CHAPTER FOUR
Chapter 4

On the optimization problems of some classes of sc perfect graphs

4.1 Introduction

The problem of finding a maximum clique, a minimum colouring, a maximum stable set and a minimum clique cover of a graph is known as optimization problems. These four problems are NP-hard in general [48]. In [59] Grotschel, Lovasz and Schrijver developed a polynomial time algorithm to solve these four optimization problems for perfect graphs. Although they gave polynomial time algorithm for perfect graphs, but the algorithm is extremely difficult; it uses the ellipsoid method from linear programming theory as a subroutine. No purely combinatorial algorithm is known for solving these problems on perfect graphs. This is the reason that the researchers are still studying optimization algorithms for perfect graphs and subclasses of perfect graphs too.

In this chapter we also study these optimization problems for sc perfect graphs and their subclasses namely, sc chordal graphs, sc weakly chordal graphs, sc quasi-chordal and sc brittle graphs. The section 4.2 deals with chromatic number of sc perfect graphs. In section 4.3, we discuss optimization
problem for sc weakly chordal graph followed by proposed algorithms. We extend our study for the class of sc perfectly orderable graphs in section 4.4.

Optimization problems can be divided mainly in two cases, weighted versions and unweighted versions. In this chapter we consider only unweighted problems and their algorithms.

4.2 On the chromatic number of sc perfect graphs

In general obtaining the exact value of the chromatic number of a graph is quite difficult. However researchers had obtained bounds for the chromatic number of graph and several classes of graphs [82] and [139]. In this section we also obtain bounds on the chromatic number of sc perfect graphs and using the result given by Sridharan and Balaji [131], we show that the upper bound is attained iff the graph is sc chordal.

The following result on the chromatic number of a graph and its complement was obtained by Nordhaus and Gaddum [98]

**Theorem 4.1 [98].** For any graph $G$ with $n$ vertices the following holds.

(i) $\left\lceil 2\sqrt{n} \right\rceil \leq \chi(G) + \chi(G) \leq n + 1$

(ii) $n \leq \chi(G) \chi(G) \leq \left\lceil \left( \frac{n+1}{2} \right)^2 \right\rceil$

The following result gives bounds for the chromatic number of a sc graph.

**Corollary 4.2.** Let $G$ be a sc graph with $n$ vertices. Then

(i) $\left\lceil 2\sqrt{p} \right\rceil \leq \chi(G) \leq 2p$ when $n = 4p$

(ii) $\left\lceil 2\sqrt{p+1} \right\rceil \leq \chi(G) \leq 2p + 1$ when $n = 4p + 1$. 

84
Proof. (i) Let $G$ be a sc graph with $4p$ vertices. We note that $\chi(G) = \chi(\overline{G})$ since $G$ is sc graph. So by theorem-4.1, $\chi(G)$ should satisfy the relation given by (4.1).

$$\left[2\sqrt{4p}\right] \leq 2\chi(G) \leq 4p + 1$$

and

$$4p \leq (\chi(G))^2 \leq \left[\frac{4p+1}{2}\right]$$

............... (4.1)

Then it follows that

$$\left[2\sqrt{p}\right] \leq \chi(G) \leq 2p$$

(ii) Let $G$ be a sc graph with $4p+1$ vertices. We note that $\chi(G) = \chi(\overline{G})$ since $G$ is sc graph. So by theorem-4.1, $\chi(G)$ should satisfy the relation given by (4.2).

$$\left[2\sqrt{4p+1}\right] \leq 2\chi(G) \leq 4p + 2$$

and

$$4p + 1 \leq (\chi(G))^2 \leq \left[\frac{4p+2}{2}\right]$$

............... (4.2)

Then it follows that

$$\left[\sqrt{4p+1}\right] \leq \chi(G) \leq 2p + 1.$$  \hspace{1cm} \Box

In [131] Sridharan and Balaji gave the following result.

**Theorem 4.3.** Let $G$ be a sc graph. Then $G$ is chordal iff $\omega(G) = 2p$ for $n = 4p$ and $\omega(G) = 2p+1$ for $n = 4p+1$.

The following Corollary is immediate from the above theorem.

**Corollary 4.4.** Let $G$ be a sc perfect graph. Then $G$ is chordal if and only if

(i) $\chi(G) = 2p$ when $n = 4p$

(ii) $\chi(G) = 2p+1$ when $n = 4p+1$.

Proof. The proof follows from the fact $\omega(G) = \chi(G)$ for perfect graphs. \hspace{1cm} \Box
Next result gives the bounds for the chromatic number of a sc perfect graph.

The upper bound given by this result is attained for sc chordal graphs.

**Theorem 4.5.** Let $G$ be a sc perfect graph. Then

(i) $\left\lfloor \sqrt{4p} \right\rfloor \leq \chi(G) \leq 2p$ when $n = 4p$

(ii) $\left\lfloor \sqrt{4p+1} \right\rfloor \leq \chi(G) \leq 2p + 1$ when $n = 4p + 1$.

Moreover, the upper bounds given by (i) and (ii) are attained if $G$ is chordal graph.

**Proof.** Follows from Theorem-4.2 and Corollary-4.4. □

From the above result it is clear that all other classes of sc perfect graphs such as sc quasi-chordal, sc brittle and sc weakly chordal graphs will have lower values of $\chi(G)$ from that of sc chordal graphs.

### 4.3 Contraction method and optimization algorithms

Contraction of two vertices into a new vertex is an important tool for designing optimization algorithms for many classes of perfect graphs such as Perfectly contactile graphs, Meyniel graph and various others [108]. In 1982, Fonlupt and Uhry [46] first observed and proved that if $G$ is a perfect graph and $\{x, y\}$ is an even pair in $G$, then the graph $G_{xy}$ is also perfect and has the same chromatic number as $G$. Since then, contraction of an even pairs have become an important tool for proving that certain classes of graphs are perfect and designing their optimization algorithms. However, the problem of deciding if a graph contains an even pair is NP-hard in general graphs [14].
In [73] Hoàng and Maffray proved that every weakly chordal graph contains an even pair. However contracting even pair in a weakly chordal graph does not always yield a new weakly chordal graph, for example adding an edge between the endpoints of \( P_{2k+1} \) (\( k \geq 3 \)) produces a \( C_{2k} \). Therefore to overcome this problem and develop a polynomial time optimization algorithm for weakly chordal graphs, Hayward et al. in [67] defined a two-pair, a special case of even pair. Since then optimizing algorithms for weakly chordal graphs mainly depend on the method of two-pair contractions.

In order to solve the optimization problems for sc weakly chordal graphs, we also follow the method of contraction of two-pair in the case of maximum clique and minimum coloring, while for the solution of maximum stable set and minimum clique cover we propose a different method namely co-pair edge contraction method.

### 4.3.1 Optimizing algorithms for sc weakly chordal graphs

Optimizing algorithms for weakly chordal graphs were first studied by Hayward et al. in [67]. They gave an \( O((n + m)n^3) \) time algorithm for maximum clique and minimum coloring, while algorithms for maximum stable set and minimum clique cover problem take \( O(n^4) \) time. The maximum stable set and minimum clique cover problem are essentially solved by running the former algorithm on the complement of the graph. Later Spinrad and Sritharan [127] also studied the same problems but for only weighted versions. Recently, Hayward et al. [68] proposed an \( O(nm) \) time algorithm, using the concept of
handles for weakly chordal graphs. However, they improved the time complexity of the algorithm but because of the involvement of handles it becomes much complicated and not easy to implement.

For finding the maximum clique and minimum coloring for sc weakly chordal graphs, we present an $O(mn^2)$ time algorithm, which is based on contraction of two-pair i.e. find a two-pair and modify the graph by adding an edge between them.

We now describe algorithm-4.1, which computes the maximum clique and minimum coloring for sc weakly chordal graphs. Algorithm-4.1, takes sc weakly chordal graph $G$ as input. Step-1 of algorithm-4.1 finds a two-pair $\{x, y\}$ of $G$, using algorithm-2.1. Step-2 Computes $N(x) \cup N(y) = W(\text{let})$. In step-3 we replace vertices $x$ and $y$ with a new vertex $z$. Step-4 adds edges $zw$ such that $w \in W$. In this way the graphs that come later in the sequence have strictly less vertices than the graphs those came earlier in the sequence. Step-5 repeats the process until no non-adjacent vertices are left in the graph. Eventually in the last graph all the vertices are mutually adjacent to each other, whose order is equal to the clique number and chromatic number of the input graph $G$. The algorithm 4.1 is as follows.
Algorithm 4.1: An algorithm for finding maximum clique and minimum coloring for sc weakly chordal graph.

**Input:** A sc weakly chordal graph $G$.  

**Output:** $\omega(G)$ and $\chi(G)$ for sc weakly chordal graph $G$.  

**Step-1:** Find a two-pair $\{x,y\}$ of $G$ (using algorithm 2.1)  

**Step-2:** Compute $N(x) \cup N(y) = W$ (let)  

**Step-3:** Replace $xy \rightarrow z$, where $z$ is new vertex.  

**Step-4:** Add edges $zw$ such that $w \in W$.  

**Step-5:** Repeat the process from step-1 to step-4, until no non-adjacent pair of vertices are left.  

**Step-6:** Order of last graph $= \omega(G) = \chi(G)$.  

**End.**  

**Complexity.** Since a two-pair can be found in $O(mn)$ time using algorithm-2.1. From step-2 to step-4 can be done in linear time. Now step-5 repeats the process $n$ times, so overall time complexity of algorithm-4.1 is $O(mn^2)$.  

The correctness of algorithm-4.1 rely on the following fact, which are due to Meyneil [93].  

**Theorem 4.6.** Let $x$ and $y$ be two vertices of a graph $G$ that are not joined by a chordless path with 3 edges. Then $\omega(G(xy \rightarrow z)) = \omega(G)$.  

**Theorem 4.7.** Algorithm-4.1 finds a maximum clique and a minimum coloring of $G$.  

89
Proof. By Theorem-4.6, it is clear that if \( \{x, y\} \) is an even pair in \( G \) then replacing \( x \) and \( y \) by a new vertex \( z \) and obtaining reduced graph has same size of maximum clique as \( G \). Also every two-pair is an even pair, so the above result is also true when we merge two non adjacent vertices which are two-pair into a new vertex. Hence the Theorem. \( \Box \)

We demonstrate algorithm-4.1 by giving an example of sc weakly chordal graph on 8 vertices as shown in figure-4.1

Let us input \( G \) in algorithm-4.1. Step-1 of algorithm-4.1 finds \( \{v_1, v_2\} \) as one of the two-pair in \( G \). Step-2 computes \( N(v_1) \cup N(v_2) = \{v_4, v_6, v_7, v_8\} \cup \{v_5, v_6, v_7, v_8\} = W \). Step-3 replaces vertex \( v_1 \) and \( v_2 \) by a new vertex \( z_1 \). Step-4 adds edges such as \( z_1v_4, z_1v_5, z_1v_6, z_1v_7, z_1v_8 \). Step-5 repeats the process untill their does not remain any non adjacent pair of vertices. Continuing this way the algorithm-4.1 finds the last graph in the sequence as of order 3. Therefore the maximum clique and minimum coloring of \( G \) is \( \omega(G) = \chi(G) = 3 \).
The overall procedure of finding maximum clique and minimum coloring of $G$ is shown in below figure-4.2.

![Diagram](image)

Figure-4.2

Let us now consider the other two optimization problems i.e. maximum stable set and minimum clique cover for sc weakly chordal graphs. Actually, Hayward et al. [67] presented only one algorithm, which solves the maximum clique and minimum coloring problems. For the solution of other two problems i.e. maximum stable set and minimum clique cover, they only mentioned that
the same algorithm can be run on the complement to get solution. No more
discussion was there on these two problems, so to understand it in more detail
we discuss here and propose an algorithm for the solution of maximum stable
set problem for sc weakly chordal graphs, while giving a different approach as
compare to [67]. So, to obtain this first we give the following Corollary, which
is immediate from the Theorem-2.3 and remembering the fact that sc weakly
chordal graphs have exactly \( n(n-1)/4 \) edges.

**Corollary 4.8.** Let \( G \) be a sc weakly chordal graph. Then each induced
subgraph of \( G \) contains a two-pair.

The following Lemma also ensures us for the existence of the
complement of two-pair i.e. co-pair in sc weakly chordal graph.

**Lemma 4.9.** Let \( G \) be a sc weakly chordal graph. Then each induced subgraph
of \( G \) contains a co-pair.

**Proof.** Since a co-pair is two-pair of the complement of a graph and we are
considering sc weakly chordal graph, which is isomorphic to its complement.
Hence the result. \( \square \)

The above result shows the importance of a co-pair in a sc weakly
chordal graph, so it is natural to try to formulate the analogue of the Edge
addition Theorem-2.15 using co-pair, which we state in Theorem-4.15.

Before we state and prove Theorem-4.15, we present some Lemmas that
will be useful in the proof of Theorem-4.15. Moreover in order to prove the
following Lemma's we use the process of recognition of an edge, which is co-
pair as follows: “Two adjacent vertices form a co-pair if and only if removing their common non-neighbors leaves the vertices in different components of the complement” [68].

**Lemma 4.10.** In a given graph $G$, an edge that belongs to a hole cannot be a co-pair.

**Proof.** Suppose that $G$ has a hole $v_1v_2\ldots v_k$, $k \geq 5$ as in figure-4.3(a). Let edge $v_1v_k$ be a co-pair. Neighbors of $v_1$ and $v_k$ are $v_2$ and $v_{k-1}$ respectively. The non-neighbors of $v_1$ are all the vertices except $v_2$ and $v_k$ and non-neighbors of $v_k$ are all the vertices except $v_1$ and $v_{k-1}$, the common non-neighbors are $v_3$ and $v_4$. Now removing the common non-neighbors $v_3$ and $v_4$ from $G$ and then considering the complement of the resultant graph which is shown in figure-4.3(b).

![Figure-4.3](image)

There is a path between $v_1$ and $v_k$, which passes through neighbors i.e. $v_{k-1}$ and $v_2$. From figure-4.3(b) it is clear that $v_1$ and $v_k$ lie in the same connected component in the complement, which contradicts the assumption that $v_1v_k$ is a co-pair edge. This argument holds for any edge of hole. Thus no edge of a hole can be a co-pair. Hence the result. □
The following Lemma shows the behavior of co-pair in antihole. In fact the proof is exactly on similar lines, but for completeness we include the proof.

**Lemma 4.11.** In a given graph $G$, an edge that belongs to an antihole cannot be a co-pair edge.

**Proof.** Let us suppose that $G$ has an antihole which is the complement of a hole on $v_1v_2...v_k$, $k > 5$ (for $k = 5$ antihole is isomorphic to hole) as shown in figure-4.4. Assume that edge $v_1v_4$ is co-pair edge, now the neighbors of $v_1$ are all vertices except $v_2$ and $v_k$ and the neighbors of $v_4$ are all vertices except $v_3$ and $v_{k-1}$. The non-neighbors of $v_1$ and $v_4$ are $v_2$, $v_k$ and $v_3$, $v_{k-1}$ respectively. Clearly there is no common non-neighbor of $v_1$ and $v_k$, which implies that their is always a path between $v_1$ and $v_4$ in the complement (one of them shown by doted lines in figure-4.4). Therefore $v_1$ and $v_4$ lie in the same connected component in the complement, which contradicts that the edge $v_1v_4$ is co-pair. Thus $v_1v_4$ is not co-pair edge. The similar argument holds for every other edge of antihole. □
**Lemma 4.12.** Let $G$ be a cycle of length 5 and having only one chord. Then this chord cannot be a co-pair.

**Proof.** Let $G$ be a cycle of length 5 with edges $v_1v_2$, $v_2v_3$, $v_3v_4$, $v_4v_5$ and $v_5v_1$. Let us suppose its only chord be $v_1v_3$, which is a co-pair. The non-neighbors of vertex $v_1$ and $v_3$ are $v_4$ and $v_5$ respectively. There is no common non-neighbors between $v_1$ and $v_3$ so vertex $v_1$ and $v_3$ always lie in the same connected component in the complement as shown by figure-4.5 (b).

This contradicts that $v_1v_3$ is a co-pair. Similarly other chords i.e. $v_1v_4$, $v_2v_4$, $v_2v_5$, $v_3v_5$ can also be shown that, they are not co-pairs. Hence the result. □

**Note.** Since an antihole of length 5 is isomorphic to itself, so along the same line of the proof of above Lemma we can prove that, when a graph, which contains a $C_k$ $(k = 5)$ with exactly one chord. Then this chord will never be a co-pair.

**Lemma 4.13.** Let $G$ be a graph, which contains a cycle with edges $v_1v_2$, $v_2v_3$, $v_3v_4$, $v_4v_5$, $v_kv_1$ for $k > 5$. Let there be only one chord in this cycle. Then this chord cannot be a co-pair.

**Proof.** Suppose that $G$ has a cycle as shown in figure-4.6(a).
Let us suppose that one of the chord \( v_1v_3 \) is a co-pair. The non-neighbors of \( v_1 \) and \( v_3 \) are \( v_4, v_{k-1} \) and \( v_{k-1}, v_k \) respectively. So their common non-neighbor is \( v_{k-1} \). After removing this vertex \( v_{k-1} \) from \( G \) and considering the complement of the resultant graph the vertices \( v_1 \) and \( v_3 \) lie in the same connected component, which contradicts the assumption that \( v_1v_3 \) is a co-pair. This argument holds for every other chord of the same type in the cycle.

Let one of the other type of chord \( v_1v_4 \) be a co-pair. The non-neighbors of \( v_1 \) and \( v_4 \) are \( v_3, v_{k-1} \) and \( v_2, v_k \) respectively. Clearly there is no common non-neighbor between them, which implies that in the complement vertices \( v_1 \) and \( v_4 \) always lie in the same connected component. This contradicts that \( v_1v_4 \) is a co-pair. By a similar argument it can be shown that every other chord of same type cannot be a co-pair. Therefore in either case the added chord is not a co-pair. Hence the result. \( \Box \)

**Lemma 4.14.** Let \( G \) be any graph, which contains \( \overline{C_k} \) with vertices \( v_1v_2...v_k \), \( k \geq 6 \). Suppose there is exactly one chord between any two non-adjacent pair of vertices of \( \overline{C_k} \). Then this chord cannot be a co-pair.
Proof. Let $G$ contains a $\overline{C}_k$, $k \geq 6$. we have to show that the chord which is added is not a co-pair. Suppose it is added between vertices $v_2$ and $v_3$, as shown in figure-4.7(a). It is clear from figure-4.7(a) that each vertex of $G$ except $v_2$ and $v_3$ have non-neighbors at most 2, while vertex $v_2$ and $v_3$ have non-neighbors one each.

![Figure-4.7](image)

Now as we know that the complement of $\overline{C}_k$ is cycle of regular degree 2 (as shown in figure-4.7(b)), after deleting a chord from $\overline{C}_k$, its complement becomes chordless path $P_k$ (as shown in figure-4.7(c)). Clearly in any chordless path $P_k$ ($k > 3$) no two end vertices have common neighbors, which implies that the vertices $v_2$ and $v_3$ in $G$ have no common non-neighbors. Therefore $v_1$ and $v_3$ always lie in same connected component in the complement, which contradicts that $v_1v_3$ is a co-pair. This argument holds for any chord (exactly one in $G$) for any $\overline{C}_k$ ($k \geq 6$). Hence the result $\Box$

In the above results we consider, hole and antihole with exactly one chord because this is the only case, when it is possible that after deleting a chord, a hole or antihole may be generated. On the other hand if hole and
anthohole have more than one chord then deletion of a chord at a time certainly
does not create a hole or antihole. We are now in a position to state and prove
Theorem-4.15.

**Theorem 4.15.** Let $e_{xy}$ be a co-pair of a graph $G$. Let $G_1$ be the graph obtained
from $G$ by deleting $e_{xy}$. Then $G$ is weakly chordal if and only if $G_1$ is weakly
chordal.

**Proof.** For the necessary condition we have to show that the deletion of $e_{xy}$
does not destroy the property of weakly chordal graph. Deletion of $e_{xy}$ from $G$
will make $G_1$ not weakly chordal only when $G_1$ becomes either a hole or
antihole. But from Lemma 4.12 and Lemma 4.13, we know that if deleted edge
makes any graph hole then it will not be a co-pair edge i.e. we are deleting co-
pair edge which cannot be a chord of a hole or antihole. Also from Lemma 4.12
and Lemma 4.14, it follows that the deleted edge which makes any graph
antihole will not be a co-pair. Therefore deletion of $e_{xy}$ from $G$ will always
produce $G_1$, which is weakly chordal. Now to prove the sufficient condition let
$G$ be not a weakly chordal, then either $G$ is hole or antihole or contains them.
Suppose $G$ is a hole then from Lemma 4.10, no $G_1$ will ever be produced.
Similarly if $G$ is antihole then from Lemma 4.11, again no $G_1$ will be produced.
So if $G_1$ is weakly chordal then $G$ is also. Hence the theorem. $\square$

Using Theorem-4.15, we suggest the following procedure for finding
maximum stable set for sc weakly chordal graph via edge deletion method. The
process is as follows: find a co-pair edge $e_{xy}$ and replace the two adjacent
vertices $x$ and $y$ by a new vertex $w$ with neighborhood $N(w) = N(x) \cap N(y)$ and then repeat the process. However an equivalent way of viewing the operation is as follows: mark $x$ to $w$ in the new graph, delete every edge incident on $y$, mark $y$ as deleted and delete every edge $xu$ such that $u \in N(x) - N(y)$.

The algorithm is as follows.

**Algorithm-4.2: An algorithm for finding largest sable set and minimum clique cover for sc weakly chordal graph.**

**Input:** A sc weakly chordal graph $G$.

**Output:** $\alpha(G)$ and $\theta(G)$ for sc weakly chordal graph $G$.

**Step-1:** Find a co-pair $e_{xy}$ of $G$.

and let $V$ = set of vertices of $G$.

**Step-2:** Compute $N(x) - N(y) = U$ (let)

**Step-3:** Replace vertex $x$ with $w$ and delete edges $xu$ such that $u \in U$.

**Step-4:** Mark vertex $y$ as deleted

**Step-5:** Update $V$ by replacing $x$ and $y$ by $w$.

**Step-6:** Repeat the whole process until no co-pair found.

**Step-7:** Return $|V| = \alpha(G) = \theta(G)$

End.

**Complexity.** An edge is a co-pair and it can be tested in $O(n+m)$ time in step-1. Step-2 to step-5 can be done in linear time. Step-6 repeats the whole process $n$ time. Therefore the overall time complexity of algorithm-4.2 is $O((n+m)n)$ time.
The correctness of algorithm-4.2 follows from the fact $\alpha(G) = \omega(\overline{G})$ and from Theorem-4.6 and Theorem-4.7.

**Theorem 4.16.** Algorithm-4.2 finds maximum stable set and minimum clique cover.

We illustrate algorithm-4.2 by giving an example of sc weakly chordal on 9 vertices as shown in figure-4.8.

Let us input sc weakly chordal graph $G$ as shown in figure-4.8 in algorithm-4.2. Step-1 starts with vertex set $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$ and finds edge $(v_1, v_8)$ as one of the co-pair edge of $G$. Step-2 computes $N(v_1) - N(v_8) = \{v_3, v_4, v_5, v_7, v_9\}$. Step-3 replaces vertex $v_1$ with $w_1$ and deletes edges $v_1v_3$ and $v_1v_8$. Step-4 deletes vertex $v_8$ and updates the vertex set $V$ as $\{w_1, v_2, v_3, v_4, v_5, v_6, v_7, v_9\}$. Step-6 repeats the process, until no co-pair edge remains in $G$. Continuing in this way the algorithm-4.2 finally produces the maximum stable set of $G$ as $|V| = \alpha(G) = 3$, where $V = \{w_1, w_5, w_6\}$. The overall procedure can be seen in figure-4.9.
Chapter 4

In the optimization problems of some classes of gc perfect graphs

Figure-4.9

(a)

V = \{v_5, v_2, v_3, v_4, v_6, v_7, v_8\}

\(v_4 \leftrightarrow v_5\)

(b)

V = \{w_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}

\(v_7 \leftrightarrow v_4\)

(c)

V = \{w_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}

\(v_5 \leftrightarrow v_4\)

(e)

V = \{w_2, w_3, v_2, w_4, v_6\}

\(v_5 \leftrightarrow w_4\)

(f)

V = \{w_2, v_3, w_4, w_5\}

\(v_5 \leftrightarrow w_4\)

(g)

V = \{w_2, w_3, w_6\}

\(v_5 \leftrightarrow w_4\)

\(v_5 \leftrightarrow w_4\)

\(v_5 \leftrightarrow w_4\)

\(v_5 \leftrightarrow w_4\)

\(v_5 \leftrightarrow w_4\)

101
4.4 Optimization algorithms for the class of sc perfectly orderable graphs

As a first step to understand this section, let us define, one of the natural way to color a graph i.e. greedy coloring algorithm [82], which is as follows: The first step is to impose an order < on the vertices \( v_i \) of a graph and then to progress through the vertices in the assigned order. If \( v_i < v_j \) then \( v_i \) appears before \( v_j \) in the assigned ordering. At each vertex \( v_i \) we look at all the neighbors \( v_j \) of \( v_i \) such that \( v_j < v_i \) and assign \( v_i \) the lowest possible color not assigned to one of these neighbors. This method is called greedy coloring algorithm. It is important to realize that the greedy algorithm does not guarantee that the coloring is an optimal coloring. So, the next logical question is: can we find a class of graphs for which the greedy coloring algorithm always produces an optimal coloring? In order to answer this question, Chvátal [26] proposed the concept of perfect order (<) and perfectly orderable graphs and gave the following result.

**Theorem 4.17** [26]. Given a perfect order of a graph, the greedy coloring algorithm using the order computes a minimal coloring.

Where an order (<) is perfect if for each induced subgraph, the chordless path on four vertices i.e. \( P_4 \) with vertices \( a, b, c, d \) and edges \( ab, bc, cd \) such that \( a < b, d < c \).

However, it is much difficult to find a perfect order of a graph. In [94] Middendorf and Pfeiffer proved that to decide whether a graph is perfectly
orderable or have a perfect order is NP-complete. This has motivated researchers to study subclasses of perfectly orderable graphs in the hope that such an effort will lead to a better understanding of the combinatorial structure of the perfectly orderable graphs.

For this purpose, we study the optimization problems for sc perfectly orderable graph classes such as sc quasi-chordal graphs and sc brittle graphs and propose a polynomial time algorithm, which solves the problem of minimum coloring in sc quasi-chordal and sc brittle graphs.

4.4.1 Optimization algorithm for sc quasi-chordal graphs and sc brittle graphs

In order to solve the optimization problems for sc quasi-chordal graphs and sc brittle graphs, we need to find first their perfect order. Often, elimination order of any subclass of perfectly orderable graphs becomes perfect order for that graph. For example, simplicial vertex elimination order of chordal graph is known as simplicial order and is also perfect order. Unfortunately, it is neither true for sc quasi-chordal graphs nor for sc brittle graphs. For example consider the following sc quasi-chordal graph $G$ which is also sc brittle as shown in figure-4.10. One of its elimination order is $(v_3 < v_4 < v_7 < v_8 < v_1 < v_5 < v_2 < v_6)$. Now consider one of the induced $P_4$ of $G$ which is $[v_3, v_6, v_2, v_5]$ in which $v_3 < v_6, v_2 < v_5$, this clearly implies that this is an obstruction (an obstruction is an ordered graph is the $P_4 [a, b, c, d]$ with $a < b$ and $d < c$ [108]) in $G$. Therefore this elimination order cannot be perfect order.
However, after some modification on elimination order $\sigma$ of either sc quasi-chordal graph or sc brittle graph we can obtain perfect order $\sigma'$ from $\sigma$. For obtaining this we use following process.

Let $G$ be a sc quasi-chordal graph, then the eliminated vertex of $G$ is either a co-simplicial vertex or simplicial vertex. Thus $\sigma$ i.e. elimination ordering of $G$ will contain co-simplicial and simplicial vertices. Now $\sigma'$ is easily obtained from $\sigma$ by placing first all co-simplicial vertices followed by all simplicial vertices. Similarly in the case of sc brittle graphs, the eliminated vertex is either no-mid or no-end, we also know that every simplicial vertex is no-mid and every co-simplicial vertex is no-end. So we can obtain $\sigma'$ from $\sigma$ in sc brittle graphs by first placing all no-mid vertices and then placing all no-end vertices.

We are now in a position to discuss the following algorithm-4.3, which computes minimum coloring of sc quasi-chordal graphs and sc brittle graphs. The algorithm works as follows; after inputting the graph $G$ for which we want to compute minimum coloring. Step-1, finds its elimination order, which can be
done by either algorithm-3.2 (for sc brittle graphs) or algorithm-3.4 (for sc quasi-chordal graphs). In step-2 it obtains $\sigma'$ from $\sigma$ by doing necessary calculation. Now apply greedy coloring algorithm on $\sigma'$ in step-3. Step-4 returns the output $\chi(G)$ of the input graph $G$. The algorithm is as follows.

**Algorithm-4.3: An Algorithm for finding minimum coloring for sc quasi-chordal graph and sc brittle graph.**

**Input:** A graph $G$ either sc quasi-chordal or sc brittle graph.

**Output:** $\chi(G)$ of $G$.

**Step-1:** Find $\sigma$ of $G$. (using algorithm-3.2 or 3.4)

**Step-2:** Obtain $\sigma'$ from $\sigma$.

**Step-3:** Apply greedy algorithm on $\sigma'$.

**Step-4:** Return number of minimum color i.e. $\chi(G)$.

**End.**

**Complexity.** Since any elimination order $\sigma$ for sc quasi-chordal graphs and sc brittle graphs can be obtained in $O(n^3m)$ time. Now this $\sigma$ can be converted into perfect order $\sigma'$ in $O(n^3)$ time. Step-3 can be done in $O(n^3m)$ time. Hence overall time complexity of algorithm-4.3 is $O(n^3m)$ time

**Theorem 4.18.** Algorithm-4.3 finds minimum coloring of $G$.

**Proof.** Since we are applying greedy coloring algorithm on $\sigma'$, which is perfect order. From Theorem-4.17, we know that applying greedy coloring algorithm on any perfect order will always produce minimum number of color.

Hence the Theorem. \qed
We illustrate algorithm-4.3 by using the same example as shown in figure-4.10.

Let us input this graph $G$ in algorithm-4.3. Step-1 finds one of its elimination order using algorithm-3.4 as $\sigma = (v_3 < v_4 < v_7 < v_8 < v_1 < v_5 < v_2 < v_6)$. Now step-2 converts this elimination order $\sigma$ into perfect order $\sigma'$ as $\sigma' = (v_7 < v_8 < v_1 < v_4 < v_1 < v_5 < v_2 < v_6)$. Since $\sigma'$ is perfect order so applying greedy coloring algorithm on this order will necessarily produce optimal coloring for $G$, this is done in step-3, assigning colors 1, 2, 3... as follows: vertex $v_7$ is the first in $\sigma'$ so obviously it takes color 1. The next vertex in $\sigma'$ is $v_8$, which is neighbor of $v_7$ so it cannot take color 1 so give it color 2. Next vertex $v_1$ gets color 1 because it is non-neighbor of $v_7$. Similarly vertices $v_5, v_2$ and $v_6$ get color 1, 2 and 3 respectively, as shown in figure-4.11(b). Since all the vertices are colored, thus step-4 provides $\chi(G) = 3$ i.e. minimum coloring of $G$. Also it can be seen from the input graph $G$ that it contains a triangle, hence optimal color cannot be less than 3.
4.4.2 Algorithm for finding the solution of optimization problems for sc quasi-chordal graphs and sc brittle graphs without given perfect order

To design a polynomial algorithm, which solves the optimization problem for perfectly orderable graphs without given perfect order is an open problem. Till today, there is no polynomial time algorithm for perfectly orderable graphs, this again shows the importance of the study of subclasses of perfectly orderable graphs. In this subsection we show that how to get solution of optimization problem for sc quasi-chordal graphs and sc brittle graphs without given perfect order.

Since sc quasi-chordal graphs and sc brittle graphs both are the subclasses of sc weakly chordal graphs so any optimizing algorithm for sc weakly chordal graphs will also work on these subclasses. If we input these two subclasses in either algorithm-4.1 or algorithm-4.2, then it will produce their maximum clique, minimum coloring, maximum stable set and minimum clique cover respectively. The important fact is that both the algorithms do not need any perfect order of input graph in input. Hence we have polynomial algorithms for sc quasi-chordal graphs and sc brittle graphs, which can solve their optimization problems without given a perfect order. So in view of above discussion we have following result.

**Theorem 4.19.** Let $G$ be a sc quasi-chordal graphs or sc brittle graphs, then maximum clique, minimum coloring, maximum stable set and minimum clique cover of $G$ can be solved in polynomial time without given a perfect order.