Then OLA^F remains hard for* $\frac{|\overline{V}|}{|V|} \le \frac{m+1}{mB+m+1}$. □

For large values of m and B, this means a very small ratio (in the limit about $\frac{1}{B}$) of fixed labels to total vertices can be enough to make the problem difficult.

Corollary 3: *Let G satisfy the conditions of Theorem 1. Then OLA^F remains hard when only one of the paths contain members of $\overline{V}$.* $\square$

We also note that by employing a similar reduction strategy (*i.e.,* from 3-PARTITION), we can create modifications that establish the $\mathcal{NP}$-completeness of some related problems. Again, our intention is that we preserve the desired effect of having problems which are trivially solvable without fixed labels but which are not so otherwise.

Theorem 4: *OLA^F is $\mathcal{NP}$-complete when G is a caterpillar, i.e. a tree such that the removal of all the degree-1 vertices results in a path.* $\square$

Theorem 5: *OLA^F is $\mathcal{NP}$-complete when (the multigraph) G is a path with multiple edges.* $\square$

Theorem 6: *Given a graph $G = (V, E)$ that is a disjoint union of paths, an integer k, and a subset $\overline{V} \subseteq V$, deciding if G is a spanning subgraph of some connected graph $G' = (V, E')$ such that $L^*(G') \leq k$ is $\mathcal{NP}$-complete.* $\square$

Proofs of Theorems 4 and 5 appear in Easton (1999), and Theorem 6 is due to Horton (1997).

2.2 Bandwidth

In this section, we give some results for the BANDWIDTH problem that are similar in spirit to those of the previous section. Given a simple, finite graph $G = (V, E)$ of order n, the BANDWIDTH problem seeks a vertex labeling $f : V \rightarrow \{1, 2, \ldots, n\}$ such that $\max_{(u,v)\in E} |f(u) - f(v)|$ is minimum over all such labelings. We again denote the value of an admissible labeling of a graph G by $L(G)$. Optimal bandwidth labelings are denoted by f^{**} and their values by L^{**}. As is the case with OLA, the decision version of BANDWIDTH is $\mathcal{NP}$-complete. Unlike OLA, BANDWIDTH remains hard when the instances are restricted to be trees with no vertex of degree greater than 3. Note however that BANDWIDTH is similar to OLA in that it is trivial to solve when G is a forest of paths. We again modify the basic problem by assuming that some (possibly empty) subset $\overline{V} \subseteq V$ is pre-labeled from integers in $\alpha \subseteq \{1, 2, \ldots, n\}$ and our aim is to label the remaining vertices in $V \backslash \overline{V}$ with the other, "unused" labels and to do so in an optimal way overall, given the fixed labels of the initial labeling. The decision version of the problem is simply to determine if there exists a labeling $f^{**}(G)$ satisfying the constraints of the pre-labeled vertices such that $L^{**}(G) \leq k$. We will call this problem PARTIAL BANDWIDTH (BW^F). As was the case with OLA^F, weighted bipartite matching efficiently solves instances of BW^F in which there are no edges between vertices in $V \backslash \overline{V}$.

Next, we sketch the proof of a theorem for BW^F that is analogous to Theorem 1 for OLA^F.

Theorem 7: *Given a graph G that is the disjoint union of simple paths, BW^F is $\mathcal{NP}$-complete.*

Outline of Proof: The problem is clearly in $\mathcal{NP}$; given a labeling it is easy to test if its value satisfies the threshold, k.

Our reduction is again from the 3-PARTITION problem. From an arbitrary instance of 3-PARTITION we create an instance of BW^F as follows. Let $k = 1$. The instance graph G consists of a disjoint union of $3m$ paths and $m + 1$ isolated vertices. As in the reduction in Theorem 1, the $3m$ paths correspond to the elements of A and have length $s(a_j)$ for each respective $a_j \in A$. The $m + 1$ isolated vertices each have a fixed label; for $i = 0, 1, \ldots, m$, the ith vertex has label $1 + i(B + 1)$. The construction looks just like the illustration shown in Figure 1 except the horizontally embedded edges should be ignored. Now with a line of reasoning very similar to that of Theorem 1, it is easy to see that the resulting graph can be labeled so that $L^{**}(G) = 1$ if and only if there is a 3-partition of the integers from the instance of 3-PARTITION. $\Box$

Figure 3 demonstrates this reduction using the same example that was used in section 2.1. The graph is completely labeled to satisfy $L^{**}(G) = k = 1$.

The following corollaries are immediate.

Corollary 8: *Let G satisfy the conditions of Theorem 7. Then BW^F remains hard for $\frac{|\overline{V}|}{|V|} \leq \frac{m+1}{mB+m+1}$.* $\Box$

Corollary 9: *Let G satisfy the conditions of Theorem 7. Then BW^F remains hard when only isolated vertices have fixed labels.* $\Box$

It is interesting to compare Corollary 9 with Corollary 3. While the latter shows OLA^F is hard on a forest of paths with fixed labels restricted to one component, the former demonstrates that BW^F is hard for such graphs even when the instance has *no* edges incident with vertices with fixed labels.

Corollary 10: *Let G satisfy the conditions of Theorem 7. Then BW^F remains hard for $k = 1$.* $\Box$

Therefore, it is hard to determine if a partially labeled forest of paths can be completely labeled to achieve $L^{**}(G) = k = 1$.

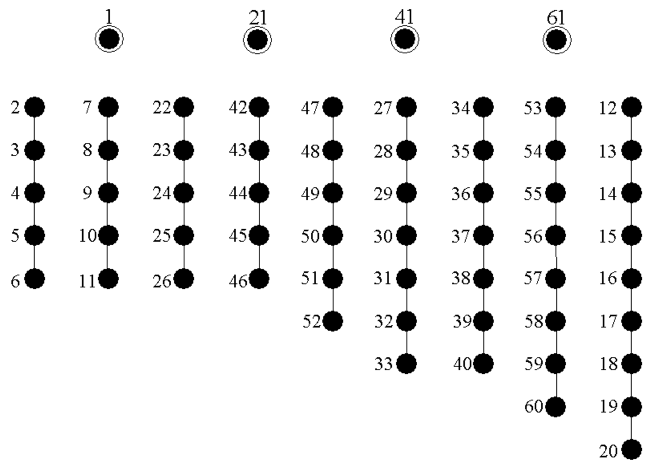

Figure 3: BW^F Reduction Mapping Example

2.3 Latin Squares

Latin squares have been extensively studied, because they arise in a variety of combinatorial design settings including problems in group theory, statistics, scheduling theory, *etc*. Interested readers are directed to Colbourn and Dinitz (1996) for a more extensive list of applications, problems and results.

A *latin square* of order n is an $n \times n$ array with each cell containing an element from the set $\psi = \{1, 2, \ldots, n\}$ and where each row and column of the array contains each element in ψ exactly once. An easy way to create a latin square is to simply fix in the first row, any permutation of the integers $1, 2, \ldots, n$ and then cyclically permute this row in consecutive rows 2 through n. Clearly this can be accomplished in polynomial time.

A solution to a latin square consists of an assignment of integers to cells. Thus, a partially completed latin square is a latin square with some cells prelabeled. Naturally, we want to know the complexity status of completing a partially filled square. For ease, we will refer to this as a *latin square completion problem* (LS^F).

In 1984, Colbourn proved that a LS^F is $\mathcal{NP}$-complete. Here we give a strengthened result showing that it remains $\mathcal{NP}$-complete even if there are at most 3 blank cells in any row or column and the only integers missing are 1, 2 and 3. This outcome coupled with the following polynomial-time case completely specifies the complexity status of completing partial latin squares.

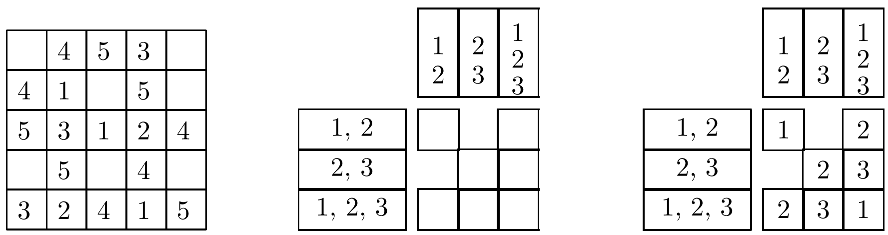

Figure 4: A partially completed latin square, its framework, and a completion of the framework

Let S be a partially completed latin square with at most 2 blank cells in any row and column. Create a graph $G = (V, E)$ with the vertex set corresponding to the blank cells with two vertices adjacent if they have either the same a row or column in S. Clearly, G consists of disjoint paths and cycles. Now solve a restricted vertex coloring problem (each vertex can be colored by at most 2 colors) for each component. If no coloring exists for at least one component, then S is not completable. Otherwise, an admissible labeling is generated. Solving such a restricted graph coloring problem can easily be accomplished in linear time.

Keeping with the expository nature of this paper, we will not give detailed proofs of all of the theorems and lemmas presented hereafter. Still, we will provide a sufficient amount of insight so that the reader can provide the details. Each of the proofs presented in this section can be found in their entirety in Easton and Parker (2000).

In establishing the complexity result, we will employ a notion and hence some language that coincides with that used in Colbourn's paper: the concept of a *latin square framework* (hereafter called a framework). Specifically, a framework is identified with blank cells of a partially completed latin square. In each row and column of the implied framework, certain elements are missing. Accordingly, let us create lists of these missing elements that are identified with the respective rows and columns of the framework. These lists will assist with bookkeeping and represent which elements can be placed in the blank cells of a particular row (column). Figure 4 contains a partially completed latin square, the corresponding framework with its row/column lists, and an admissible labeling of the framework.

Some observations regarding frameworks are in order. Naturally, the number of cells in a row (column) corresponds exactly to the number of elements in the list of that row (column). Second, the number of appearances of an element, say a, in the row lists equals the number of appearances of a in the column lists for every $a \in \psi$. Finally, and of central importance, there exists a completion to a partially completed latin square instance if and only if there exists a completion to the corresponding framework.

We now establish a particularly key outcome.

Lemma 11: *Completing a latin square framework with at most 3 cells in any row or column is $\mathcal{NP}$-complete even if the only elements are 1, 2 and 3.*

Outline of Proof: We employ a reduction from the $\mathcal{NP}$-complete problem that we will refer to as MONOTONE ONE-IN-THREE 3SAT (MO3-SAT) (Schaefer (1978)) which is specified formally as follows:

Given a set of literals $U = \{u_1, \ldots, u_n\}$, a collection $C = \{c_1, \ldots, c_m\}$ of clauses over U such that $|c_k| = 3$ and no $c_k \in C$ contains a negated literal for $k = 1, \ldots, m$, does there exist a truth assignment such that exactly one literal is true in each $c_k \in C$?

Our aim is to construct a latin square framework L from an instance of MO3-SAT. To this end, let p_i denote the total number of appearances of literal u_i in C for $i = 1, \ldots, n$. Now we construct two elemental frameworks given by U_i and C_k which will be copied a number of times in the construction of L. The U_i and C_k frameworks correspond to literals u_i and clauses c_k, respectively.

Each U_i framework has $8p_i$ rows and $9p_i$ columns where $i = 1, \ldots, n$ (see Figure 5 for the case when $p_i = 3$). Observe that this structure is formed from an 8×9 framework that is repeated p_i times (Figure 5 uses bold lines to denote these repetitions). These repetitions are placed along the "main" diagonal with one column of overlap. Further, the last of these 8×9 frameworks contains a blank cell in row $8(p_i - 1) + 6$, column 1 instead of column $9(p_i - 1) + 10$. Finally, observe that every row and column has the same number of cells as elements in the respective list except for rows and columns of the form $8(q - 1) + 2$ and $9(q - 1) + 9$ for $q = 1, \ldots, p_i$ respectively. The other two cells will be added in the construction of the C_k frameworks.

Now, in order to insure that there is no row or column overlap between any pair of frameworks U_i and U_j for $i \neq j$ we will place the U_i structure along the main diagonal of L beginning in the upper left hand corner. With this, we complete the portion of L that is associated with the literals of the MO3-SAT instance.

We now turn to the construction of C_k frameworks (the general pattern depicted in Figure 6 will be instructive in this regard). Without loss of generality, we may assume that clause c_k contains the r^{th} appearance of literal u_i, the s^{th} appearance of u_j, the t^{th} appearance of u_l and that $i < j < l$. We place blank cells in row $8(r - 1) + 2$ of framework U_i in both the first and second columns of the constructed C_k framework. Similarly, blank cells are also located in both the first and second row of the C_k framework in column $9(r - 1) + 9$ of the U_i framework. The missing cells for the U_j and U_l frameworks are dealt with similarly; again, the structure depicted in Figure 6 should clarify.

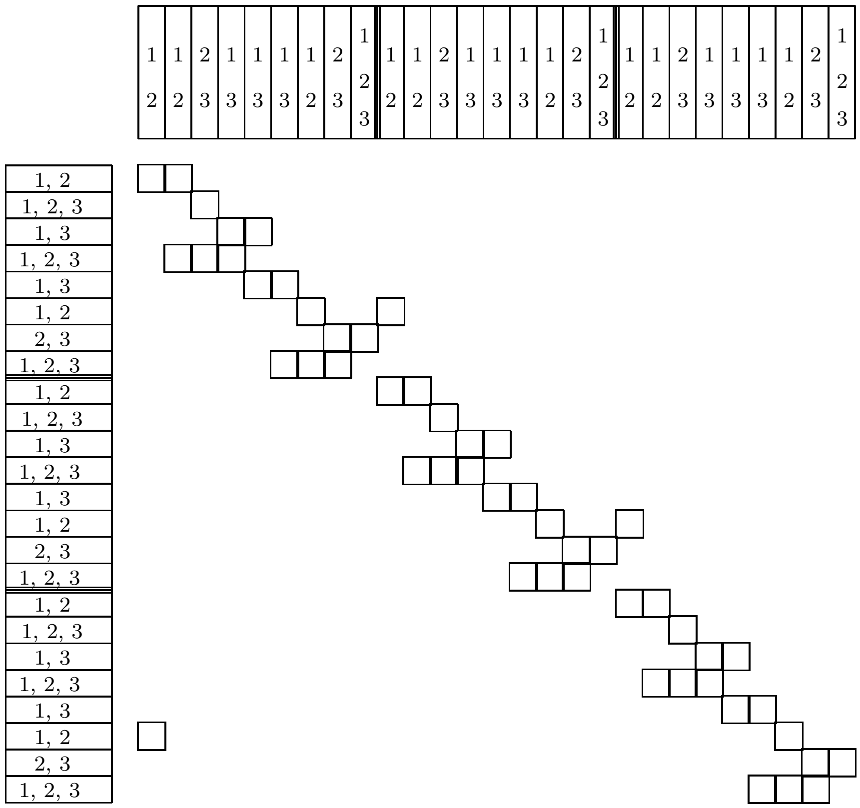

Figure 5: The U_i framework if $p_i = 3$

One C_k framework is created for each clause c_k and these are placed so that there is no column or row overlap between any pair C_k and C_l for $k \neq l$. Furthermore, no C_k and U_i frameworks overlap with the exception of the cells specified previously. This can be assured by also placing the C_k frameworks along the main diagonal of L and with this, the creation of the entire framework L is now complete. A conforming framework L is shown in Figure 7 for the simple instance $C = c_1 = (u_1 \wedge u_2 \wedge u_3)$. Note that we have again denoted the separation between the U_1, U_2, U_3 and C_k frameworks by bold lines. In addition, we have labeled L with a feasible labeling as this will help with the remainder of the proof.

In every row (column) of L the number of cells in each row (column) is at most 3 and is identical to the number of labels in the respective lists. The construction producing L is polynomial in the size of the MO3-SAT instance and so it remains to show that C can be satisfied if and only if L can be completed.

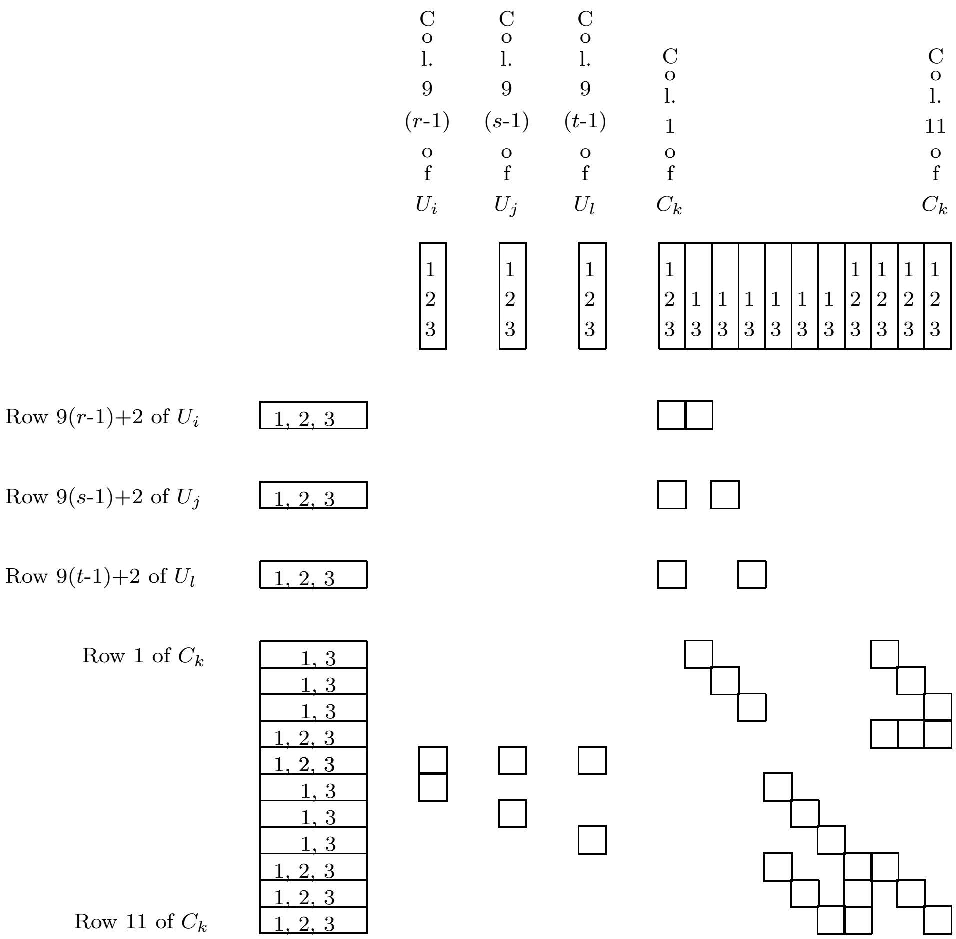

Figure 6: An example of the C_k framework with $u_i, u_j, u_l \in c_k$

($\Rightarrow$) Suppose that there exists an assignment to the literals, say T such that exactly one literal in each c_k is true for $k = 1, \ldots, m$. Now, if u_i is *true* in T, then label the cell in row 1, column 1 of U_i with the integer 1. With this, the remaining cells in U_i have to take on specific values. A sample of this labeling is in Figure 7 in the U_1 framework for $p_i = 1$. The reader can make the extension for other p_i values. Important is that every cell in row $8(q-1)+2$, column $9(q-1)+3$ and also row $8(q-1)+7$, column $9(q-1)+9$ is labeled with a 3 for all $q = 1, \ldots, p_i$ and the appropriate $k \in \{1, \ldots, m\}$.

If u_i is *false* in T, then label the cell in row 1, column 1 of the U_i framework with 2. Once again this forces the labeling of the remainder of the cells in U_i. An example can be found in the U_2 or U_3 frameworks in Figure 7. Observe, that the two cells mentioned in the preceding paragraph are now forced to be

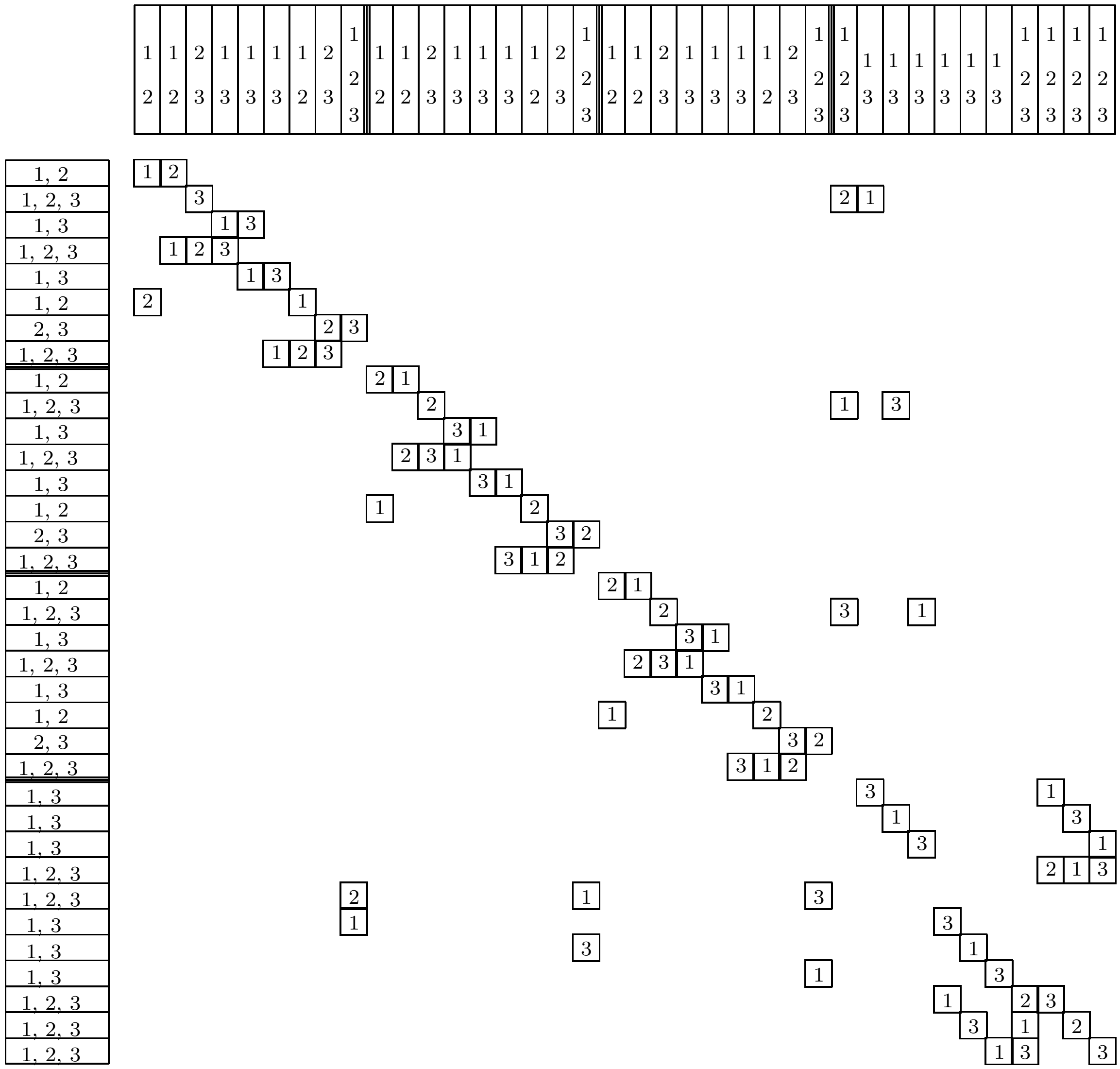

Figure 7: An example of a labeled L if $C = c_1 = \{u_1, u_2, u_3\}$

labeled 2.

This labeling scheme is performed for every literal. The outcome is that the only cells remaining unlabeled in L are those present in the C_k frameworks. Accordingly, assume that clause c_k contains an appearance of u_i, u_j and u_l. Then since exactly one literal satisfies each c_k, there are three possible labelings of the C_k framework. A sample labeling of C_k is given in Figure 7 (the C_1 framework) if u_i is *true*, u_j is *false*, and u_l is *false*. The reader can easily modify this labeling for the other two cases. Finally, adopting one of these concludes the labeling process; all cells in L are admissibly labeled using only and all the available labels for each row (column) and the first direction is complete.

($\Leftarrow$) Conversely, suppose that there is a completion to the framework L. Then we can begin by examining a single C_k framework of L. Without loss of generality, let us assume that clause c_k contains the r^{th} appearance of u_i, the s^{th} appearance of u_j, and the t^{th} appearance of u_l. Column 1 of the C_k framework must contain the labels 1, 2 and 3. We will only examine the cells in row $8(r-1)+2$ of the U_i framework; the cells in row $8(s-1)+2$ of the U_j framework and $8(t-1)+2$ of the U_l framework follow similarly. Three cases exist.

Case 1: The cell in row $8(r-1)+2$ of the U_i framework, column 1 of the C_k framework is labeled 1. Then the cell in row $8(r-1)+2$ of the U_i framework, column 2 of the C_k framework cannot be labeled a 2 (see the column list) and hence must be labeled a 3. Therefore, the cell in row $9(r-1)+2$, column $9(r-1)+3$ of the U_i framework must be labeled 2 and then all of the cells in the U_i framework are forced to take on specific values. Set u_i to be *false*.

Case 2: The cell in row $8(r-1)+2$ of the U_i framework, column 1 of the C_k framework is labeled 2. Then clearly the cell in row $9(r-1)+2$, column $9(r-1)+3$ of the U_i framework is labeled 3 and as a consequence, all of the cells in the U_i framework are required to be specific values. Set u_i to be *true*.

Case 3: The cell in row $8(r-1)+2$ of the U_i framework, column 1 of the C_k framework is labeled 3. Then, the cell in row $9(r-1)+2$, column $9(r-1)+3$ of the U_i framework is labeled 2 and as before, all of the cells in the U_i framework are again forced to take on specific values. Set u_i to be *false*.

The above scheme is repeated for every $k \in \{1, \ldots, m\}$. Clearly, there is no inconsistency in the assignments (*i.e.* setting u_i to both *true* and *false*) to the literals because every cell in row $8(q-1)+2$ of U_i, column 1 of C_k must all be labeled 2 or all be labeled either 1 or 3 for $q = 1, \ldots, p_i$ and the relevant $k \in \{1, \ldots, m\}$.

Now, since the framework has an admissible completion, every column 1 of the C_k framework must contain a single 1 (equivalently one literal set to *false*), a single 2 (equivalently one literal set to *true*), and a single 3 (another literal set to *false*) for $k = 1, \ldots, m$. Therefore, the assignment given has exactly one literal true in every c_k for $k = 1, \ldots, m$. This completes the second direction.

Finally, given a candidate labeling, verification that it is admissible can be accomplished in polynomial time. This establishes membership in $\mathcal{NP}$ and the proof of the lemma is complete. $\Box$

The aim then is to create a partially completed latin square having a framework identical to that produced from Lemma 11. Fortunately, by employing a result from Ryser (1951) we can guarantee that for *any* framework L there exists a partially completed latin square containing L. The proof follows almost identically to a similar proof by Colbourn (1984), and once again the exact details can be found in Easton and Parker (2000). Formally, we have:

Theorem 12: *For every latin square framework L with r rows and c columns, there exists a latin square S of size $2\max\{r,c\} \times 2\max\{r,c\}$ which has L as* *a framework. Moreover, S can be constructed in polynomial time.* $\square$

We can now strengthen Colbourn's result with a short proof. Formally:

Theorem 13: *LS^F is $\mathcal{NP}$-complete even if at most 3 cells are unfilled in any row or column and only the integers 1, 2 and 3 are missing from the square.*

Proof: Given an arbitrary instance of MO3-SAT we simply create the latin square framework L as indicated in Lemma 11 after which we apply Theorem 12 to this framework to obtain a partially completed latin square S. Since L has at most 3 blank cells (using only 1, 2, and 3) in any row or column , so does S. Clearly, S is completable if and only if L is completable.

Constructions implied by both Lemma 11 and Theorem 12 require polynomial time and therefore, the construction of S is polynomial. A candidate solution for a partially completed latin square can be verified easily, so the desired result follows. $\square$

Now with the main result of Theorem 13 established, various useful corollaries and theorems easily follow.

Corollary 14: *Completing a partially filled latin square is $\mathcal{NP}$-complete even if the proportion of prelabeled space to total space is arbitrarily close to one* $(1-\frac{3}{n})$. $\square$

Taking an opposite perspective, we could also consider how sparse a partially completed latin square can be and still maintain an $\mathcal{NP}$-completness outcome. Accordingly, a well known result holds that if a partially completed latin square has at most n (the length of a side) filled cells, then determining whether it can be completed is decidable in polynomial time (Smetaniuk (1981)). Thus, an outcome with only a constant number of elements in every row and column would, in a sense, be the best attainable (if $\mathcal{P} \neq \mathcal{NP}$). In fact, we can obtain a result along these lines.

Theorem 15: *Completing an $n \times n$ partially completed latin square is $\mathcal{NP}$-complete even if there are, on average, no more than $3 + \frac{1}{n}$ prelabeled elements in any row or column.*

Outline of Proof: Let S be a $n \times n$ latin square derived Theorem 13. Now, create a partially completed latin square S' from S by adding several 3×3 latin squares (using the numbers 1 through 3) along the main diagonal. Add a sufficient number of these 3×3 latin squares so that the length of a side is n^3. To do this we must require that $|S|$ *mod* $3 \equiv 0$ (it is an exercise to show that this is always possible). It is also an exercise to prove that S is completable if and only if S' is completable. Thus, this construction of the latin square is also

$\mathcal{NP}$-complete. But the total number of prelabeled elements divided by the size of the square is bounded above by $\frac{n^2+3n^3}{n^3} = 3 + \frac{1}{n}$. $\square$

Obviously, Theorem 15 could be strengthened to produce an average which is even closer to 3, *i.e.,* create S' such that the length of a side is n^4. Then the average number of prelabled cells is no more than $3 + \frac{1}{n^2}$.

2.4 Graph Coloring

A natural source for completion problems arises in the context of traditional graph coloring; however, some care needs to be exercised in order that interest be preserved. In both edge and vertex coloring cases, the general problems are hard and so color-completion questions accordingly are not relevant to us here. On the other hand, when graph classes are restricted, certain coloring problems are easy and so examining complexity status when edges/vertices are precolored is meaningful. Classes that are most interesting in this regard are bipartite graphs (for both edge and vertex coloring cases) and planar graphs (vertex coloring).

2.4.1 Edge Coloring

Given a finite graph $G = (V, E)$, the *edge coloring problem* (EC) seeks the minimum number of colors required in order that each edge be colored and no two adjacent edges share the same color. Vizing (1964) showed that the minimum number of colors (the chromatic index) is either Δ or $\Delta + 1$ where Δ is the maximum vertex degree of the graph. Holyer (1981) proved that determining the actual value remains $\mathcal{NP}$-complete even on planar graphs with maximum degree 3. On the other hand, König (1916) showed that all bipartite graphs can be colored with exactly Δ colors. Moreover this coloring can be found in polynomial time.

A solution to EC is an assignment of colors to edges. Hence, if we denote a completion version by EC^F then the *edge color-completion problem*, EC^F assumes that some edges are precolored (with no more than Δ colors) and asks if the graph can be color-completed with at most Δ colors used overall.

First we note that if there are at most 2 uncolored edges incident to any vertex, then EC^F is solvable for *any* graph. Formally:

Theorem 16: *Given a graph G, EC^F is polynomial time solvable if each vertex has at most 2 incident uncolored edges.* $\square$

It is worth pointing out that this result is a bit trickier than one might imagine. Removing from G those edges that are precolored, leaves a subgraph with components that are either paths or cycles. The proof then involves a construction of a certain auxiliary digraph, one for each component of this subgraph. Within each such digraph, a particular source-sink directed path is sought. If this path exists, then the respective component will be colorable in a valid way; if not, then no such valid coloring (and hence color completion of G) is possible.

While Theorem 16 holds for arbitrary graphs, a general result regarding EC^F is not interesting to us because edge-coloring is hard in general. However, for arbitrary bipartite graphs, edge-coloring is easy as was indicated previously, so examining the status of EC^F possesses interest in our stated, completion context. Since every latin square can be viewed as an edge coloring of a $K_{n,n}$ (a complete bipartite graph on n and n vertices), the following is an obvious outcome from Theorem 13.

Corollary 17: *EC^F is $\mathcal{NP}$-complete if G is a complete bipartite graph, at most Δ colors are used, and every vertex has at most 3 uncolored incident edges.* $\square$

The above theorem does not bound Δ by a constant. Trivially, the best possible $\mathcal{NP}$-hardness outcome would be for $\Delta = 3$. A result along these lines is available from the problem known as *list coloring* (edge and vertex) (Vizing (1976), Erdos (1979)). Edge list coloring is defined as follows: Is there a coloring of the edges of a graph such that no two adjacent edges are colored the same color and each edge e is colored from a list of permitted colors $L(e)$. Observe that every EC^F problem is an edge list coloring problem, however; there exists list coloring problems which are not EC^F problems. In 1992, Kubale proved that list coloring bipartite graphs with degree at most 3 using only 3 colors is $\mathcal{NP}$-complete. Fortunately, this list coloring problem can be represented in an EC^F context. Therefore, determining whether a partially edge colored bipartite graph with max degree 3 can be colored with at most 3 colors is $\mathcal{NP}$-complete. However, using Theorem 13 we can strengthen this result by requiring that the bipartite graph also be 3-regular. Formally:

Theorem 18: *EC^F is $\mathcal{NP}$-complete if G is a 3-regular bipartite graph and at most $\Delta = 3$ colors are used.*

Proof: We employ Theorem 13 to generate a latin square S of size $n \times n$ with at most 3 blank cells in any row or column and only missing the integers 1, 2, and 3. Create a partially colored complete bipartite graph $G = K_{n,n}$ where edge $\{i, j\}$ is assigned color k, if cell i, j is labeled k in S. Naturally, if $\{i, j\}$ is blank, then edge $\{i, j\}$ is uncolored. Generate G' by removing from G all edges colored colors 4 to n. Since every row and column of S contains exactly one appearance of each of the labels 4 to n, G' is a 3-regular bipartite graph. Furthermore, G' has some edges precolored from colors 1, 2 and 3. It remains to show that G' is 3 edge colorable if and only if S is completable.

($\Rightarrow$) For the first direction, assume C is an edge coloring of G' using only 3 colors.

Label the blank cells in S according to the colors assigned to the corresponding edges by C. Trivially, this is an admissible labeling of S.

($\Leftarrow$) Conversely, assume that T is an assignment that completes S. Color any uncolored edges in G' according to the label supplied by T. By definition of a latin square, this coloring of G' is a valid coloring. Furthermore, only 3 colors are used as required.

Finally, the construction of G' is clearly polynomial. Furthermore, verifying a candidate solution to EC^F can easily be accomplished in polynomial time. $\square$

We have completely classified EC^F on bipartite graphs. That is, if each vertex has at most two incident uncolored edges, then the problem is polynomial time solvable whereas if the bound is relaxed to at most three uncolored incident edges, the problem is $\mathcal{NP}$-complete.

2.4.2 Vertex Coloring

The VERTEX COLORING problem (VC) is celebrated in graph theory. The aim is to find the minimum number of colors that can be assigned to vertices of a graph so that each vertex is colored and no two adjacent vertices share the same color. Garey, *et al.* (1976) proved that determining the actual value or chromatic number, is $\mathcal{NP}$-complete even for planar graphs with maximum degree 3. However, all bipartite graphs can be colored with 2 colors in $O|E|$ effort, and of course, all planar graphs can be colored with at most 4 colors as established in Appel, *et al.* (1977). The latter requires $O(n^2)$ steps as shown by Robertson *et al.* (1997).

Now, for VC the relevant "completion version" assumes that at most k colors have been used to precolor vertices. Then given this partial coloring, VC^F asks if a k-color completion of the graph is possible where k is meaningful for the particular graph class being considered.

Suppose G is a bipartite graph, *i.e.* 2-colorable. Then consistent with the color-completion theme, the natural question that arises is whether or not a partially k-colored bipartite graph can be color-completed using no more than k colors in total. Of course, the algorithm to decide the issue when $k = 2$ is immediate. We simply start with any precolored vertex and attempt to color the uncolored adjacencies in the obvious way. This process is well-defined in the sense that it will either terminate with a total, 2-coloring or stop with an inconsistency, whereupon we can safely conclude that a 2-color completion of the stated graph is not possible.

Note that if a bipartite graph is not k-color completable then it must at least be $k + 2$-color completable, which is easy to establish. That is, if the graph is not so completable, simply remove the portion that is colored; the remainder is certainly bipartite which is two colorable and the outcome is evident. In fact, we cannot do better than this in the sense that there are instances that when partially 2-colored, cannot be completed with 3 colors.

On the other hand, if we do not restrict k to 2, the status of VC^F changes substantially. As before, a result can be obtained from the list coloring literature. A vertex list coloring problem seeks a coloring of the vertices such that no two adjacent vertices are colored the same color and every vertex v is colored from a set $L(v)$ of admissible colors.

Unlike edge list colorings, vertex list colorings can always be expressed as a VC^F problem. In stating this, care needs to taken because if a vertex list coloring problem is limited to a specific class of graphs (*i.e.* 2-connected graphs), then a the graph of the corresponding VC^F problem may not be contained in the same class of graphs. To create a VC^F problem from a vertex list coloring problem simply attach precolored pendants to every vertex to reduce the available colors from which each vertex can be colored, creating the desire lists. Kubale (1992) showed that vertex list coloring a bipartite graph with at most 5 colors overall is $\mathcal{NP}$-complete. Clearly, adding these precolored pendants preserves bipartness. Therefore, we can obtain:

Theorem 19: *Given a bipartite graph G with some vertices precolored where at most 5 colors are used, then deciding whether G can be colored with at most 5 colors is $\mathcal{NP}$-complete.* $\Box$

We omit the proof since it is a trivial exercise to adapt Kubale's proof to our setting here. In any event, the outcome that deciding bipartite k-color completability is hard coupled with the easy outcome that such graphs are always $k+2$-color completable, begs the question regarding $k+1$ completability. The following result settles the matter:

Theorem 20: *Given a bipartite graph G with some vertices precolored where at most k colors are used, then deciding whether G can be colored with at most $k+1$ colors is $\mathcal{NP}$-complete.* $\Box$

Again, we omit the proof. The details are nearly identical to those in Theorem 19 with only a minor adjustment.

Suppose now that G is planar. Unlike the bipartite case, fixing k here and asking for a k-color completion possesses interest in at least one obvious setting, *i.e.* $k = 4$. That is, given a planar graph G and with some subset of vertices colored using no more that 4 colors, is G 4-color completable?

Theorem 21: *VC^F is $\mathcal{NP}$-complete on planar graphs and with $k = 4$.*

Proof: The reduction is from the $\mathcal{NP}$-complete problem of deciding 3-colorability on planar graphs (Garey, *et al.* (1976)). Let $G = (V, E)$ be any planar graph and let H denote the set of faces of G excluding the infinite face. Then create $G' = (V', E')$ where $V'(G') = V(G) \bigcup \{v_h : h \in H\}$ and $E'(G') = E(G) \bigcup \{\{u, v_h\} : u$ is on the border of the face $h, h \in H\}$. Pick a single color and assign it to each vertex in the set $\{v_h : h \in H\}$. Clearly, deciding whether G' is 4-color completable is equivalent to asking whether the original graph G can be 3 colored. VC^F is certainly in $\mathcal{NP}$ and so the theorem follows. $\Box$

3 SUMMARY

The four cases that we have described in the previous section are natural candidates for the "completion" analysis that forms the basis of this exposition. In each setting, the general problem is hard but when instances are restricted to various, special classes of structures, easy, even trivial algorithms exist. Interestingly then, is that even when confined to these primitive structures, the respective completion versions become hard. Most notable are those results suggesting that very little needs to be "fixed" in order to change complexity status.

Of course, further work would be worthwhile in terms of extending this overall exercise to other problems. Some of this has been done in the research reported in Easton (1999). Still, there remain sources where new (and interesting) cases may be found. Among these, likely contexts would include scheduling theory, integer programming, and related subdisciplines in combinatorial analysis. In the context of the problems considered in this paper, resolution of the open cases is also an evident direction of study. Prominent in this regard would be definitive results for OLA^F and BW^F on simple paths.

4 REFERENCES

[1] Adolphson, D. and T. C. Hu, "Optimal Linear Ordering," *SIAM J. Appl. Math*, **25**, 403-423 (1973).

[2] Appel, K. and W. Haken, "Every planar map is 4-colorable-1: Discharging," *Ill. J. Math.*, **21**, 429-490 (1977).

[3] Appel, K. and W. Haken, "Every planar map is 4-colorable-2: Reducibility," *Ill. J. Math.*, **21**, 491-567 (1977).

[4] Chung, F. R. K., "On Optimal Linear Arrangements of Trees," *Comp. & Maths. with Appls.*, **10**, 43-60 (1984).

[5] Colbourn, C., "The Complexity of Completing Partial Latin Squares," *Discrete Applied Mathematics*, **8**, 151-158 (1984).

[6] Colbourn, C. and J. H. Dinitz (editors), *The CRC Handbook of Combinatorial Designs*, CRC Press; Boca Raton, Florida, (1996).

[7] Diestel, R., *Graph Theory*, Springer-Verlag, New York (1997).

[8] Easton, T., "On Partial Completion Problems," Ph.D. Dissertation, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA (1999).

[9] Easton, T. and R. G. Parker "On Completing Latin Squares," to appear in *Discrete Applied Mathematics*.

[10] Easton, T., S. B. Horton and R. G. Parker, "A Solvable Case of the Linear Arrangement Problem on Halin Graphs," *Congressus Numerantium*, **119**, 3-17 (1996).

[11] Erdös, P., A. Rubin and H. Taylor, "Choosability in graphs," *Congressus Numerantium*, **26**, 125-157 (1979).

[12] Even, S. and Y. Shiloach, "NP-completeness of Several Arrangement Problems," Report No. 43, Department of Computer Science, Technion, Haifa, Israel (1975).

[13] Frederickson, G. N. and S. E. Hambrusch, "Planar Linear Arrangements of Outerplanar Graphs," *IEEE Transactions on Circuits and Systems*, **35**, 323-332 (1988).

[14] Garey, M. and D. Johnson, *Computers and Intractibility: A guide to the theory of $\mathcal{NP}$-completeness*, W.H. Freeman, San Francisco, (1979).

[15] Garey, M., D. Johnson and L. Stockmeyer "Some simplified $\mathcal{NP}$-complete graph problems," *Theor. Comput. Sci.*, **1** 237-267 (1976).

[16] Harper, L. H., "Optimal Numberings and Isoperimetric Problems on Graphs," *J. Combinatorial Theory*, **1**, 385-393 (1966).

[17] Holyer, I., "The $\mathcal{NP}$-completeness of edge-coloring," *SIAM J. Comp.*, **10** 718-720 (1981).

[18] Hopcroft, J.E. and R.M. Karp, "An $n^{\frac{5}{2}}$ Algorithm for Maximum Matching in Bipartite Graphs," *SIAM J. Computing*, **2** 225-231 (1973).

[19] Horton, S. B.,"The Optimal Linear Arrangement Problem: Algorithms and Approximation," Ph.D. Dissertation, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA (1997).

[20] König, D., "Über Graphen und ihre anwendung auf determinantentheorie und mengenlehre" (German), *Math. Ann.*, **77** 453-465 (1916).

[21] Kubale, M., "Some Results Concerning the Complexity of Restricted Colorings of Graphs," *Discrete Applied Mathematics*, **36** 571-582 (1992).

[22] Kumar, S. R., A. Russell, and R. Sundaram, "Approximating Latin Square Extensions," *Algorithmica*, **24** 128-138 (1999).

[23] Robertson, N., D. P. Sanders, P. D. Seymour and R. Thomas, "A new proof of the 4 colour theorem", *Journal Combin. Theory Ser. B.*, **70** 2-44 (1997).

[24] Ryser, H., "A Combinatorial Theorem with Application to Latin Rectangles," *Proc. Amer. Math. Soc.*, **2** 550-552 (1951).

[25] Schaefer, T.J., "The Complexity of Satisfiability Problems," *Proc. 10th Ann. ACM Symp. on Theory of Computing*, Assoc. Computing Machinery, New York, 216-226 (1978).

[26] Shiloach, Y.,"A Minimum Linear Arrangement Algorithm for Undirected Trees," *SIAM Journal of Computing*, **8** 15-32 (1979).

[27] Smetaniuk, B., "A New Construction on Latin Squares - I: A proof of the Evans Conjecture," *Ars Combin.*, **11** 155-172 (1981).

[28] Tuza, Z. and M. Voigt, "List Colorings and Reducibility," *Discrete Applied Mathematics*, **79** 247-256 (1997).

[29] Vizing, V., "On an estimate of the chromatic class of a P-graph," *Diskret. Analiz* (Russian), **3** 25-30 (1964).

[30] Vizing, V., "Vertex colorings with given colors," *Diskret. Analiz* (Russian), **29** 3-10 (1976).