

# Generalization of the Consecutive-ones Property <sup>c1</sup>

<sup>c1</sup> *fyi*: Synopsis  
title is  
"Generalization  
of Consecutive  
Ones Property  
of Binary  
Matrices"

*A THESIS*

*submitted by*

**ANJU SRINIVASAN**

*for the award of the degree of*

**MASTER OF SCIENCE *by Research***

*from the department of*

**COMPUTER SCIENCE AND ENGINEERING**

*at*

**INDIAN INSTITUTE OF TECHNOLOGY MADRAS**

Guindy, Chennai - 600036



**JANUARY 2012**

# TABLE OF CONTENTS

<b>ACKNOWLEDGEMENTS</b>	<b>i</b>
<b>ABSTRACT</b>	<b>iii</b>
<b>LIST OF TABLES</b>	<b>v</b>
<b>LIST OF FIGURES</b>	<b>vi</b>
<b>ABBREVIATIONS</b>	<b>vii</b>
<b>NOTATION</b>	<b>viii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Organization of the document . . . . .	1
1.2 Illustration of the problem . . . . .	1
1.2.1 Special case . . . . .	3
1.3 Basic preliminaries - general definitions and nomenclature . . .	5
1.4 Consecutive-ones Property Testing - a Brief Survey . . . . .	6
1.4.1 Matrices with COP . . . . .	6
1.4.2 Optimization problems in COP . . . . .	8
1.5 Application of COP in Areas of Graph Theory and Algorithms .	9
1.6 Generalization of COP - the Motivation . . . . .	9
1.7 Summary of New Results in this Thesis . . . . .	11
<b>2 Consecutive-ones Property – A Survey of Important Results</b>	<b>14</b>
2.1 COP in Graph Theory . . . . .	14
2.2 Matrices with COP . . . . .	17
2.2.1 Tucker’s forbidden submatrices for COP . . . . .	19
2.2.2 Booth and Lueker’s <i>PQ</i> tree – a linear COT algorithm .	21

2.2.3	<i>PQR</i> -tree – COP for set systems . . . . .	23
2.2.4	<i>PC</i> -tree– a generalization of <i>PQ</i> -tree . . . . .	25
2.3	Optimization problems in COP . . . . .	27
2.4	COP in Graph Isomorphism . . . . .	28
<b>3</b>	<b>Tree Path Labeling of Path Hypergraphs - the New Results</b>	<b>28</b>
3.1	Summary of Proposed Problems . . . . .	28
3.2	Preliminaries to new results . . . . .	30
3.2.1	Hypergraph Preliminaries . . . . .	30
3.3	Characterization of Feasible Tree Path Labeling . . . . .	33
3.4	Computing feasible TPL with special target trees . . . . .	40
3.4.1	Target tree is a Path . . . . .	40
3.4.2	Target tree is a $k$ -subdivided Star . . . . .	41
3.4.3	Description of the Algorithm . . . . .	43
3.5	TPL with no restrictions . . . . .	47
3.5.1	Finding an assignment of tree paths to a set system . . . . .	48
3.6	Complexity of Tree Path Assignment-A Discussion . . . . .	52
3.6.1	Consecutive Ones Testing is in Logspace . . . . .	52
<b>4</b>	<b>Conclusion</b>	<b>55</b>
<b>A</b>	<b>More proofs</b> <sup>c1</sup>	<b>57</b>
<b>REFERENCES</b>	<sup>c2</sup>	<b>29</b>

<sup>c1</sup> remove/modify

<sup>c2</sup> *minor*: (i) make names in bib file uniform style w.r.t. firstname/initials (ii) have back refs - i.e. pages that cite the item.

# LIST OF TABLES

1.1	Students and study groups in <i>Wallace Studies Institute</i> . . . . .	2
1.2	A solution to study group accomodation problem . . . . .	2
2.1	Graph matrices <a href="#">PLACEHOLDER</a> . . . . .	17
2.2	A brief history of COP research . . . . .	18

# LIST OF FIGURES

1.1	<i>Infinite Loop</i> street map. . . . .	4
1.2	<i>Infinite Loop</i> street map with study group routes allocated. . . . .	4
1.3	Solution to the student accommodation problem. . . . .	4
1.4	Matrices with and without COP. . . . .	5
1.5	Examples of $k$ -subdivided stars. (a) $k = 0$ (b) $k = 2$ . . . . .	12
2.1	Matrices defined in Def. 2.1.1 . . . . .	16
2.2	Tucker's forbidden subgraphs . . . . .	20
2.3	Tucker's forbidden submatrices . . . . .	20
2.4	An example for $PQ$ -tree . . . . .	22
2.5	$PC$ -treePLACEHOLDER IMGS [HM03, Dom08] . . . . .	26
3.1	(a) 8-subdivided star with 7 rays (b) 3-subdivided star with 3 rays . . . .	41

# CHAPTER 2

## Consecutive-ones Property – A Survey of Important Results

This chapter surveys several results that are significant to this thesis or to COP in general. These predominantly pertain to characterizations of COP, algorithmic tests to check for COP (COT), optimization problems on binary matrices that do not have COP and some applications of COP.

c1

c1 *important:* All have a few lines about organization of chapter

### 2.1 COP in Graph Theory

COP is closely connected to several types of graphs by way of describing certain combinatorial graph properties. There are also certain graphs, like convex bipartite graphs, that are defined solely by some of its associated matrix having COP. In this section we will see the relevance of consecutive-ones property to graphs. To see this we introduce certain binary matrices that are used to define graphs in different ways. While adjacency matrix is perhaps the most commonly used such matrix, Definition 2.1.1 defines this and a few more.

**Definition 2.1.1.** *Matrices that define graphs.* [Dom08, Def. 2.4]. Let  $G$  and  $H$  be defined as follows.  $G = (V, E_G)$  is a graph with vertex set  $V = \{v_i \mid i \in [n]\}$  and edge set  $E_G \subseteq \{(v_i, v_j) \mid i, j \in [n]\}$  such that  $|E_G| = m$ .  $H = (A, B, E_H)$  is a bipartite graph with partitions  $A = \{a_i \mid i \in [n_a]\}$  and  $B = \{b_i \mid i \in [n_b]\}$ .

- 2.1.1–i. *Adjacency matrix* of  $G$  is the symmetric  $n \times n$  binary matrix  $M$  with  $m_{i,j} = 1$  if and only if  $(v_i, v_j) \in E_G$  for all  $i, j \in [n]$ .
- 2.1.1–ii. *Augmented adjacency matrix* of  $G$  is obtained from its adjacency matrix by setting all main diagonal elements to 1, i.e.  $m_{i,i} = 1$  for all  $i \in [n]$ .
- 2.1.1–iii. *Maximal clique matrix* or *vertex-clique incidence matrix* of  $G$  is the  $n \times k$  binary matrix  $M$  with  $m_{i,j} = 1$  if and only if  $v_i \in C_j$  for all  $i \in [n], j \in [k]$  where  $\{C_j \mid j \in [k]\}$  is the set of maximal cliques of  $G$ .
- 2.1.1–iv. *Half adjacency matrix* of  $H$  is the  $n_a \times n_b$  binary matrix  $M$  with  $m_{i,j} = 1$  if and only if  $(a_i, b_j) \in E_H$ .

Now we will see in Definition 2.1.2 certain graph classes that is related to COP or CROP.

**Definition 2.1.2.** *Graphs that relate to COP.* [Dom08, Def. 2.5] Let  $G$  be a graph and  $H$  be a bipartite graph.

- 2.1.2–i.  $G$  is *convex-round* if its adjacency matrix has the CROP.
- 2.1.2–ii.  $G$  is *concave-round* if its augmented adjacency matrix has CROP. <sup>c1</sup> <sup>c2</sup>
- 2.1.2–iii.  $G$  is an *interval graph* if its vertices can be mapped to intervals on the real line such that two vertices are adjacent if and only if their corresponding intervals overlap <sup>c3</sup>.  $G$  is an interval graph if and only if its maximal clique matrix has COP [FG65] <sup>c3</sup>
  - a.  $G$  is a *unit interval graph* if it is an interval graph such that all intervals have the same length. <sup>c3</sup>
  - b.  $G$  is a *proper interval graph* if it is an interval graph such that no interval properly contains another. <sup>c3</sup> <sup>c4</sup>
- 2.1.2–iv.  $G$  is a *circular-arc graph* if its vertices can be mapped to a set of arcs on a circle such that two vertices are adjacent if and only if their corresponding arcs overlap. <sup>c5</sup>
- 2.1.2–v.  $H$  is *convex bipartite on columns (rows)* if its half adjacency matrix has COP on rows (columns).
- 2.1.2–vi.  $H$  is *biconvex bipartite* or *doubly convex* [YC95] if its half adjacency matrix has COP on both rows and columns.
- 2.1.2–vii.  $H$  is *circular convex* if its half adjacency matrix has CROP.

Interval graphs <sup>c5</sup> and circular-arc graphs have a long history in research. The interest around them is due to their very desirable property that several problems that are NP-hard <sup>c6</sup> on general graphs, like finding a maximum clique or minimum

<sup>c3</sup>This follows [GH64] which states that the maximal cliques of interval graph  $G$  can be linearly ordered such that for all  $v \in V(G)$ , cliques containing  $v$  are consecutive in the ordering [Gol04, Th. 8.1].

<sup>c3</sup>The set of unit interval graphs and the set of proper interval graphs are the same

<sup>c5</sup>[McC04] cites that the problem of recognizing interval graphs has significance in molecular biology. Interestingly, in the late 1950s, before the structure of DNA was well-understood, Seymour Benzer was able to show that the intersection graph of a large number of fragments of genetic material was an interval graph [Ben59]. This was regarded as compelling evidence that genetic information was somehow arranged inside a structure that had a linear topology which we now know to be true from the discovery of linear structure of DNA.

<sup>c1</sup> cite BHY00

<sup>c2</sup>*minor*: add CROP to glossary

<sup>c3</sup> cite Ben59, Haj57

<sup>c4</sup> cite rob69,gar07 in footnote. pg 33 dom

<sup>c5</sup>*pressing*: how is CO/ROP related?

<sup>c6</sup>*minor*: all NPC problems are NPH yes?

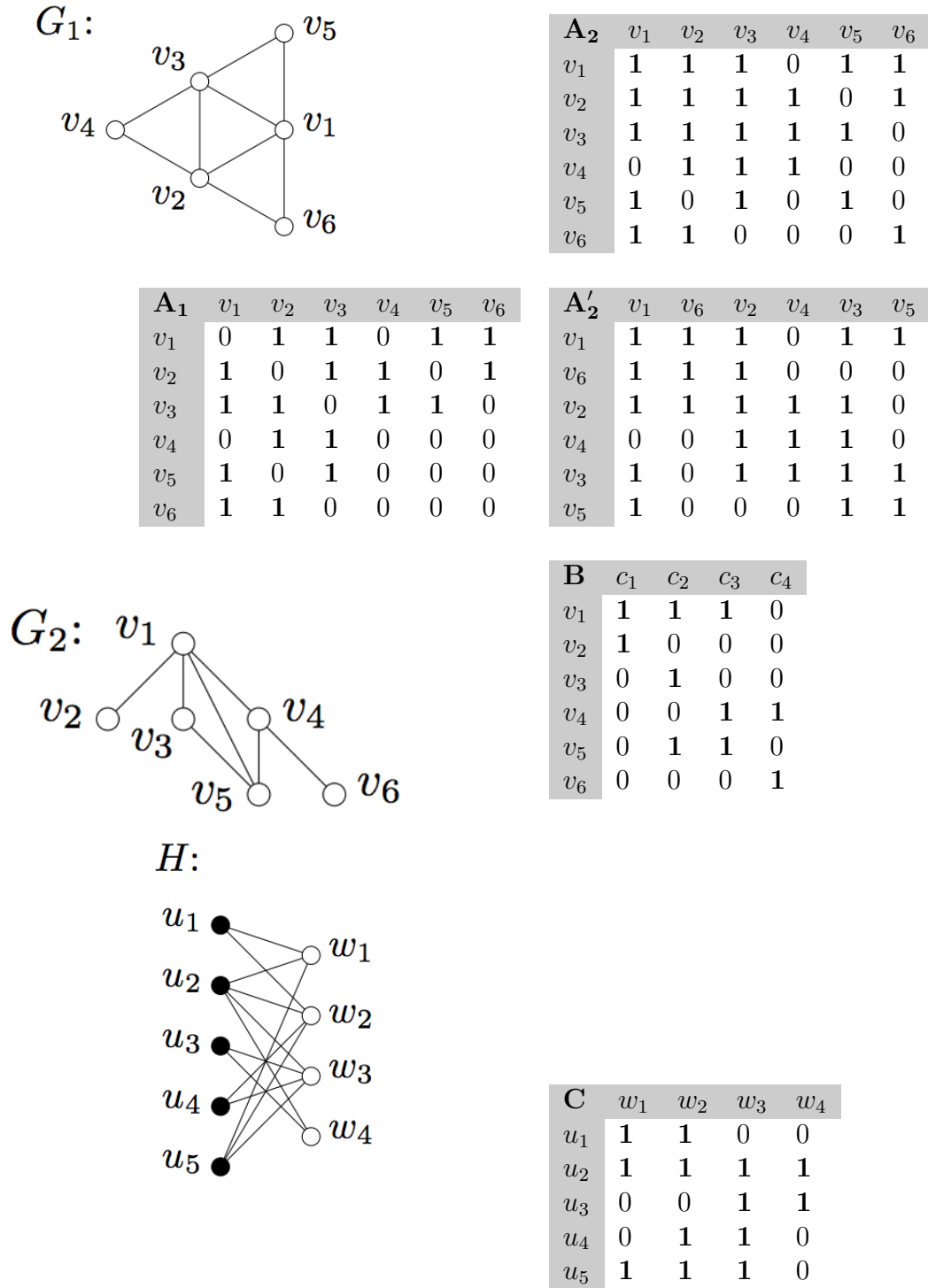


Figure 2.1:  $A_1$  is the *adjacency matrix* and  $A_2$  is the *augmented adjacency matrix* of  $G_1$ .  $A'_2$  is obtained from  $A_2$  by permuting its rows and columns to achieve *CROP order*, i. e.  $A_2$  has CROP – thus  $G_1$  is a *concave-round graph* (Def. 2.1.2 ii) and a *circular-arc graph* (Tab. 2.1)  $B$  is the maximal clique matrix of  $G_2$  and has COP – thus  $G_2$  is an *interval graph* (Def. 2.1.2 ii).  $C$  is the half adjacency matrix of bipartite graph  $H$  and has COP on rows – thus  $H$  is a convex bipartite graph. – [PLACEHOLDER IMAGES](#) –



coloring or independent set, are polynomial time solvable in these graph classes [CLRS01]. In a similar fashion, a lot of problems that are hard on general matrices have efficient solutions on matrices with COP or CROP [Dom08, more citations pg. 33].

Table 2.1 summarises the way these graphs are characterized by their matrices having COP or CROP. Our focus in this chapter (and thesis) is mainly COP and having seen how useful COP is in identifying or characterizing many types of graphs<sup>c0</sup>, we will now see results that study recognition of COP in matrices in the following section.

tab 2.1 - have an abridged version of table 2.1 in dom. see notes for the possibility of a “circle diagram”.

Table 2.1: Graph matrices [PLACEHOLDER](#)

## 2.2 Matrices with COP

The most important questions with respect to a particular property desired in a structure/object are perhaps the following.

- Does the desired property exist in the given input?
- If the test is affirmative, what is a certificate of the affirmative?
- If the test is negative, what are the optimization possibilities for the property in the input? In other words, how close to having the property can the input be?
- If the test is negative, what is a certificate of the negative?

In this section and the rest of the chapter we see results that shaped the corresponding areas respectively for consecutive-ones property in binary matrices.

- a. Does a given binary matrix have COP?
- b. What is the COP permutation for the given matrix with COP?
- c. What are the optimizations possible and practically useful on the given matrix without COP?
- d. If algorithm for (a) returns **false**, can a certificate for this be computed?

---

<sup>c0</sup>COP has several other applications [fill up](#).

Without doubt, besides computing answers to these questions, we are interested in the efficiency of these computations in terms of computational complexity theory. Results towards questions (a) and (b) are surveyed in this section. Those for question (c) are discussed in Section 2.3 and question (d) is discussed in Section 2.2.3.

It may be noted that one way to design an algorithm to test for COP is by deriving one from any interval graph recognition algorithm using the result HMPV00<sup>c1</sup> <sup>c2</sup> [Dom08] which demonstrates how such a derivation can be done. However, this does not necessarily yield an efficient algorithm. We will see results that directly solve the problem on matrices since it is known that questions (a) and (b) stated above for COP are efficiently solvable. Table 2.2 gives a snapshot of these results.

<sup>c1</sup> cite hmpv00  
<sup>c2</sup> in footnote put the theorem 2.7 - dom pg 43

<sup>c3</sup>

<sup>c3</sup> important: 18 by petrie - cite kendall 69 pacific journal of mathematics 1969

1899	First mention of COP (archaeology)	
1951	Heuristics for COT	[Rob51]
1965	First polynomial time algorithm for COP testing	[FG65]
1972	Characterization for COP- forbidden submatrices	[Tuc72]
1976	First linear time algorithm for COT - $PQ$ -tree	[BL76]
1992	Linear time algorithm COT without $PQ$ -tree	[Hsu02]
2001	$PC$ -tree - a simplification of $PQ$ -tree	[Hsu01, HM03]
1996	$PQR$ -tree - generalization of $PQ$ -tree for any binary matrix regardless of its COP status	[MM96]
1998	Almost linear time to construct $PQR$ -tree	[MPT98]
2004	A certifying algorithm for no COP. Generalized $PQ$ -tree.	[McC04]
2009	Set theoretic, cardinality based characterization of COP - ICPIA	[NS09]
2010	Logspace COP testing	[KKLV10]

Table 2.2: A brief history of COP research

The first polynomial time algorithm for COP testing was by [FG65] which uses overlapping properties of columns with 1s. Their result has close relations to the characterization of interval graphs by [GH64]. A graph  $G$  is an interval graph if and only if all its maximal cliques can be linearly ordered such that for any vertex  $v$  in  $G$ , all the cliques that  $v$  is incident on are consecutive in this order. Clearly, this means that the maximal clique incidence matrix<sup>c3</sup> must have COP on rows.

A few years later, a deeply significant result based on very different ideas in understanding COP came from Tucker which gave a combinatorial (negative) characterization of matrices with COP [Tuc72]. This result influenced most of the COP results that followed in the literature including linear time algorithms<sup>c4</sup> for COP recognition.

<sup>c4</sup> pressing: linear did it? which ones?

---

<sup>c3</sup> Definition 2.1.1 iii

### 2.2.1 Tucker's forbidden submatrices for COP

[Tuc72] discovered certain forbidden structures for convex bipartite graphs<sup>c4</sup> and by definition of this graph class, this translates to a set of forbidden submatrices for matrices with consecutive-ones property. The following are the theorems from [Tuc72] that achieved this characterization.

Theorem 2.2.1 states that convex bipartite graphs cannot have *asteroidal triples*<sup>c0</sup> contained in the corresponding vertex partition<sup>c0</sup>. Theorem 2.2.2 lists the structures in a bipartite graph that force one of its vertex partitions to have asteroidal triples – in other words, it identifies the subgraphs that prevent the graph from being convex bipartite.

**Theorem 2.2.1.** ([Tuc72, Th. 6], [Dom08, Th. 2.3])

A bipartite graph  $G = (V_1, V_2, E)$  is convex bipartite on columns<sup>c0</sup> if and only if  $V_1$  contains no asteroidal triple of  $G$ .

**Theorem 2.2.2.** ([Tuc72, Th. 7], [Dom08, Th. 2.4])

In a bipartite graph  $G = (V_1, V_2, E)$  the vertex set  $V_1$  contains no asteroidal triple if and only if  $G$  contains none of the graphs  $G_{I_k}$ ,  $G_{II_k}$ ,  $G_{III_k}$  (with  $k \geq 1$ ),  $G_{IV}$ ,  $G_V$  as shown in Figure 2.2 as subgraphs.

Theorem 2.2.1 and Theorem 2.2.2 result in the following Theorem 2.2.3 which characterizes matrices with COP.

**Theorem 2.2.3.** ([Tuc72, Th. 9], [Dom08, Th. 2.5])

A matrix  $M$  has COP if and only if it contains none of the matrices  $M_{I_k}$ ,  $M_{II_k}$ ,  $M_{III_k}$  (with  $k \geq 1$ ),  $M_{IV}$ ,  $M_V$  as shown in Figure 2.3 as submatrices.

It can be verified that the matrices in Figure 2.3 are the half adjacency matrices of the graphs in Figure 2.2 respectively which is not surprising due to Definition 2.1.2 v.

---

<sup>c4</sup>The terminology in [Tuc72] differs. It uses the term *graphs with  $V_1$ -consecutive arrangement* instead of *convex bipartite graphs*.

<sup>c0</sup>If  $G = (V, E)$  is a graph, a set of three vertices from  $V$  form an *asteroidal triple* if between any two of them there exists a path in  $G$  that does not contain any vertex from the closed neighborhood of the third vertex.

<sup>c0</sup>The partition corresponds to columns (rows) if its half adjacency matrix has COP columns (rows).

<sup>c0</sup>Abridged to match terminology adopted in this document. See previous note.

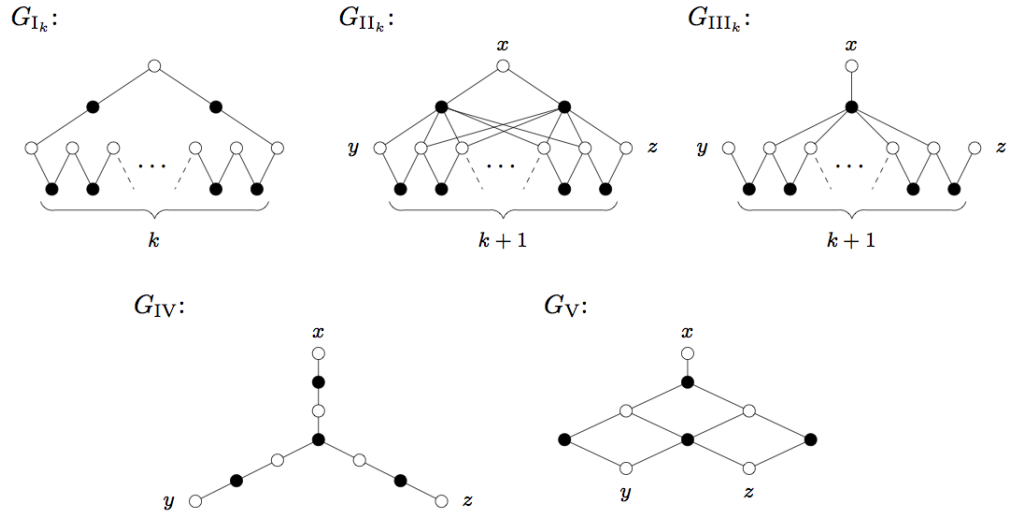


Figure 2.2: Tucker's forbidden subgraphs for convex bipartite graphs. [PLACEHOLDER IMG](#)

$$\begin{array}{cc}
 M_{I_k}, k \geq 1 & M_{II_k}, k \geq 1 \\
 \left( \begin{array}{ccccc} \overbrace{1 & 1 & 0 & \dots & 0}^{k+2} \\ 0 & 1 & 1 & 0 & \dots & 0 \\ & & \dots & & & \\ 0 & \dots & 0 & 1 & 1 \\ \hline 1 & 0 & \dots & 0 & 1 \end{array} \right)_{k+2} & \left( \begin{array}{ccccc} \overbrace{1 & 1 & 0 & \dots & 0}^{k+3} \\ 0 & 1 & 1 & 0 & \dots & 0 \\ & & \dots & & & \\ 0 & \dots & 0 & 1 & 1 & 0 \\ \hline 0 & 1 & \dots & 1 & 0 & 1 \\ 1 & \dots & 1 & 0 & 1 \end{array} \right)_{k+3} \\
 \\
 M_{III_k}, k \geq 1 & M_{IV} \\
 \left( \begin{array}{ccccc} \overbrace{1 & 1 & 0 & \dots & 0}^{k+3} \\ 0 & 1 & 1 & 0 & \dots & 0 \\ & & \dots & & & \\ 0 & \dots & 0 & 1 & 1 & 0 \\ \hline 0 & 1 & \dots & 1 & 0 & 1 \end{array} \right)_{k+2} & \begin{array}{ccccc} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{array} \\
 \\
 M_V \\
 \begin{array}{ccccc} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{array}
 \end{array}$$

Figure 2.3: Tucker's forbidden submatrices for convex bipartite graphs. [Tuc72]

## 2.2.2 Booth and Lueker's $PQ$ tree – a linear COT algorithm

Booth and Lueker in their paper [BL76] gave the first linear algorithm<sup>c0</sup> for consecutive-ones property testing while given a linear time interval graph recognition algorithm by a simplification of <sup>c1</sup>'s planarity test algorithm. [BL76] introduces a data structure called  $PQ$ -tree and their COP testing algorithm is a constructive one that outputs a  $PQ$ -tree if the input has COP. A  $PQ$ -tree represents all the COP orderings of the matrix it is associated with. [BL76]'s algorithm uses the fact that if a matrix has COP, a  $PQ$ -tree for it can be constructed. It is interesting to note that aside from interval graph recognition and COP testing,  $PQ$ -tree is also useful in other applications like finding planar embeddings of planar graphs [?, McC04] and recognizing CROP in a matrix.

<sup>c1</sup> cite - A. Lempel, S. Even and I. Cederbaum, An Algorithm for Planarity Testing of Graphs, Theory of Graphs, ed., P. Rosenstiehl, Gordon and Breach, New York, (1967), 215-232.

**Definition 2.2.1 ( $PQ$ -tree [BL76, McC04]).** A  $PQ$ -tree of matrix  $M$  with COP on columns (rows), is a tree with the following properties.

- i. Each leaf uniquely represents a row (column) of  $M$ . The leaf order of the tree gives a COP order for column (row)<sup>c0</sup> for  $M$ .
- ii. Every non-leaf node in the tree is labeled  $P$  or  $Q$ .
- iii. The children of  $P$  nodes are unordered. They can be permuted in any fashion to obtain a new COP order for  $M$ .
- iv. The children of  $Q$  nodes are linearly ordered. Their order can be reversed to obtain a new COP order for  $M$ .

See Figure 2.4 for an example of  $PQ$ -tree. It may be noted that there is no way an empty set of COP orderings can be represented in this data structure. For this reason,  $PQ$ -tree is undefined for matrices that do not have COP. Thus effectively, there exists a bijection between set of matrices with COP and the set of  $PQ$ -trees (accurately speaking, each matrix with COP bijectively maps to an equivalence class of  $PQ$ -trees resulting from properties (iii) and (iv)).

The [BL76] algorithm with input  $n \times m$  matrix  $M$  starts with a  $PQ$ -tree for a vacuous  $n \times 0$  matrix  $M'$  (submatrix induced by 0 columns). This is known as a *universal*  $PQ$ -tree which is one with its root as a  $P$  node and only leaves as its children – each leaf representative of a row of input (by definition of COP for

---

<sup>c0</sup>Time complexity is  $O(m + n + f)$  where  $m \times n$  is the order of the input matrix and  $f$  is the number of 1s in it.

<sup>c0</sup>Note that COP order for column requires permutation of rows and vice versa.

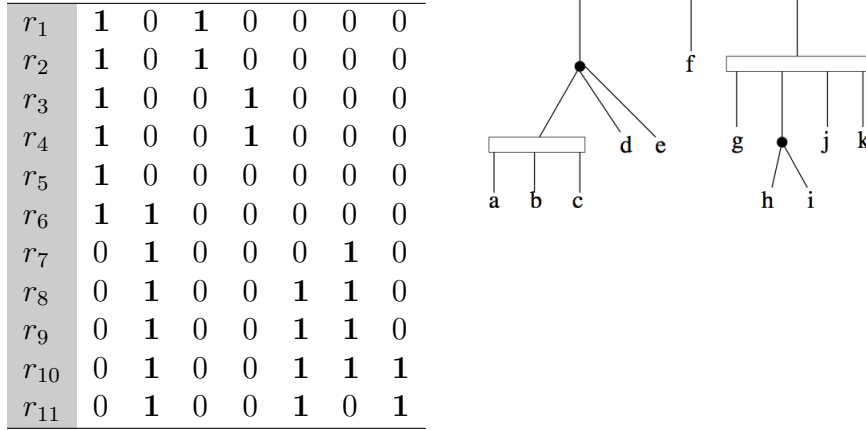


Figure 2.4: An example for  $PQ$ -tree. Permuting the order of the left child of the root, we see that  $(d, a, b, c, e, f, g, h, i, j, k)$  is a COP order. Reversing the order of the right child of the root, we see that  $(a, b, c, d, e, f, k, j, h, i, g)$  is yet another COP order. [PLACEHOLDER IMG.](#) [McC04, Fig. 1<sup>10</sup>]

columns). This induced submatrix  $M'$  vacuously has COP. Each column is then added iteratively to  $M'$  to check if the new  $M'$  has COP. By a complicated, but linear, procedure the algorithm does one of the following actions in each iteration: (a) declare that  $M$  has no COP, or (b) modify the current  $PQ$ -tree to represent the new  $M'$  (which clearly, must have COP, since if not, option (a) would have been executed).

Judging from notes in literature, this algorithm is apparently notoriously difficult to program. In the procedure to modify the  $PQ$ -tree at each iteration, nodes are considered from leaves to tree. At each node considered, it uses one of nine templates to determine how the tree must be altered in the vicinity of this node. Recognition of this template poses a difficult challenge in terms of implementing it. Each template is actually a representative of a larger class of similar templates, which must be dealt with explicitly by a program[McC04].

After the invention of  $PQ$ -trees, presumably due to the implementation challenge it posed, there has been several variants of the same in the literature, like  $PC$ -tree [SH99, Hsu01, HM03], generalized  $PQ$ -tree [KM89, McC04],  $PQR$ -tree [MM96, MPT98] etc. Most of these are generalizations of  $PQ$ -tree— for instance,  $PC$ -tree is generalized to matrices with CROP,  $PQR$ -tree and generalized  $PQ$ -tree are generalized to matrices and set systems with or without COP. [KM89] invented a modified form of  $PQ$ -tree a simpler incremental update of the tree only for recognizing interval graphs. [KR88] constructed efficient parallel algorithms for manipulating  $PQ$  trees. [dom chapter 2 pg 40 Variations of PQTrees first para - summarize.](#)

In the next few following sections we will see some of these variations.

### 2.2.3 *PQR*-tree – COP for set systems

Section 1.6 mentions how a binary matrix naturally maps to a system of sets. A set can be constructed for each column of matrix with its elements being those row indices at which the column has **1**s. Thus the collection of sets corresponding each column of the matrix forms a set system with universe as the set of all row indices of the matrix. This simple construction is formally described in Definition 2.2.2 along with the idea of consecutive-ones property for set systems<sup>c0</sup>.

**Definition 2.2.2.** *Consecutive-ones property for set systems* Let  $M$  be a binary matrix of order  $n \times m$  and  $\{c_i \mid i \in [m]\}$  be the columns in  $M$ . A set system  $\mathcal{F}_M = \{S_i \mid S_i \subseteq [n], i \in [m]\}$  is defined such that for every column  $c_i$  of  $M$ , set  $S_i = \{j \mid m_{ji} = \mathbf{1}\}$ . The collection  $\mathcal{F}_M$  is the *set system of binary matrix*  $M$ .

A set system  $\mathcal{F}$  from universe  $U = \{1, 2, \dots, n\}$  has the *consecutive-ones property* if it is possible to assign a linear order, i.e. a permutation  $\sigma = u_1 u_2 \dots u_n$  to  $U$  such that each set  $S \in \mathcal{F}$  appears as a consecutive subsequence  $u_i u_{i+1} \dots u_{i+k-1}$  (also called an interval) of  $\lambda$  in the linear order, where positive integers  $k = |S|$  and  $i \leq n + 1 - k$ .

It is easy to see the equivalence of this definition to COP for matrices in Definition 1.3.3.

Generalized *PQ*-tree or *gPQ*-tree is a data structure defined in [Nov89] to represent all orthogonal sets of a set system  $\mathcal{F}$ . *Orthogonal sets* of  $\mathcal{F}$  are sets that do not overlap with any set in  $\mathcal{F}$ . A data structure with the same name was later defined in [McC04] as part of a *substitution decomposition* for a set system  $\mathcal{F}$ .<sup>c1</sup> This is later used to define a certificate if  $\mathcal{F}$  has no COP. *PQR*-tree is a data structure defined in [MM96] as a data structure to represent any set system  $\mathcal{F}$  with additional information in their *R*-nodes if  $\mathcal{F}$  has no COP. All three of these data structures are proposed as generalizations of [BL76]’s *PQ*-tree and hence produce the *PQ*-tree if  $\mathcal{F}$  has COP. As a whole, all these three data structures are near-identical with their differences being notional. In this section, we will discuss the basic theory that they all hold and stick to the name of the data structure as *PQR*-tree.

A *PQR*-tree of a given set system  $\mathcal{F}$  from universe  $U$  represents all the sets from  $2^U$  that have trivial intersections with every set in  $\mathcal{F}$ . Two sets  $A$  and  $B$

---

<sup>c0</sup>As seen in Section 1.2.1

<sup>c1</sup> define  
substitutio  
decomp.

are said to have a *trivial intersection* if  $A \cap B$  is one of the following –  $\emptyset$ ,  $A$  or  $B$ . In other words,  $A$  and  $B$  are either (i) disjoint or (ii) one is the subset of the other. Such sets are also called *orthogonal sets* or said that  $A$  is orthogonal to  $B$ . *Trivial subsets* of a universe  $U$ , denoted by  $\mathcal{T}(U)$ , are sets that have trivial intersections with any set in  $2^U$ . Namely,  $\mathcal{T}(U)$  consists of  $U$ , singleton sets in  $2^U$  and  $\emptyset$  [Nov89, MM96]. In other words, the all sets presented in a *PQR-tree* share the property that they do not overlap with any set in  $\mathcal{F}$ . We will now see how this helps in testing for COP in  $\mathcal{F}$ .

An important observation made by [MM96] is that if  $\mathcal{F}$  is a set system with COP then, aside from every set in  $\mathcal{F}$  being consecutive (after applying the COP order) the following sets must also be consecutive for any  $A, B \in \mathcal{F}$ .

1. The intersection  $A \cap B$
2. The union  $A \cup B$  if  $A \cap B \neq \emptyset$
3. The relative complements  $A \setminus B$  and  $B \setminus A$  if  $B \not\subseteq A$  and  $A \not\subseteq B$  respectively.

Also note that trivially, sets in  $\mathcal{T}(U)$  are consecutive in any permutation of  $U^{c0}$ .

Any set system that is closed under the above operations and contains  $\mathcal{T}(U)$  is called *complete* [MM96] or a *weakly partative family* [McC04]. Such a set system is easier to work with for reasons that will become apparent.<sup>c1</sup> For any set system  $\mathcal{F}$ ,<sup>c1</sup> *VAGUE!* the smallest super set system that is complete/weakly partative is called the *weak closure* of  $\mathcal{F}$ , denoted  $\mathcal{W}(\mathcal{F})$  [McC04, Def. 3.2]. [MM96, Def. 2, smallest *complete supercollection*].

What is interesting is that the set of COP permutations of  $\mathcal{F}$  is the same as that of  $\mathcal{W}(\mathcal{F})$  [MM96, Th. 3]. The reason this is interesting is because it becomes apparent by this theory that it is easier to use  $\mathcal{W}(\mathcal{F})$  to create the *PQR-tree* for  $\mathcal{F}$ .

The generalized *PQ-tree* of  $\mathcal{F}$  is defined as the *decomposition tree* of its weak closure  $\mathcal{W}(\mathcal{F})$ . Decomposition tree is the inclusion tree (inclusion p.o. Hasse diagram) of the strong elements of  $\mathcal{W}(\mathcal{F})$  with labels *prime*, *linear* and *degenerate*. [McC04]

Overlap components are computable in linear time [MM95, Hsu92].

Certifying algorithm (Generalized PQ trees) [McC04] –

Set theoretic characterizations [Hsu02, NS09]  
[Hsu02] describes the simpler algorithm for COT.<sup>c2</sup>  
[NS09] describes a characterization of consecutive-ones property solely based on the cardinality

<sup>c2</sup> *pressing*: in terms of what?

---

<sup>c0</sup>  $\emptyset$  is considered consecutive by convention.



properties of the set representations of the columns (rows); every column (row) is equivalent to a set that has the row (column) indices of the rows (columns) that have one entries in this column (row). This is interesting and relevant, especially to this thesis because it simplifies COT to a great degree by reducing the solution search space owing to the a simple set theoretic characterization.

[McC04] describes a different approach to COT. While all previous COT algorithms gave the COP order if the matrix has the property but exited stating negative if otherwise, this algorithm gives an evidence by way of a certificate of matrix even when it has no COP. This enables a user to verify the algorithm's result even when the answer is negative. This is significant from an implementation perspective because automated program verification is hard and manual verification is more viable. Hence having a certificate reinforces an implementation's credibility. Note that when the matrix *has* COP, the COP order is the certificate. The internal machinery of this algorithm is related to the weighted betweenness problem addressed<sup>c1</sup> in [COR98].<sup>c2</sup>

c1 in what way??

c2 expand on the COP order graph creation and it having to be bipartite for M to have COP. and thus an odd cycle being an evidence of no COP.

## 2.2.4 *PC-tree*— a generalization of *PQ-tree*

*PC-tree* is a generalization of *PQ-tree*. It is a data structure that is analogous to *PQ-tree* but for matrices with circular-ones property. *PC-tree* was introduced by [SH99] for the purpose of planarity testing where this data structure represents partial embeddings of planar graphs. In [Hsu01], Hsu reintroduces *PC-tree* as a generalization of *PQ-tree* and shows how it simplifies [BL76]'s planarity test by making the *PQ-tree* construction much less complicated. Later [HM03], discovers that *PC-tree* is a representation of all circular-ones property orders of a matrix when it is unrooted. *PC-tree* presented in [Hsu01] is rooted; however the construction of *PC-tree* is the same in both results. The property of being unrooted is necessary in order to use *PC-tree* as a data structure for encoding circular ordering. Definition 2.2.3 defines *PC-tree*.

**Definition 2.2.3.**[*PC-tree* [Hsu01, Dom08].] A *PC-tree* of matrix  $M$  with CROP on columns (rows), is a tree with the following properties.

- i. Is unrooted – thus it has (a) no parent child relationship between nodes (b) there is no left to right (or vice versa) ordering.
- ii. Each leaf uniquely represents a row (column) of  $M$ . The leaf order of the tree gives a CROP order for column (row) for  $M$ . Moreover, any sequence obtained by considering the leaves in clockwise or counter-clockwise order describes a CROP order for  $M$ .
- iii. Every non-leaf node in the tree is labeled  $P$  or  $C$ .
- iv. The neighbors of  $P$  nodes can be permuted in any fashion to obtain a new CROP order for  $M$ .
- v. The tree can be changed by applying the following “mirroring” operation to obtain a new CROP order for  $M$ . Root the *PC-tree* at a neighbor of a  $C$ -node,  $v$  and mirror the subtree whose root is  $v$  and finally unrooting the tree. Mirroring a subtree is done by putting the children of every node of the subtree in reverse order.

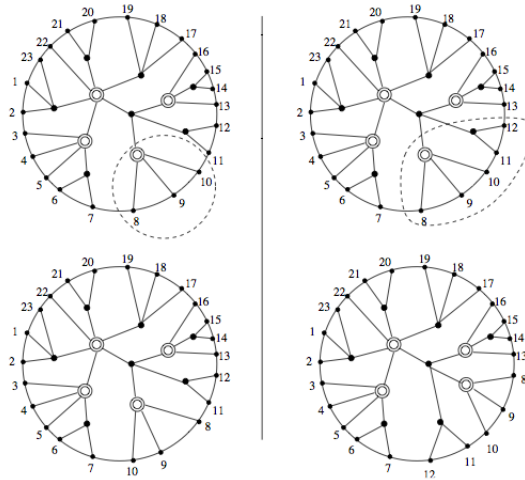


Figure 2: The PC tree can be viewed as a gadget for generating the circular-ones permutations of the columns. The C nodes are represented by double circles and the P nodes are represented by black dots. The subtree lying at one side of an edge can be flipped over to reverse the order of its leaves. The order of leaves of a consecutive set of subtrees that would result from the removal of a P node can also be reversed. All circular-ones arrangements can be obtained by a sequence of such reversals.

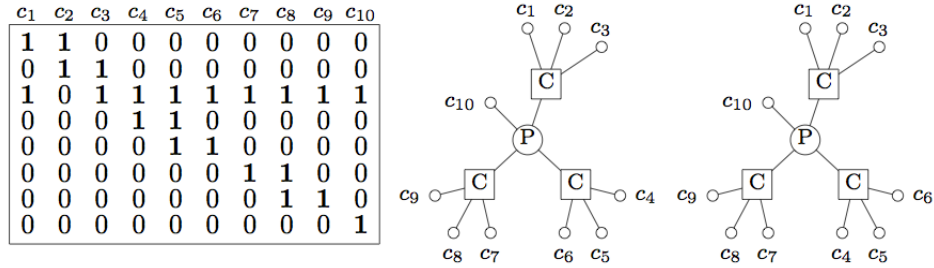


Figure 2.5: *PC*-tree [PLACEHOLDER IMGS \[HM03, Dom08\]](#)

As a data structure when  $PC$ -tree is compared with  $PQ$ -tree, the differences are, (i) it is unrooted, (ii) it represents all CROP order of a matrix (iii) it has  $C$  nodes instead of  $Q$  nodes which can be “mirrored” (operation defined in Definition 2.2.3 v). The algorithms of construction of  $PQ$ -tree in [BL76] and that of  $PC$ -tree in [Hsu01, HM03] starkly differ since the latter is a much simplified procedure.

## 2.3 Optimization problems in COP

c1

So far we have been concerned about matrices that have the consecutive ones property. However in real life applications, it is rare that data sets represented by binary matrices have COP, primarily due to the noisy nature of data available. At the same time, COP is not arbitrary and is a desirable property in practical data representation [COR98, JKC<sup>+</sup>04, Kou77]. In this context, there are several interesting problems when a matrix does not have COP but is “close” to having COP or is allowed to be altered to have COP. These are the optimization problems related to a matrix which does not have COP. Some of the significant problems are surveyed in this section.

c2c3 couple of lines referring to tucker’s submatrices. refer earlier section.

Once a matrix has been detected to not have COP (using any of the COT algorithms mentioned earlier), it is naturally of interest

1. to find out the smallest forbidden substructure (in terms of number of rows and/or columns and/or number of entries that are 1s). [Dom08] discusses a couple of algorithms which are efficient if the number of 1s in a row is small. This is of significance in the case of sparse matrices where this number is much lesser than the number of columns.
2.  $(*, \Delta)$ -matrices are matrices with no restriction on number of 1s in any column but has at most  $\Delta$  1s in any row. MIN COS-R (MIN COS-C), MAX COS-R (MAX COS-C) are similar problems which deals with *inducing COP* on a matrix.
  - (a) In the dual problem MAX COS-R (MAX COS-C) the search is for the maximum number of rows (columns) that induces a submatrix with COP.
  - (b) In MIN COS-R (MIN COS-C) the question is to find the minimum number of rows (columns) that must be deleted to result in a matrix with COP.

Given a matrix  $M$  with no COP, [Boo75] shows that finding a submatrix  $M'$  with all columns<sup>c4</sup> but a maximum cardinality subset of rows such that  $M'$  has COP is NP complete. [HG02] corrects an error of the abridged proof of this reduction as given in [GJ79]. [Dom08] discusses all these problems in detail giving an extensive survey of the previously existing results which are almost exhaustively all approximation results and hardness results. Taking this further, [Dom08] presents new results in the area of parameterized algorithms for this problem<sup>c5</sup>.

3. Another problem is to find the minimum number of entries in the matrix that can be toggled to result in a matrix with COP. [Vel85] discusses approximation of COP AUGMENTATION which is the problem of changing of the minimum number of zero entries to 1s so that the resulting matrix has COP. As mentioned earlier, this problem is known to be NP complete due to [Boo75]. [Vel85] also proves, using a reduction to the longest path problem,<sup>c6</sup> that finding a Tucker’s forbidden submatrix of at least  $k$  rows is NP complete.<sup>c7 c8</sup>
4. [JKC<sup>+</sup>04] discusses the use of matrices with almost-COP (instead of one block of consecutive 1s, they have  $x$  blocks, or *runs*, of consecutive 1s and  $x$  is not too large) in the storage of very large databases. The problem is that of reordering of a binary matrix such that the resulting matrix has at most  $k$  runs of 1s. This is proved to be NP hard using a reduction from the Hamiltonian path problem.<sup>c9 c10c11 c12 c13</sup>

c1 Expand on ref:sec:optcop

c2 – sect 4.1 in cite:d08phd has many results surveyed. hardness results, approx. results. results are usually for a class of matrices  $(a, b)$  where number 1s in columns and rows are restricted to  $a$  and  $b$ . – problem of flipping at most  $k$  entries of  $M$  to make it attain COP. this is NP complete cite:b75-phd

c3 (1) scite:lb62 showed that interval graphs are chordal and AT-free.

c4 check if b75 deals with COP col or COP row. also is it any submatrix with  $k$  less than  $r$  rows or submatrix must have all columns?

c5 elaborate - what are the results?

c6 or is it a survey of another result? check.

c7 how is this different from booth’s 75 result??

c8 where should this go? cite—tz04 (approx submatrix with COP sparse matrices)

c9 Theorem 2.1 in jkckv

c10 (1) A connection of COP problem

## 2.4 COP in Graph Isomorphism

c14 c15

c1

The survey from kklv10 conclusion.

c14 Expand  
on  
sec:appgraphiso

c15 (canonization  
theme)

c1 *pressing*: ADDE  
peo exists iff  
chordal.  
lexicographic  
BFS  
[tag:chordalGraph]

# REFERENCES

- [ABH98] J. E. Atkins, E. G. Boman, and B. Hendrickson. A spectral algorithm for seriation and the consecutive ones problem. *SICOMP: SIAM Journal on Computing*, 28, 1998.
- [ADP80] Giorgio Ausiello, Alessandro D’Atri, and Marco Protasi. Structure preserving reductions among convex optimization problems. *J. Comput. Syst. Sci.*, 21(1):136–153, 1980.
- [AS95] Annexstein and Swaminathan. On testing consecutive-ones property in parallel. In *SPAA: Annual ACM Symposium on Parallel Algorithms and Architectures*, 1995.
- [Ben59] S. Benzer. On the topology of the genetic ne structure. *Proc. Nat. Acad. Sci. U.S.A.*, 45:1607–1620, 1959.
- [BL76] Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using *PQ*-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, December 1976.
- [BLS99] Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. *Graph classes: a survey*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.
- [Boo75] Kellogg S. Booth. *PQ-tree algorithms*. PhD thesis, Univ. California, Berkeley, 1975.
- [BP92] J. R. S. Blair and B. Peyton. An introduction to chordal graphs and clique trees. Technical report, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831, 1992.
- [BS03] David A. Bader and Sukanya Sreshta. A new parallel algorithm for planarity testing, April 11 2003.
- [BTV09] Chris Bourke, Raghunath Tewari, and N. V. Vinodchandran. Directed planar reachability is in unambiguous log-space. *ACM Trans. Comput. Theory*, 1:4:1–4:17, February 2009.
- [CKL96] R. Chandrasekaran, S. N. Kabadi, and S. Lakshminarayanan. An extension of a theorem of Fulkerson and Gross. *Linear Algebra and its Applications*, 246(1–3):23–29, October 1996.
- [CLRS01] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*. McGraw Hill, Boston, MA, USA, 2001.
- [COR98] Thomas Christof, Marcus Oswald, and Gerhard Reinelt. Consecutive ones and a betweenness problem in computational biology. *Lecture Notes in Computer Science*, 1412, 1998.
- [CY91] Lin Chen and Yaacov Yesha. Parallel recognition of the consecutive ones property with applications. *J. Algorithms*, 12(3):375–392, 1991.
- [DF99] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.
- [DFH<sup>+</sup>06] Dujmovic, Fellows, Hallett, Kitching, Liotta, McCartin, Nishimura, Ragde, Rosamond, Suderman, Whitesides, and Wood. A fixed-parameter approach to 2-layer planarization. *ALGRTHMICA: Algorithmica*, 45, 2006.
- [DGN07] Michael Dom, Jiong Guo, and Rolf Niedermeier. Approximability and parameterized complexity of consecutive ones submatrix problems. In *Theory and Applications of Models of Computation, 4th International Conference, TAMC 2007, Shanghai, China, May 22-25, 2007, Proceedings*, volume 4484 of *Lecture Notes in Computer Science*, pages 680–691. Springer, 2007.

- [Dom08] Michael Dom. *Recognition, Generation, and Application of Binary Matrices with the Consecutive-Ones Property*. PhD thesis, Institut für Informatik, Friedrich-Schiller-Universität Jena, Germany, 2008. Published by Cuvillier, 2009.
- [Fei98] Uriel Feige. A threshold of  $\ln n$  for approximating set cover. *J. ACM*, 45(4):634–652, 1998.
- [Fer05a] H. Fernau. *Parameterized Algorithmics: A Graph-Theoretic Approach*. Habilitationsschrift, Universität Tübingen, Germany, 2005.
- [Fer05b] H. Fernau. Two-layer planarization: improving on parameterized algorithmics. In *SOFSEM*, volume 3381 of *LNCS*, pages 137–146. Springer, 2005.
- [Fer08] Henning Fernau. Parameterized algorithmics for linear arrangement problems. *Discrete Appl. Math.*, 156:3166–3177, October 2008.
- [FG65] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pac. J. Math.*, 15:835–855, 1965.
- [Gav78] Fanica Gavril. A recognition algorithm for the intersection graphs of paths in trees. *Discrete Mathematics*, 23(3):211 – 227, 1978.
- [GH64] P. C. Gilmore and A. J. Hoffman. A characterization of comparability graphs and of interval graphs. *Can. J. Math.*, 16:539–548, 1964.
- [GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability*. Freeman, San Francisco, 1979.
- [Gol04] Martin Charles Golumbic. *Algorithmic graph theory and perfect graphs*, volume 57 of *Annals of Discrete Mathematics*. Elsevier Science B.V., 2004. Second Edition.
- [HG02] Hajiaghayi and Ganjali. A note on the consecutive ones submatrix problem. *IPL: Information Processing Letters*, 83, 2002.
- [HL06] Dorit S. Hochbaum and Asaf Levin. Cyclical scheduling and multi-shift scheduling: Complexity and approximation algorithms. *Discrete Optimization*, 3(4):327–340, 2006.
- [HM03] Wen-Lian Hsu and Ross M. McConnell. PC trees and circular-ones arrangements. *Theoretical Computer Science*, 296:99–116, 2003.
- [Hsu92] Wen-Lian Hsu. A simple test for the consecutive ones property. In *Proc. of the ISAAC’92* [Hsu02], pages 459–469. Later appeared in *J. Algorithms* 2002.
- [Hsu01] Wen-Lian Hsu. PC-trees vs. PQ-trees. *Lecture Notes in Computer Science*, 2108:207–217, 2001.
- [Hsu02] Wen-Lian Hsu. A simple test for the consecutive ones property. *J. Algorithms*, 43(1):1–16, 2002.
- [HT98] T. C. Hu and P. A. Tucker. Minimax programs, bitonic columns and PQ trees, November 29 1998.
- [HT02] Hochbaum and Tucker. Minimax problems with bitonic matrices. *NETWORKS: Networks: An International Journal*, 40, 2002.
- [JJLM97] Michael Jünger, Michael Junger, Sebastian Leipert, and Petra Mutzel. Pitfalls of using PQ-trees in automatic graph drawing, 1997.
- [JKC<sup>+</sup>04] Johnson, Krishnan, Chhugani, Kumar, and Venkatasubramanian. Compressing large boolean matrices using reordering techniques. In *VLDB: International Conference on Very Large Data Bases*. Morgan Kaufmann Publishers, 2004.
- [JLL76] Neil D. Jones, Y. Edmund Lien, and William T. Laaser. New problems complete for nondeterministic log space. *MST: Mathematical Systems Theory*, 10, 1976.
- [KKLV10] Johannes Köbler, Sebastian Kuhnert, Bastian Laubner, and Oleg Verbitsky. Interval graphs: Canonical representation in logspace. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:43, 2010.

- [KM89] N. Korte and R.H. Möhring. An incremental linear-time algorithm for recognizing interval graphs. *SIAM J. Comput.*, pages 68–81, 1989.
- [KM02] P. S. Kumar and C. E. Veni Madhavan. Clique tree generalization and new subclasses of chordal graphs. *Discrete Applied Mathematics*, 117:109–131, 2002.
- [Kou77] Lawrence T. Kou. Polynomial complete consecutive information retrieval problems. *SIAM Journal on Computing*, 6(1):67–75, March 1977.
- [KR88] P.N. Klein and J.H. Reif. An efficient parallel algorithm for planarity. *Journal of Computer and System Science*, 18:190–246, 1988.
- [Lau09] Bastian Laubner. Capturing polynomial time on interval graphs. *CoRR*, abs/0911.3799, 2009. informal publication.
- [Lau10] Bastian Laubner. Capturing polynomial time on interval graphs. In *LICS*, pages 199–208. IEEE Computer Society, 2010.
- [LB63] C. G. Lekkerkerker and J. Ch. Boland. Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, 51:45–64, 1962/1963.
- [Lin92] Steven Lindell. A logspace algorithm for tree canonization (extended abstract). In *STOC*, pages 400–404. ACM, 1992.
- [McC04] Ross M. McConnell. A certifying algorithm for the consecutive-ones property. In *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 2004.
- [MM95] J. Meidanis and Erasmo G. Munuera. A simple linear time algorithm for binary phylogeny. In N. Ziviani, J. Piquer, B. Ribeiro, and R. Baeza-Yates, editors, *Proc. of the XV International Conference of the Chilean Computing Society*, pages 275–283, Nov 1995.
- [MM96] J. Meidanis and E. G. Munuera. A theory for the consecutive ones property. In *Proc. of the III South American Workshop on String Processing [MPT98]*, pages 194–202.
- [MPT98] Meidanis, Porto, and Telles. On the consecutive ones property. *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 88, 1998.
- [Nov89] Mark B. Novick. Generalized PQ-trees. Technical Report 1074, Dept. of Computer Science, Cornell University, Ithaca, NY 14853-7501, Dec 1989.
- [NS09] N. S. Narayanaswamy and R. Subashini. A new characterization of matrices with the consecutive ones property. *Discrete Applied Mathematics*, 157(18):3721–3727, 2009.
- [PPY94] Barry W. Peyton, Alex Pothén, and Xiaoping Yuan. A clique tree algorithm for partitioning a chordal graph into transitive subgraphs. Technical report, Old Dominion University, Norfolk, VA, USA, 1994.
- [Rei84] John H. Reif. Symmetric complementation. *JACM: Journal of the ACM*, 31(2):401–421, 1984.
- [Rei08] Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4), 2008.
- [Ren70] Peter L. Renz. Intersection representations of graphs by arcs. *Pacific J. Math.*, 34(2):501–510, 1970.
- [Rob51] W. S. Robinson. A method for chronologically ordering archaeological deposits. *American Antiquity*, 16(4):293–301, 1951.
- [Sch93] Alejandro A. Schaffer. A faster algorithm to recognize undirected path graphs. *Discrete Applied Mathematics*, 43:261–295, 1993.
- [SH99] W.K. Shih and W.L. Hsu. Note a new planarity test. *TCS: Theoretical Computer Science*, 223:179–191, 1999.
- [SW05] M. Suderman and S. Whitesides. Experiments with the fixed-parameter approach for two-layer planarization. *jgaa*, 9(1):149–163, 2005.

- [TM05] Guilherme P. Telles and João Meidanis. Building PQR trees in almost-linear time. *Electronic Notes in Discrete Mathematics*, 19:33–39, 2005.
- [Tuc72] Alan Tucker. A structure theorem for the consecutive 1’s property. *J. Comb. Theory Series B*, 12:153–162, 1972.
- [TZ04] Jinsong Tan and Louxin Zhang. Approximation algorithms for the consecutive ones submatrix problem on sparse matrices. In —, volume 3341 of *Lecture Notes in Computer Science*, pages 835–846, 2004.
- [TZ07] J. Tan and L. Zhang. The consecutive ones submatrix problem for sparse matrices. *Algorithmica*, 48(3):287–299, 2007.
- [Vel85] Marinus Veldhorst. Approximation of the consecutive ones matrix augmentation problem. *SIAM Journal on Computing*, 14(3):709–729, August 1985.
- [YC95] Yu and Chen. Efficient parallel algorithms for doubly convex-bipartite graphs. *TCS: Theoretical Computer Science*, 147:249–265, 1995.