# Exact algorithms for maximum weighted independent set on sparse graphs (Extended Abstract)

Sen Huang, Mingyu Xiao, and Xiaoyu Chen

University of Electronic Science and Technology of China, China
{huangsen47,myxiao,x312035}@gmail.com

**Abstract.** The maximum independent set problem is one of the most important problems in graph algorithms and has been extensively studied in the line of research on the worst-case analysis of exact algorithms for NP-hard problems. In the weighted version, each vertex in the graph is associated with a weight and we are going to find an independent set of maximum total vertex weight. In this paper, we design several reduction rules and a fast exact algorithm for the maximum weighted independent set problem, and use the measure-and-conquer technique to analyze the running time bound of the algorithm. Our algorithm works on general weighted graphs and it has a good running time bound on sparse graphs. If the graph has an average degree at most 3, our algorithm runs in $O^*(1.1443^n)$ time and polynomial space, improving previous running time bounds for the problem in cubic graphs using polynomial space.

**Keywords:** Maximum Weighted Independent Set · Exact Algorithms· Measure-and-Conquer· Graph Algorithms· Reduction Rules.

## 1 Introduction

The Maximum Independent Set problem on unweighted graphs belongs to the first batch of 21 NP-hard problems proved by Karp [14]. In the line of research on the worst-case analysis of exact algorithms for NP-hard problems, Maximum Independent Set, as one of the most fundamental problems, is used to test the efficiency of new techniques of exact algorithms. There is a long list of contributions to exact algorithms for Maximum Independent Set in unweighted graphs [22, 13, 20, 10, 15, 2]. Now it can be solved in $O^*(1.1996^n)$ time and polynomial space [26]. If the maximum degree of the graph is 3, the running time bound can be improved to $O^*(1.0836^n)$ [25].

In this paper, we will consider the weighted version of Maximum Independent Set, called Maximum Weighted Independent Set, where each vertex in the graph has a nonnegative weight and we are asked to find an independent set with maximum total vertex weight. It has many applications in various real-world problems. For example, the dynamic map labeling problem [1, 17] can be naturally encoded as Maximum Weighted Independent Set. Some experimental algorithms, such as the algorithms in [16, 24] have been developed to solve instances from real world and known benchmarks. These algorithms run fast even on large scale sparse instances but lack running time analysis. For running time bounds, most known results were obtained via two counting problems: Counting Maximum Weighted Independent Set and Counting Max 2-SAT. Most of these counting algorithms can also list out all independent sets and then we can find a maximum one by increasing only a polynomial factor. Dahllöf et al. [6] presented an $O^*(1.3247^n)$-time algorithm for Counting Maximum Weighted Independent Set. Later, the running time bound was improved to $O^*(1.2431^n)$ by Fomin et al. [8]. Counting Maximum Weighted Independent Set can also be reduced to Counting Max 2-SAT, preserving the exponential part of the running time. For Counting Max 2-SAT, the running time bound was improved from $O^*(1.2561^n)$ [7] to $O^*(1.2461^n)$ [12] and then to $O^*(1.2377^n)$ [23]. Wahlström [23] also showed that the running time bound could be further improved to $O^*(1.1499^n)$ and $O^*(1.2117^n)$ if the maximum degree of the variables or the vertices in the graph is bounded by 3 and 4, respectively. Most of the above algorithms use only polynomial space. If exponential space is allowed, dynamic programming algorithms based on tree decompositions, by using the treewidth bound on degree-3 graphs in [9], may achieve a better running time bound $O^*(1.1225^n)$.

In this paper, we will focus on exact algorithms specifying for Maximum Weighted Independent Set. We develop structural properties and design reduction rules for the problem, and then design a fast exact algorithm based on them. By using the measure-and-conquer technique, we can prove that the algorithm runs in $O^*(1.1443^{(0.624x-0.872)n})$ time and polynomial space, where $x$ is the average degree of the graph. For some sparse graphs, our result beats the known bounds. For example, the running time bound of our algorithm in graphs with the average degree at most three is $O^*(1.1443^n)$, which improves the previously known bound of $O^*(1.1499^n)$ using polynomial space [23].

Due to the limited space, the proofs of lemmas marked with (*) were omitted, which can be found in the full version in Appendix.

## 2 Preliminaries

Let $G = (V, E, w)$ denote an undirected vertex-weighted graph with $|V| = n$ vertices and $|E| = m$ edges, where each vertex $v \in V$ is associated with a positive weight $w(v)$. Although our graphs are undirected, we may use an arc to denote the relation of the weights of the two endpoints of an edge. An *arc* $\overrightarrow{uv}$ from vertex $u$ to vertex $v$ means that there is an edge between $u$ and $v$ and it holds that $w(u) \geq w(v)$.

For a vertex subset $V' \subseteq V$, we let $w(V') = \sum_{v \in V'} w(v)$, and $N(V')$ denote the set of open neighborhood of $V'$. We also let $d(V') = |N(V')|$. For a vertex subset $S \subseteq V$, we use $G[S]$ to denote the subgraph of $G$ induced by $S$ and use $G - S$ to denote $G[V \setminus S]$. For a graph $G'$, we use $\mathcal{C}(G')$ to denote the set of connected components of $G'$. A *chain* is an induced path such that the degree of each vertex except the two endpoints of the path is exactly 2. One vertex is a *chain-neighbor* of another vertex if there are connected by a chain. For a vertex-weighted graph, a *maximum weighted independent set* is an independent set $S$ such that $w(S)$ is maximized among all independent sets in the graph. We use $S(G)$ to denote a maximum weighted independent set in graph $G$ and $\alpha(G)$ to denote the total vertex weight of $S(G)$. Our problem is defined below.

---

Maximum Weighted Independent Set (MWIS)
**Input**: An undirected vertex-weighted graph $G = (V, E, w)$.
**Output**: the weight of a maximum weighted independent set in $G$., i.e., $\alpha(G)$.

---

### 2.1 Measure-and-conquer

Our algorithm is a branch-and-search algorithm. We will use a measure to evaluate the time complexity. For a branching operation, if the measure decreases by at least $a_i$ in the $i$-th substance, then we say the *branching vector* of the operation is $[a_1, a_2, \ldots, a_l]$. The largest root of the function $f(x) = 1 - \sum_{i=1}^{l} x^{-a_i}$ is called the *branching factor* of the recurrence.

The measure-and-conquer technique, introduced in [10], is a powerful tool to analyze branch-and-search algorithms. The main idea of the measure-and-conquer technique is to use a non-traditional measure to evaluate the size of the search tree generated by the branch-and-search algorithm. In this paper, we will use the measure-and-conquer technique to analyze our algorithm. Our measure $p$ is a combination of several parameters defined below. This measure may catch more structural properties of the problem and then we can analyze the running time by using amortization. Let $n_i$ denote the number of vertices of degree $i$ in the graph. We associate a cost $\delta_i \geq 0$ for each degree-$i$ vertex in the graph. Our measure is set as follows:

$$p := \sum_{i=0}^{n} n_i \delta_i. \tag{1}$$

The cost $\delta_i$ in this paper is given by

$$\delta_i = \begin{cases} 0 & \text{if } i \leq 1 \\ 0.376 & \text{if } i = 2 \\ 1 & \text{if } i = 3 \\ 1 + 0.624(i-3) & \text{if } i \geq 4. \end{cases} \tag{2}$$

We also define $\delta_i^{<-k>} := \delta_i - \delta_{i-k}$ for each integer $k \geq 0$. In our analysis, we may use the following inequalities and equalities to simplify some arguments: $\delta_i^{<-1>} = \delta_3^{<-1>}$ for $i \geq 4$; $\delta_3 \geq 2.5\delta_2; 3\delta_2 \geq \delta_3$.

With the above setting, we know that when $p \leq 0$, the instance contains only degree-0 and degree-1 vertices and can be solved directly. We will design an algorithm with running time bound $O^*(c^p)$ for some constant $c$. If the initial graph has degree at most 3, then we have that $p \leq n$ and then the running time bound of the algorithm is $O^*(c^n)$. In general, if we have $p \leq f(n)$ for some function $f$ on $n$, then we can get a running time bound of $O^*(c^{f(n)})$. We have the following lemma for the relation between $p$ and $n$.

**Lemma 1.** *(*) For a graph of $n$ vertices, if the average degree of the graph is at most $x$, then the measure $p$ of the graph is at most $(0.624x - 0.872)n$.*

## 3 Reduction Rules

We first introduce reduction rules that will be applied to reduce the instance directly by eliminating some local structures of the graph. Some reduction rules may include a set $S$ of vertices in the solution set directly. We use $M_c$ to store the weight of the vertices that have been included in the solution set. When a set $S$ of vertices is included in the solution set, we will remove $N[S]$ from the graph and update $M_c$ by adding $w(S)$.

### 3.1 General Reductions for Some Special Structures

We use several reduction rules based on unconfined vertices, twins, vertices with a clique neighborhood, and heavy vertices. Some of these reduction rules were introduced in [16] and [24].

**Unconfined Vertices.** A vertex $v$ in $G$ is called *removable* if $\alpha(G) = \alpha(G - v)$, i.e., there is a maximum weighted independent set in $G$ that does not contain $v$. We can say that a vertex $v$ is removable if a contradiction is obtained from the assumption that every maximum weighted independent set in $G$ contains $v$. A sufficient condition for a vertex to be removable in unweighted graphs has been studied in [25]. We extend this concept to weighted graphs.

For an independent set $S$ of $G$, a vertex $u \in N(S)$ is called a *child* of $S$ if $w(u) \geq w(S \cap N(u))$. A child $u$ is called an *extending child* if it holds that $|N(u) \setminus N[S]| = 1$, and the only vertex $v \in N(u) \setminus N[S]$ is called a *satellite* of $S$.

**Lemma 2.** *(*) Let $S$ be an independent set that is contained in any maximum weighted independent set in $G$. Then every maximum weighted independent set contains at least one vertex in $N(u) \setminus N[S]$ for each child $u$ of $S$.*

Lemma 2 provides a sufficient condition for a vertex set contained in any maximum weighted independent set. Next, we introduce a method based on Lemma 2 to find some possible removable vertices.

Let $v$ be an arbitrary vertex in the graph. After starting with $S := \{v\}$, we repeat (1) until (2) or (3) holds:

(1) If $S$ has some extending child in $N(S)$, then let $S'$ be the set of satellites. Update $S$ by letting $S := S \cup S'$.
(2) If $S$ is not an independent set or there is a child $u$ such that $N(u) \setminus N[S] = \emptyset$, then halt and conclude that $v$ is *unconfined.*
(3) If $|N(u) \setminus N[S]| \geq 2$ for all children $u \in N(S)$, then halt and return $S_v = S$.

Obviously, the procedure can be executed in polynomial time for any starting set $S$ of a vertex. If the procedure halts in (2), we say vertex $v$ *unconfined*. If the procedure halts in (3), then we say that the set $S_v$ returned in (3) *confines* vertex $v$ and vertex $v$ is also called *confined*. Note that the set $S_v$ confining $v$ is uniquely determined by the procedure with starting set $S := \{v\}$. It is easy to observe the following lemma.

**Lemma 3.** *(*) Any unconfined vertex is removable.*

**Reduction Rule 1 (R1)** *If a vertex $v$ is unconfined, remove $v$ from $G$.*

**Twins.** A set $A = \{u, v\}$ of two non-adjacent vertices is called a *twin* if they have the same neighbor set, i.e., $N(u) = N(v)$.

**Reduction Rule 2 (R2)** *[16] If there is a twin $A = \{u, v\}$, delete $v$ and update the weight of $u$ by letting $w(u) := w(u) + w(v)$.*

**Clique Neighborhood**. A vertex $v$ has a *clique neighborhood* if the graph $G[N(v)]$ induced by the open neighbor set of $v$ is a clique, which was introduced as isolated vertices in [16].

**Reduction Rule 3 (R3)** *[16] If there is a vertex $v$ having a clique neighborhood and $w(v) < w(u)$ holds for all $u \in N(v)$, then remove $v$ from the graph, update the weight $w(u) := w(u) - w(v)$ for all $u \in N_G(v)$, and add $w(v)$ to $M_c$.*

**Heavy Vertices**. A vertex $v$ is called a *heavy vertex* if its weight is not less the weight of the maximum weighted independent set in subgraph induced by the open neighborhood of it, i.e., $w(v) \geq \alpha(G[N(v)])$.

**Reduction Rule 4 (R4)** *If there is a heavy vertex $v$ of degree at most 5, then delete $N[v]$ from the graph and add $w(v)$ to $M_c$.*

It is an effective rule that has been used in some experimental algorithms [16, 24]. In this paper, we will only check heavy vertices of degree bounded by 5 and then it can be done in polynomial time. Note that degree-0 vertices will be reduced as heavy vertices in this step.

## 3.2 Reductions Based on Degree-2 Vertices

For unweighted graphs, we have good reduction rules to deal with all degree-2 vertices (see the reduction rule in [4]). However, for weighted graphs, it becomes much more complicated. The following R5 is generalized from the concept of folding degree-2 vertices in unweighted graphs in [4], which has been also used in some experimental algorithms [16, 24]. We also consider more reduction rules for degree-2 vertices in some complicated structures. Full proofs of the correctness of the rules can also be found in Appendix.

**Reduction Rule 5 (R5)** *If there is a degree-2 vertex $v$ with two neighbors $\{u_1, u_2\}$ such that $w(u_1) + w(u_2) > w(v) \geq max\{w(u_1), w(u_2)\}$, then delete $\{v, u_1, u_2\}$ from the graph $G$, introduce a new vertex $v'$ adjacent to $N_G(\{v, u_1, u_2\})$ with weight $w(v') := w(u_1) + w(u_2) - w(v)$, and add $w(v)$ to $M_c$.*

**Reduction Rule 6 (R6)** *If there is a path $v_1v_2v_3v_4$ such that $d_G(v_2) = d_G(v_3) = 2$ and $w(v_1) \geq w(v_2) \geq w(v_3) \geq w(v_4)$, then remove $v_2$ and $v_3$ from the graph, add an edge $v_1v_4$ if it does not exist, update the weight of $v_1$ by letting $w(v_1) := w(v_1) + w(v_3) - w(v_2)$, and add $w(v_2)$ to $M_c$.*

**Reduction Rule 7 (R7)** *If there is a 4-cycle $v_1v_2v_3v_4$ such that $d_G(v_2) = d_G(v_3) = 2$ and $w(v_1) \geq w(v_2) \geq w(v_3)$, then remove $v_2$ and $v_3$, update the weight of $v_1$ by letting $w(v_1) := w(v_1) + w(v_3) - w(v_2)$, and add $w(v_2)$ to $M_c$.*

**Reduction Rule 8 (R8)** *If there is a 4-path $v_1v_2v_3v_4v_5$ such that $d_G(v_2) = d_G(v_3) = d_G(v_4) = 2$ and $w(v_1) \geq w(v_2) \geq w(v_3) \leq w(v_4) \leq w(v_5)$, then remove $v_2$ and $v_4$, add edges $v_1v_3$ and $v_3v_5$, update the weight of $v_1$ by letting $w(v_1) := w(v_1) + w(v_3) - w(v_2)$ and the weight of $v_5$ by letting $w(v_5) := w(v_5) + w(v_3) - w(v_4)$, and add $w(v_2) + w(v_4) - w(v_3)$ to $M_c$.*

**Reduction Rule 9 (R9)** *For a 5-cycle $v_1v_2v_3v_4v_5$ such that $d_G(v_2) = d_G(v_3) = d_G(v_5) = 2$, $\min\{d(v_1), d(v_4)\} \geq 3$, and $w(v_1) \geq w(v_2) \geq w(v_3) \leq w(v_4)$,*

*(1) if $w(v_3) > w(v_5)$, then remove $v_5$, update the weight of $v_i$ by letting $w(v_i) := w(v_i) - w(v_5)$ for $i = 1, 2, 3, 4$, and add $2w(v_5)$ to $M_c$.*
*(2) if $w(v_3) \leq w(v_5)$, then remove $v_2$ and $v_3$, update the weight of $v_1$ by letting $w(v_1) := w(v_1) - w(v_2)$, the weight of $v_4$ by letting $w(v_4) := w(v_4) - w(v_3)$ and the weight of $v_5$ by letting $w(v_5) := w(v_5) - w(v_3)$, and add $w(v_2) + w(v_3)$ to $M_c$.*

**Reduction Rule 10 (R10)** *For a 6-cycle $v_1v_2v_3v_4v_5v_6$ such that $d_G(v_2) = d_G(v_3) = d_G(v_5) = d_G(v_6) = 2$, $w(v_1) \geq \max\{w(v_2), w(v_6)\}$, $w(v_4) \geq \max\{w(v_3), w(v_5)\}$, and $w(v_6) \geq w(v_5)$,*

*(1) if $w(v_2) \geq w(v_3)$, then remove $v_5$ and $v_6$, and update the weight of $v_2$ by letting $w(v_2) := w(v_2) + w(v_6)$ and the weight of $v_3$ by letting $w(v_3) := w(v_3) + w(v_5)$;*
*(2) if $w(v_2) < w(v_3)$, then remove $v_6$, add edge $v_1v_5$, and update the weight of $v_2$ by letting $w(v_2) := w(v_2) + w(v_6)$, the weight of $v_3$ by letting $w(v_3) := w(v_3) + w(v_5)$, and the weight of $v_5$ by letting $w(v_5) := w(v_6) + w(v_3) - \max\{w(v_2) + w(v_6), w(v_3) + w(v_5)\}$.*

## 3.3 Reductions Based on Small Cuts

We also have some reduction rules to deal with vertex-cuts of size one or two, which can even be used to design a polynomial-time divide-and-conquer algorithm. However, a graph may not always have vertex-cuts of small size.

**Reduction Rule 11 (R11)** *For a vertex-cut $\{u\}$ with a connected component $G^*$ in $G-u$ such that $2\delta_3-\delta_2 \leq \sum_{v\in G^*}\delta_{d_G(v)} \leq 10$,*

*(1) if $w(u)+\alpha(G^*-N[u]) \leq \alpha(G^*)$, then remove $G^*$ and $\{u\}$ from $G$ and add $\alpha(G^*)$ to $M_c$;*
*(2) if $w(u)+\alpha(G^*-N[u]) > \alpha(G^*)$, then remove $G^*$ from $G$, update the weight of $u$ by letting $w(u) := w(u)+\alpha(G^*-N[u])-\alpha(G^*)$, and add $\alpha(G^*)$ to $M_c$.*

A proof of the correctness and an illustration can be found in Appendix.

**Lemma 4.** *(*) Let $\{u,u'\}$ be a vertex-cut of size two in $G$ and $G^*$ be a connected component in $G-\{u,u'\}$, where we assume w.l.o.g. that $\alpha(G^*-N[u]) \geq \alpha(G^*-N[u'])$. We construct a new graph $G'$ from $G$ as follows: remove $G^*$; add three new vertices $\{v_1,v_2,v_3\}$ with weight $w(v_1)=\alpha(G^*-N[u'])-\alpha(G^*-N[\{u,u'\}])$, $w(v_2)=\alpha(G^*-N[u])-\alpha(G^*-N[\{u,u'\}])$ and $w(v_3)=\alpha(G^*)-\alpha(G^*-N[u])$, and add five new edges $uv_1$, $v_1v_2$, $v_2u'$, $uv_3$ and $u'v_3$. It holds that*

$$\alpha(G)=\alpha(G')+\alpha(G^*-N[\{u,u'\}]).$$

**Reduction Rule 12 (R12)** *For a vertex-cut $\{u,u'\}$ of size two with a connected component $G^*$ in $G-\{u,u'\}$ such that $2\delta_3+\delta_2 \leq \sum_{v\in G^*}\delta_{d_G(v)} \leq 10$, we construct the graph $G'$ in Lemma 4, replace $G$ with $G'$, and add $\alpha(G^*-N[\{u,u'\}])$ to $M_c$.*

## 3.4 Analyzing Reduction Rules

It is easy to see that each application of our reduction rules can be executed in polynomial time. We also show that

**Lemma 5.** *The measure $p$ will not increase after applying any reduction rule.*

**Definition 1.** *An instance is* reduced*, if none reduction rule can be applied.*

**Lemma 6.** *(*) In a reduced instance, any two degree-2 vertices in different chains have at most one common chain-neighbor of degree at least 3, and each cycle contains at least three vertices of degree $\geq 3$.*

**Lemma 7.** *(*) For a triangle $C$ in a reduced instance, each vertex in $C$ is a vertex of degree $\geq 3$ and it has a chain-neighbor of degree at least 3 not in $C$.*

## 4 Branching Rules

Next, we introduce our branching rules, which will only be applied to reduced instances. After applying a branching rule, the algorithm will apply reduction rules as more as possible on each sub instance.

### 4.1 Two Branching rules

The first branching rule is to branch on a vertex $v$ by considering two cases: (i) there is a maximum weighted independent set in $G$ which does not contain $v$; (ii) every maximum weighted independent set in $G$ contains $v$. For the former case, we simply delete $v$ from the graph. For the latter case, by Lemma 2 we know that we can include the set $S_v$ confining $v$ in the independent set. So we delete $N[S_v]$ from the graph.

**Branching Rule 1 (Branching on a vertex)** *Branch on a vertex $v$ to generate two sub instances by either deleting $v$ from the graph or deleting $N[S_v]$ from the graph and adding $w(S_v)$ to $M_c$.*

Since each independent set contains at most two vertices in each 4-cycle, we have the second rule.

**Branching Rule 2 (Branching on a 4-cycle)** *Branch on a 4-cycle $v_1v_2v_3v_4$ to generate two sub instances by deleting either $\{v_1, v_3\}$ or $\{v_2, v_4\}$ from $G$.*

### 4.2 The analysis and some properties

The hardest part is to analyze how much we can decrease the measure in each sub-branch of a branching operation. Usually, we need to deeply analyze the local graph structure and use case-analysis. Here we try to summarize some common properties. The following notations will be frequently used in the whole paper.

Let $S$ be a vertex subset in a reduced graph $G$. We use $G_{-S}$ to denote the graph after deleting $S$ from $G$ and iteratively applying R1 to R4 until none of them can be applied. We use $R_S$ to denote the set of deleted vertices during applying R1 to R4 on $G - S$. Then $G_{-S} = G - (S \cup R_S)$. We also use $e_S$ to denote the number of edges between $S \cup R_S$ and $V \setminus (S \cup R_S)$ in $G$. We have the following lemmas for some bounds on $p(G) - p(G_{-S})$. Note that $G_{-S}$ may not be a reduced graph because of reduction rules from R5 to R12 and we may further apply reduction rules to further decrease the measure $p$.

**Lemma 8.** *(*) It holds that*

$$p(G) - p(G_{-S}) = \sum_{u \in S \cup R_S} \delta_{d_G(u)} + e_S \delta_3^{<-1>}. \tag{3}$$

In some cases, we can not use the bound in (3) directly, since we may not know the vertex set $R_S$. So we also consider some special cases and relaxed bounds.

**Lemma 9.** *(*) Let $S = \{v\}$ be a set of a vertex of degree $\geq 3$. We have that*

$$p(G) - p(G_{-S}) \geq \delta_{d(v)} + \sum_{u \in N(v)} \delta_{d(u)}^{<-1>} + q_2 \delta_3^{<-1>},$$

*where $q_2$ is the number of degree-2 vertices in $N(v)$.*

**Lemma 10.** *(*) If $S \cup R_S$ contains $N[v]$ for some vertex $v$ of degree $\geq 3$, then we have that*

$$p(G) - p(G_{-S}) \geq \sum_{u \in N[v]} \delta_{d(u)} + q_2 \delta_3^{<-1>},$$

*where $q_2$ is the number of degree-2 vertices in $N(v)$.*

Recall that we use $\mathcal{C}(G')$ to denote the set of connected components of the graph $G'$. We can easily obverse the following lemma, which will be used to prove several bounds on $p(G) - p(G_{-S})$.

**Lemma 11.** *Let $S$ be a vertex subset. Let $S'$ be a subset of $S \cup R_S$ and $R' = S \cup R_S \setminus S'$. The number of edges between $S \cup R_S$ and $V \setminus (S \cup R_S)$ is $e_S$, and the number of edges between $S'$ and $V \setminus S'$ is $k$. For any component $H \in \mathcal{C}(G[R'])$, the number of edges between $S'$ and $H$ is $l_H$ and the number of edges between $H$ and $N(S \cup R_S)$ is $r_H$. We have that*

$$k - e_S = \sum_{H \in \mathcal{C}(G[R'])} (l_H - r_H).$$

*Furthermore, for any component $H \in \mathcal{C}(G[R'])$ containing only degree-2 vertices, it holds that*

$$l_H - r_H = 0 \quad or \quad 2.$$

**Lemma 12.** *(*) For any subset $S' \subseteq S \cup R_S$ with $k$ edges between $S'$ and $V \setminus S'$, it holds that*

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S \delta_3^{<-1>} + \begin{cases} 0, & k - e_S \leq 0 \\ \delta_3, & k - e_S = 1 \\ \delta_2, & k - e_S = 2 \\ \delta_3, & k - e_S = 3 \\ 2\delta_2, & k - e_S > 3. \end{cases}$$

**Lemma 13.** *(*) Assume that a reduced graph $G$ has a maximum degree 3 and has no 3 or 4-cycles. For any subset $S' \subseteq S \cup R_S$ with $k$ edges between $S'$ and $V \setminus S'$, if the diameter of the induced graph $G[S']$ is 2, then it holds that either $p(G) - p(G_{-S}) > 10$ or*

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + 3\delta_3^{<-1>} + \begin{cases} 0, & k \leq 3 \\ \delta_3^{<-1>}, & k = 4 \\ 2\delta_2, & k = 5 \\ \delta_2 + \delta_3, & k = 6. \end{cases}$$

**Lemma 14.** *(*) Assume that a reduced graph $G$ has a maximum degree 3, and each cycle $C$ in it contains at least five vertices, where at least four vertices are degree-3 vertices. For any subset $S' \subseteq S \cup R_S$ with $k$ edges between $S'$ and $V \setminus S'$, if each path $P$ in the induced graph $G[S']$ contains either at most three vertices or at most two degree-3 vertices, then it holds either $p(G) - p(G_{-S}) > 10$ or*

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + \begin{cases} k\delta_3^{<-1>}, & k \leq 5 \\ \delta_3 + 2\delta_2 + 3\delta_3^{<-1>}, & k = 6. \end{cases}$$

## 5 The Algorithm

Now we describe the whole algorithm. The algorithm will first apply reduction rules and then branch on vertices of degree $\geq 5$ if any. Third, the algorithm will deal with 4-cycles and degree-4 vertices. Last is to deal with degree-3 vertices, which is the most complicated part of the algorithm. When the algorithm executes one step, we assume that all previous steps can not be applied.

**Step 1 (Applying Reductions)** *If the instance is not reduced, iteratively apply reduction rules in order, i.e., when one reduction rule is applied, no reduction rule with a smaller index can be applied on the graph.*

**Step 2 (Solving Small Components)** *If there is a connected component $G^*$ of $G$ such that $p(G^*) \leq 10$, solve the component $G^*$ directly and return $\alpha(G - G^*) + \alpha(G^*)$.*

**Step 3 (Branching on Vertices of Degree $\geq 5$)** *If there is a vertex $v$ with degree $d(v) \geq 5$, then branch on $v$ with Branching Rule 1 by either excluding $v$ from the independent set or including $S_v$ in the independent set.*

**Lemma 15.** *(*) Step 3 followed by applications of reduction rules creates a branching vector covered by* $[5.368, 7.248]$.

**Step 4 (Branching on 4-Cycles with Chords)** *If there is a 4-cycle $C = v_1v_2v_3v_4$ with a chord $v_1v_3 \in E$, then branch on the 4-cycle with Branching Rule 2 by excluding either $\{v_1, v_3\}$ or $\{v_2, v_4\}$ from the independent set.*

**Lemma 16.** *(*) Step 4 followed by applications of reduction rules creates a branching vector covered by one of* $[3\delta_4+\delta_3^{<-1>}, 4\delta_4+2\delta_3^{<-1>}] = [5.496, 7.744]$ *and* $[4\delta_4, 2\delta_4 + 2\delta_3 + 2\delta_2] = [6.496, 6]$.

**Step 5 (Branching on Degree-4 Vertices)** *If there is a degree-4 vertex $v$, then branch on it with Branching Rule 1 by either excluding $v$ from the independent set or including $S_v$ in the independent set.*

**Lemma 17.** *(*) Step 5 followed by applications of reduction rules creates a branching vector covered by one of* $[5.624, 5.624]$, $[5.248, 6]$, $[4.872, 6.624]$, $[4.496, 7.248]$, *and* $[4.12, 7.872]$.

**Step 6 (Branching on Other 4-Cycles)** *If there is a 4-cycle $C = v_1v_2v_3v_4$, then branch on the 4-cycle with Branching Rule 2 by excluding either $\{v_1, v_3\}$ or $\{v_2, v_4\}$ from the independent set.*

**Lemma 18.** *(*) Step 6 followed by applications of reduction rules creates a branching vector covered by* $[6\delta_3 - 2\delta_2, 6\delta_3 - 2\delta_2] = [5.248, 5.248]$.

**Step 7 (Branching on Triangles)** *If there is a triangle $C = v_1v_2v_3$, where we assume without loss of generality that $w(v_1) \geq \max\{w(v_2), w(v_3)\}$ and $v_1$ is chain-adjacent to a degree-3 vertex $u \neq v_2, v_3$, then branch on $u$ with Branching Rule 1.*

**Lemma 19.** *(*) Step 7 followed by applications of reduction rules creates a branching vector covered by one of* $[6\delta_3 - 3\delta_2, 7\delta_3 + \delta_2] = [4.872, 7.376]$ *and* $[6\delta_3 - 2\delta_2, 5\delta_3 + 2\delta_2] = [5.248, 5.752]$.

**Step 8 (Branching on Cycles Containing Three Degree-3 Vertices)** *If there is a cycle $C$ containing exactly three degree-3 vertices $\{v_1, v_2, v_3\}$, where we assume without loss of generality that $v_1$ is chain-adjacent to a degree-3 vertex $u \neq v_2, v_3$, then branch on $u$ with Branching Rule 1.*

**Lemma 20.** *(*) Step 8 followed by applications of reduction rules can create a branching vector covered by one of* $[6\delta_3 - 4\delta_2, 8\delta_3 - 2\delta_2] = [4.496, 7.248]$, $[6\delta_3 - 3\delta_2, 6\delta_3 - \delta_2] = [4.872, 5.624]$*, and* $[6\delta_3 - 2\delta_2, 6\delta_3 - 2\delta_2] = [5.248, 5.248]$.

**Step 9 (Branching on Degree-3 Vertices with Two Degree-2 Neighbors)** *If there is degree-3 vertex $u$ having two degree-2 neighbors and one degree-3 neighbor $v$, then branch on $v$ with Branching Rule 1.*

**Lemma 21.** *(*) Step 9 followed by applications of reduction rules creates a branching vector covered by one of* $[4\delta_3 - \delta_2, 8\delta_3 - \delta_2] = [3.624, 7.624]$, $[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496]$*, and* $[4\delta_3 + \delta_2, 6\delta_3] = [4.376, 6]$.

**Step 10 (Branching on Degree-3 Vertices of a Mixed Case)** *If a degree-3 vertex $u$ without degree-3 neighbors is chain-adjacent to a degree-3 vertex $v$ with exactly two degree-3 neighbors, then branch on $v$ with Branching Rule 1.*

**Lemma 22.** *(*) Step 10 followed by applications of reduction rules creates a branching vector covered by* $[4\delta_3, 8\delta_3 - 2\delta_2] = [4, 7.248]$.

**Step 11 (Branching on Degree-3 Vertices With At Least Two Degree-3 Neighbors)** *If there is a connected component $H$ containing a degree-3 vertex with at least two degree-3 neighbors, we let $u$ be the vertex of the maximum weight in $H$ and let $v$ be a degree-3 neighbor of $u$, and branch on $v$ with Branching Rule 1.*

**Lemma 23.** *(*) Step 11 followed by applications of reduction rules creates a branching vector covered by one of* $[4\delta_3 - \delta_2, 8\delta_3 - \delta_2] = [3.624, 7.624]$*, and* $[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496]$.

**Step 12 (Branching on Other Degree-3 Vertices)** *Pick up an arbitrary degree-3 vertex $v$ and branch on it with Branching Rule 1.*

**Lemma 24.** *(*) Step 12 followed by applications of reduction rules creates a branching vector covered by* $[4\delta_3 + 6\delta_2, 4\delta_3 + 6\delta_2] = [6.256, 6.256]$.

It is easy to see that above steps cover all the cases. Among all the branching vectors, the bottleneck ones are $[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496]$ in Lemma 21, $[4\delta_3 + \delta_2, 6\delta_3] = [4.376, 6]$ in Lemma 21, and $[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496]$ in Lemma 23. All of them have a branching factor of 1.14427. So we get that

**Theorem 1.** Maximum Weighted Independent Set *can be solved in* $O^*(1.1443^p)$ *time and polynomial space.*

By Lemma 1 and Theorem 1, we get that

**Corollary 1.** Maximum Weight Independent Set *in graphs with average degree* $x$ *can be solved in* $O^*(1.1443^{(0.624x-0.872)n})$ *time and polynomial space.*

Let $x = 3$ in Lemma 1, we get that $p \leq n$ and the following result.

**Theorem 2.** Maximum Weight Independent Set *in graphs with the average degree at most 3 can be solved in* $O^*(1.1443^n)$ *time and polynomial space.*

## 6 Conclusion

In this paper, we design an exact algorithm for Maximum Weighted Independent Set. With the help of the measure-and-conquer technique, we analyze a nontrivial running time bound for the algorithm, which has a good performance on sparse graphs. For graphs with an average degree at most three, the running time bound is $O^*(1.1443^n)$, improving previous running time bounds for the problem in cubic graphs using polynomial space. Although the improvement is incremental, such improvements on classic problems have became harder and harder. Any further improvement may need new observations on the structural properties or new techniques to design and analyze the algorithms. Foe unweighted Maximum Independent Set on degree-3 graphs, the running time bound was improved for several times [5, 3, 19, 2, 25]. Each improvement is small, but each improvement reveals new properties and new analysis. Our algorithm is analyzed by the measure-and-conquer technique. The framework of the analysis may also provide a way to analyze other related problems.

## References

1. Been, K., Daiches, E., Yap, C.K.: Dynamic map labeling. IEEE Transactions on Visualization and Computer Graphics **12**(5), 773–780 (2006)
2. Bourgeois, N., Escoffier, B., Paschos, V.T., van Rooij, J.M.M.: Fast algorithms for max independent set. Algorithmica **62**(1), 382–415 (2012)
3. Bourgeois, N., Escoffier, B., Paschos, V.T.: An $\mathrm{O}^*(1.0977^{\mathrm{n}})$ exact algorithm for max independent set in sparse graphs. In: Parameterized and Exact Computation. pp. 55–65 (2008)
4. Chen, J., Kanj, I.A., Jia, W.: Vertex cover: Further observations and further improvements. Journal of Algorithms **41**(2), 280–301 (2001)
5. Chen, J., Kanj, I.A., Xia, G.: Labeled search trees and amortized analysis: Improved upper bounds for np-hard problems. Algorithmica **43**(4), 245–273 (2005)
6. Dahllöf, V., Jonsson, P.: An algorithm for counting maximum weighted independent sets and its applications. In: Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 292–298 (2002)
7. Dahllöf, V., Jonsson, P., Wahlström, M.: Counting models for 2sat and 3sat formulae. Theoretical Computer Science **332**(1-3), 265–291 (2005)

8. Fomin, F.V., Gaspers, S., Saurabh, S.: Branching and treewidth based exact algorithms. In: Proceedings of Seventeenth International Symposium on Algorithms and Computation. Lecture Notes in Computer Science, vol. 4288, pp. 16–25 (2006)
9. Fomin, F.V., Gaspers, S., Saurabh, S., Stepanov, A.A.: On two techniques of combining branching and treewidth. Algorithmica **54**(2), 181–207 (2009)
10. Fomin, F.V., Grandoni, F., Kratsch, D.: A measure & conquer approach for the analysis of exact algorithms. Journal of the ACM **56**(5), 25:1–25:32 (2009)
11. Fomin, F.V., Kratsch, D.: Exact Exponential Algorithms. Texts in Theoretical Computer Science. An EATCS Series, Springer (2010)
12. Fürer, M., Kasiviswanathan, S.P.: Algorithms for counting 2-satsolutions and colorings with applications. In: Proceedings of Third International Algorithmic on Aspects in Information and Management. pp. 47–57 (2007)
13. Jian, T.: An $O(2^{0.304n})$ algorithm for solving maximum independent set problem. IEEE Transactions on Computers **35**(9), 847–851 (1986)
14. Karp, R.M.: Reducibility among combinatorial problems. In: Proceedings of a Symposium on the Complexity of Computer Computations. pp. 85–103. The IBM Research Symposia Series (1972)
15. Kneis, J., Langer, A., Rossmanith, P.: A Fine-grained Analysis of a Simple Independent Set Algorithm. In: Proceedings of IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science. Leibniz International Proceedings in Informatics (LIPIcs), vol. 4, pp. 287–298 (2009)
16. Lamm, S., Schulz, C., Strash, D., Williger, R., Zhang, H.: Exactly solving the maximum weight independent set problem on large real-world graphs. In: Proceedings of Algorithm Engineering and Experiments. pp. 144–158 (2019)
17. Liao, C.S., Liang, C.W., Poon, S.H.: Approximation algorithms on consistent dynamic map labeling. Theoretical Computer Science **640**, 84–93 (2016)
18. Niedermeier, R., Rossmanith, P.: On efficient fixed-parameter algorithms for weighted vertex cover. Journal of Algorithms **47**(2), 63–77 (2003)
19. Razgon, I.: Faster computation of maximum independent set and parameterized vertex cover for graphs with maximum degree 3. Journal of Discrete Algorithms **7**(2), 191–212 (2009)
20. Robson, J.M.: Algorithms for maximum independent sets. Journal of Algorithms **7**(3), 425 – 440 (1986)
21. Shachnai, H., Zehavi, M.: A multivariate framework for weighted FPT algorithms. Journal of Computer and System Science **89**, 157–189 (2017)
22. Tarjan, R.E., Trojanowski, A.E.: Finding a maximum independent set. SIAM Journal on Computing **6**(3), 537–546 (1977)
23. Wahlström, M.: A tighter bound for counting max-weight solutions to 2sat instances. In: Proceedings of Third International Workshop on Parameterized and Exact Computation. Lecture Notes in Computer Science, vol. 5018, pp. 202–213 (2008)
24. Xiao, M., Huang, S., Zhou, Y., Ding, B.: Efficient reductions and a fast algorithm of maximum weighted independent set. In: WWW'21: The Web Conference 2021. pp. 3930–3940 (2021)
25. Xiao, M., Nagamochi, H.: Confining sets and avoiding bottleneck cases: A simple maximum independent set algorithm in degree-3 graphs. Theoretical Computer Science **469**, 92–104 (2013)
26. Xiao, M., Nagamochi, H.: Exact algorithms for maximum independent set. Information and Computation **255**, 126–146 (2017)

# APPENDIX
# Exact algorithms for maximum weighted independent set on sparse graphs (full version)

Sen Huang, Mingyu Xiao, and Xiaoyu Chen

University of Electronic Science and Technology of China, China
{huangsen47,myxiao,x312035}@gmail.com

**Abstract.** The maximum independent set problem is one of the most important problems in graph algorithms and has been extensively studied in the line of research on the worst-case analysis of exact algorithms for NP-hard problems. In the weighted version, each vertex in the graph is associated with a weight and we are going to find an independent set of maximum total vertex weight. In this paper, we design several reduction rules and a fast exact algorithm for the maximum weighted independent set problem, and use the measure-and-conquer technique to analyze the running time bound of the algorithm. Our algorithm works on general weighted graphs and it has a good running time bound on sparse graphs. If the graph has an average degree at most 3, our algorithm runs in $O^*(1.1443^n)$ time and polynomial space, improving previous running time bounds for the problem in cubic graphs using polynomial space.

**Keywords:** Maximum Weighted Independent Set · Exact Algorithms· Measure-and-Conquer· Graph Algorithms· Reduction Rules.

## 1 Introduction

The Maximum Independent Set problem on unweighted graphs belongs to the first batch of 21 NP-hard problems proved by Karp [17]. This problem is so important in graph algorithms that it is often introduced as the first problem in textbooks and lecture notes of exact algorithms. In the line of research on the worst-case analysis of exact algorithms for NP-hard problems, Maximum Independent Set, as one of the most fundamental problems, is used to test the efficiency of new techniques of exact algorithms.

There is a long list of contributions to exact algorithms for Maximum Independent Set in unweighted graphs. Tarjan and Trojanowski [26] designed the first nontrivial algorithm in 1977, which runs in $O^*(2^{\frac{n}{3}})$ time and polynomial space. Later, Jian [16] obtained an $O^*(1.2346^n)$-time algorithm. Robson [24] gave an $O^*(1.2278^n)$-time polynomial-space algorithm and an $O^*(1.2109^n)$-time exponential-space algorithm. By using the measure-and-conquer technique, Fomin et al. [12] obtained a simple $O^*(1.2210^n)$-time polynomial-space algorithm. Based on this method, Kneis et al. [18] and Bourgeois et al. [4] improved the running time bound to $O^*(1.2132^n)$ and $O^*(1.2114^n)$, respectively. Currently, the best algorithm is the $O^*(1.1996^n)$-time polynomial-space algorithm introduced in [34], which even breaks the bound 1.2 in the base of the exponential part of the running time.

For Maximum Independent Set in degree-bounded graphs, there is also a considerable amount of contributions in the literature [3, 5, 7, 32]. For Maximum Independent Set in degree-3 graphs, let us quote the $O^*(1.1254^n)$-time algorithm by [7], the $O^*(1.1034^n)$-time algorithm by [29], the $O^*(1.0977^n)$-time algorithm by [5], the $O^*(1.0892^n)$-time algorithm by [23], the $O^*(1.0885^n)$-time algorithm by [28], the $O^*(1.0854^n)$-time algorithm by [4], the $O^*(1.0836^n)$-time algorithm by [32], and the $O^*(1.0821^n)$-time algorithm by [15]. Furthermore, Maximum Independent Set in degree-4 graphs can be solved in $O^*(1.1376^n)$ time [31], and Maximum Independent Set in degree-5 graphs can be solved in $O^*(1.1736^n)$ time [33].

In this paper, we will consider the weighted version of Maximum Independent Set, called Maximum Weighted Independent Set, where each vertex in the graph has a nonnegative weight and we are asked to find an independent set with maximum total vertex weight. Most known results for Maximum Weighted Independent Set were obtained via two counting problems: Counting Maximum Weighted Independent Set and Counting Max 2-SAT. Most of these counting algorithms can also list out all independent sets and then we can find a maximum one by increasing only a polynomial factor. Dahllöf et al. [8] presented an $O^*(1.3247^n)$-time algorithm for Counting Maximum Weighted Independent Set. Later, the running time bound was improved to $O^*(1.2431^n)$ by Fomin et al. [10]. Counting Maximum Weighted Independent Set can also be reduced to Counting Max 2-SAT, preserving the exponential part of the running time. In the reduction, we construct a clause $\overline{u} \vee \overline{v}$ for each edge $uv$ in the graph (See [9] for more details). For Counting Max 2-SAT, the running time bound was improved from $O^*(1.2561^n)$ [9] to $O^*(1.2461^n)$ [14] and then to $O^*(1.2377^n)$ [27]. Wahlström [27] also showed that the running time bound could be further improved to $O^*(1.1499^n)$ and $O^*(1.2117^n)$ if the maximum degree of the variables or the vertices in the graph is bounded by 3 and 4, respectively. Most of the above algorithms use only polynomial space. If exponential space is allowed, dynamic programming algorithms based on tree decompositions can achieve a better running time bound. On graphs of treewidth at most $t$, Maximum Weighted Independent Set can be solved in $O^*(2^t)$ time and space by a standard dynamic programming algorithm. It is known that the treewidth of graphs with maximum degree 3 and 4 is roughly bounded by $n/6$ and $n/3$, respectively [11]. Thus, Maximum Weighted Independent Set in graphs of maximum degree 3 (resp., 4) can be solved in $O^*(1.1225^n)$ time (resp., $O^*(1.2600^n)$ time) and exponential space. We also note that the due problem Minimum Weighted Vertex Cover has been extensively studied in parameterized complexity. By taking the weight value $W$ of the vertex cover as the parameter, some parameterized algorithms have been proposed in [22] and [10]. By taking the size $s$ of the minimum weighted vertex cover as the parameter, there are also some known parameterized algorithms [25].

Maximum Weighted Independent Set is an important problem with many applications in various real-world problems. For example, the dynamic map labeling problem [2, 20] can be naturally encoded as Maximum Weighted Independent Set. Some experimental algorithms, such as the algorithms in [19, 30] have been developed to solve instances from real world and known benchmarks. These algorithms run fast even on large scale sparse instances but lack running time analysis. On the other hand, the fast algorithms for Counting Maximum Weighted Independent Set and Counting Max 2-SAT and the DP algorithm based tree decompositions do not rely on the structural properties of Maximum Weighted Independent Set.

In this paper, we will focus on exact algorithms specifying for Maximum Weighted Independent Set. We develop structural properties and design reduction rules for the problem, and then design a fast exact algorithm based on them. By using the measure-and-conquer technique, we can prove that the algorithm runs in $O^*(1.1443^{(0.624x-0.872)n})$ time and polynomial space, where $x$ is the average degree of the graph. For some sparse graphs, our result beats the known bounds. For example, the running time bound of our algorithm in graphs with the average degree at most three is $O^*(1.1443^n)$, which improves the previously known bound of $O^*(1.1499^n)$ using polynomial space [27].

## 2 Preliminaries

Let $G = (V, E, w)$ denote an undirected vertex-weighted graph with $|V| = n$ vertices and $|E| = m$ edges, where each vertex $v \in V$ is associated with a positive weight $w_G(v)$, where the subscript $G$ may be omitted if it is clear from the context. Although our graphs are undirected, we may use an arc to denote the relation of the weights of the two endpoints of an edge. An *arc* $\overrightarrow{uv}$ from vertex $u$ to vertex $v$ means that there is an edge between $u$ and $v$ and it holds that $w(u) \geq w(v)$.

For a vertex subset $V' \subseteq V$, we let $w(V') = \sum_{v \in V'} w(v)$. For a vertex subset $V' \subseteq V$, we let $N_G(V')$ denote the set of *open neighborhood* of $V'$, i.e., $N_G(V') = \{v | v \text{ is adjacent to some vertex in } V' \ \& \ v \notin V'\}$. We also let $d_G(V') = |N_G(V')|$ and $N_G[V'] = N_G(V') \cup V'$, where $N_G[V']$ is the set of closed neighborhood of $V'$. When the graph $G$ is clear from the context, we may omit the subscript and simply write $N_G(V')$, $d_G(V')$ and $N_G[V']$ as $N(V')$, $d(V')$ and $N[V']$, respectively. When $V' = \{v\}$ is a singleton, we may simply write it as $v$. For a vertex subset $S \subseteq V$, we use $G[S]$ to denote the subgraph of $G$ induced by $S$ and use $G - S$ to denote $G[V \setminus S]$. For a graph $G'$, we use $\mathcal{C}(G')$ to denote the set of connected components of $G'$. A *chain* is an induced path such that the degree of each vertex except the two endpoints of the path is exactly 2. One vertex is a *chain-neighbor* of another vertex if there are connected by a chain.

A path (or cycle) is said to be a $k$-path (or $k$-cycle) if there are $k$ edges. A set $S$ of vertices in graph $G$ is called an *independent set* if for any pair of vertices in $S$ there is no edge between them. For a vertex-weighted graph, a *maximum weighted independent set* is an independent set $S$ such that $w(S)$ is maximized among all independent sets in the graph. We use $S(G)$ to denote a maximum weighted independent set in graph $G$ and $\alpha(G)$ to denote the total vertex weight of $S(G)$. The Maximum Weighted Independent Set problem is defined below.

---

Maximum Weighted Independent Set (MWIS)
**Input**: An undirected vertex-weighted graph $G = (V, E, w)$.
**Output**: the weight of a maximum weighted independent set in $G$., i.e., $\alpha(G)$.

---

## 2.1 Branch-and-search and Measure-and-conquer

**Branch-and-search paradigm.** Our branch-and-search algorithm contains several reduction rules and branching rules. Each reduction rule will reduce the instance without exponentially increasing the running time. We will first apply reduction rules to reduce this instance and then apply branching rules to search for a solution when the instance can not be further reduced.

The exponential part of the running time depends on the size of the "search tree" in the algorithm, which is generated by the branching operations. To evaluate the size of the search tree, we should use a measure. The measure can be the number of vertices or edges of the graph, the size of the solution, and so on. Usually, when the parameter becomes zero or less than zero, the instance can be solved in polynomial time directly. Let parameter $p$ be the measure adopted in the algorithm. We use $T(p)$ to denote the maximum number of leaves in the search tree generated in the algorithm for any instance with the measure being at most $p$. Assume that at a branching operation, the algorithm branches on the current instance into $l$ branches. If in the $i$-th branch the measure decreases by at least $a_i$, i.e., the $i$-th substance has the parameter at most $p_i = p - a_i$, then we obtain a recurrence relation

$$T(p) \leq T(p - a_1) + T(p - a_2) + \ldots + T(p - a_l).$$

The recurrence relation can be represented by a *branching vector* $[a_1, a_2, \ldots, a_l]$. The largest root of the function $f(x) = 1 - \sum_{i=1}^{l} x^{-a_i}$ is called the *branching factor* of the recurrence. Let $\gamma$ be the maximum branching factor among all branching factors in the algorithm. The size of the search tree that represents the branching process of the algorithm applied to an instance with parameter $p$ is given by $O^*(\gamma^p)$. More details about the analysis and how to solve recurrences can be found in the monograph [13].

For two branching vectors $\mathbf{a} = [a_1, a_2, \ldots, a_l]$ and $\mathbf{b} = [b_1, b_2, \ldots, b_l]$, if $a_i \geq b_i$ holds for all $i = 1, 2 \ldots, l$, then the branching factor of $\mathbf{a}$ is not greater than this of $\mathbf{b}$. For this case, we say $\mathbf{b}$ *dominates* $\mathbf{a}$. This property will be used in many places to simplify some arguments in the paper.

**Measure-and-conquer technique.** The Measure-and-conquer technique, introduced in [12], is a powerful tool to analyze branch-and-search algorithms. The main idea of the measure-and-conquer technique is to use a non-traditional measure to evaluate the size of the search tree generated by the branch-and-search algorithm. In this paper, we will use the measure-and-conquer technique to analyze our algorithm. Our measure $p$ is a combination of several parameters defined below. This measure may catch more structural properties of the problem and then we can analyze the running time by using amortization. Let $n_i$ denote the number of vertices of degree $i$ in the graph. We associate a cost $\delta_i \geq 0$ for each degree-$i$ vertex in the graph. Our measure is set as follows:

$$p := \sum_{i=0}^{n} n_i \delta_i. \tag{1}$$

The cost $\delta_i$ in this paper is given by

$$\delta_i = \begin{cases} 0 & \text{if } i \leq 1 \\ 0.376 & \text{if } i = 2 \\ 1 & \text{if } i = 3 \\ 1 + 0.624(i-3) & \text{if } i \geq 4. \end{cases} \tag{2}$$

We also define

$$\delta_i^{<-k>} := \delta_i - \delta_{i-k},$$

for each integer $k \geq 0$. In our analysis, we may use the following inequalities and equalities to simplify some arguments:

$$\delta_i^{<-1>} = \delta_3^{<-1>} \quad \text{for each } \ i \geq 4; \tag{3}$$

$$\delta_3 \geq 2.5\delta_2; \tag{4}$$

$$3\delta_2 \geq \delta_3. \tag{5}$$

With the above setting, we know that when $p \leq 0$, the instance contains only degree-0 and degree-1 vertices and can be solved directly. We will design an algorithm with running time bound $O^*(c^p)$ for some constant $c$. If the initial graph has degree at most 3, then we have that $p \leq n$ and then the running time bound of the algorithm is $O^*(c^n)$. In general, if we have $p \leq f(n)$ for some function $f$ on $n$, then we can get a running time bound of $O^*(c^{f(n)})$. We have the following lemma for the relation between $p$ and $n$.

**Lemma 1.** *For a graph of $n$ vertices, if the average degree of the graph is at most $x$, then the measure $p$ of the graph is at most* $(0.624x - 0.872)n$.

*Proof.* According to the definition of the measure, we have that

$$p = \sum_{i=1}^{n} n_i \delta_i = \sum_{i=2}^{n} n_i \delta_i$$

$$= \sum_{i=2}^{n} n_i \delta_2 + \sum_{i=2}^{n} n_i (i-2) \delta_3^{<-1>}.$$

Since the average degree of the graph is at most $x$, we have that

$$\sum_{i=0}^{n} n_i i \leq x \sum_{i=0}^{n} n_i.$$

Thus,

$$p \leq \sum_{i=2}^{n} n_i \delta_2 + \sum_{i=0}^{n} n_i (x-2) \delta_3^{<-1>} \leq n(\delta_2 + (x-2)\delta_3^{<-1>})$$
$$= n + (x-3)\delta_3^{<-1>} n = (0.624x - 0.872)n.$$

## 3 Reduction Rules

We first introduce some reduction rules, which can be applied to reduce the instance directly by eliminating some local structures of the graph. Reduction rules for the unweighted case have been extensively studied. However, most of they do not work in weighted graphs. In weighted graphs, we may not be able to reduce all degree-2 vertices, which is an easy case in unweighted graphs. There are also two papers [19, 30] systematically study reduction rules for the weighted case. Here we try to contribute more reduction rules based on degree-2 vertices, small vertex-cuts, and some other special local structures.

Some reduction rules may include a set $S$ of vertices in the solution set directly. We use $M_c$ to store the weight of the vertices that have been included in the solution set. When a set $S$ of vertices is included in the solution set, we will remove $N[S]$ from the graph and update $M_c$ by adding $w(S)$.

### 3.1 General Reductions for Some Special Structures

We use several reduction rules based on unconfined vertices, twins, vertices with a clique neighborhood, and heavy vertices. Some of these reduction rules were introduced in [19] and [30].

**Unconfined Vertices.** A vertex $v$ in $G$ is called *removable* if $\alpha(G) = \alpha(G-v)$, i.e., there is a maximum weighted independent set in $G$ that does not contain $v$. We can say that a vertex $v$ is removable if a contradiction is obtained from the assumption that every maximum weighted independent set in $G$ contains $v$. A sufficient condition for a vertex to be removable in unweighted graphs has been studied in [32]. We extend this concept to weighted graphs.

For an independent set $S$ of $G$, a vertex $u \in N(S)$ is called a *child* of $S$ if $w(u) \geq w(S \cap N(u))$. A child $u$ is called an *extending child* if it holds that $|N(u) \setminus N[S]| = 1$, and the only vertex $v \in N(u) \setminus N[S]$ is called a *satellite* of $S$. See Fig. 1 for an illustration.

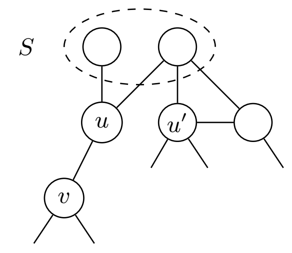

**Fig. 1.** An independent $S$ of $G$, where vertices $u$ and $u'$ are children of $S$, vertex $u$ is an extending child and $v$ is a satellite of $S$

**Lemma 2.** *Let $S$ be an independent set that is contained in any maximum weighted independent set in $G$. Then every maximum weighted independent set contains at least one vertex in $N(u) \setminus N[S]$ for each child $u$ of $S$.*

*Proof.* Assume to the contrary that there is a maximum weighted independent set $S_G$ in $G$ such that $S_G \cap (N(u) \setminus N[S]) = \emptyset$ for some child $u$ of $S$. We have that $S \subseteq S_G$ by the assumption on the independent set $S$. Thus, we could replace $S \cap N(u)$ with $u$ to obtain another independent set $S'_G = (S_G \setminus N(u)) \cup \{u\}$ in $G$. Furthermore, since $u$ is a child of $S$, we have that $w(u) \geq w(S \cap N(u))$. Thus, $S'_G$ is also a maximum weighted independent set in $G$. However $S'_G$ does not contain $S$, contradicting that $S$ is contained in any maximum weighted independent set in $G$.

Lemma 2 provides a sufficient condition for a vertex set contained in any maximum weighted independent set. Next, we introduce a method based on Lemma 2 to find some possible removable vertices.

Let $v$ be an arbitrary vertex in the graph. After starting with $S := \{v\}$, we repeat (1) until (2) or (3) holds:

(1) If $S$ has some extending child in $N(S)$, then let $S'$ be the set of satellites. Update $S$ by letting $S := S \cup S'$.
(2) If $S$ is not an independent set or there is a child $u$ such that $N(u) \setminus N[S] = \emptyset$, then halt and conclude that $v$ is *unconfined.*
(3) If $|N(u) \setminus N[S]| \geq 2$ for all children $u \in N(S)$, then halt and return $S_v = S$.

Obviously, the procedure can be executed in polynomial time for any starting set $S$ of a vertex. If the procedure halts in (2), we say vertex $v$ *unconfined.* If the procedure halts in (3), then we say that the set $S_v$ returned in (3) *confines* vertex $v$ and vertex $v$ is also called *confined.* Note that the set $S_v$ confining $v$ is uniquely determined by the procedure with starting set $S := \{v\}$. It is easy to observe the following lemma.

**Lemma 3.** *Any unconfined vertex is removable.*

*Proof.* Assume that vertex $v$ is contained in all maximum weighted independent sets. By Lemma 2, we know that after each execution of (1) of the above procedure, the set $S$ should also be contained in all maximum weighted independent sets. However, when the procedure halts in (2), the final set $S$ could not be contained in any maximum weighted independent set by Lemma 2. So $v$ is removable.

**Reduction Rule 1 (R1)** *If a vertex $v$ is unconfined, remove $v$ from $G$.*

We here observed some structures that are involved in unconfined vertices. We say that a vertex $v$ *dominated* by a neighbor $u$ of it if $v$ is adjacent to all neighbors of $u$, i.e., $N[u] \subseteq N[v]$. Clearly, any dominated vertex $v$ with $w(v) \leq w(u)$ is unconfined, since $S = \{v\}$ has a child $u$ with $N(u) \setminus N[S] = \emptyset$. See Fig. 2 for an illustration of this case. For a degree-1 vertex $u$ with the unique neighbor $v$, if $w(v) \leq w(u)$, then vertex $v$ is unconfined. For a degree-2 vertex $u$ with two adjacent neighbors, if one neighbor, say $v$, holds that $w(v) \leq w(u)$, then vertex $v$ is unconfined. R1 can deal with some degree-1 and degree-2 vertices, but not all of them.

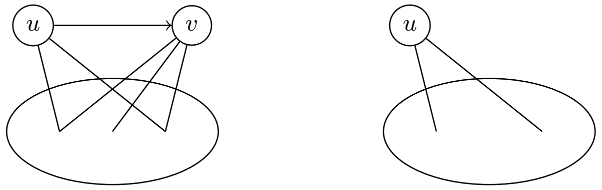

**Fig. 2.** (a) the graph $G$, where vertex $v$ is dominated by vertex $u$, and $v$ is removable; (b) the graph $G'$ that is obtained from $G$ by deleting the removable vertex $v$

**Twins.** A set $A = \{u, v\}$ of two non-adjacent vertices is called a *twin* if they have the same neighbor set, i.e., $N(u) = N(v)$. Reductions based on twins are used not only for independent sets [1, 19] but also for feedback sets and other problems [21]. Clearly, a vertex in a twin is in a maximum weighted independent set if and only if the other vertex in the twin is also in the same maximum weighted independent set. So we can treat the two vertices in a twin as a single vertex.

**Reduction Rule 2 (R2)** *[19] If there is a twin $A = \{u, v\}$, delete $v$ and update the weight of $u$ by letting $w(u) := w(u) + w(v)$.*

See Fig. 3 for an illustration of R2.

**Clique Neighborhood**. A vertex $v$ has a *clique neighborhood* if the graph $G[N(v)]$ induced by the open neighbor set of $v$ is a clique, which was introduced as isolated vertices in [19].

**Reduction Rule 3 (R3)** *[19] If there is a vertex $v$ having a clique neighborhood and $w(v) < w(u)$ holds for all $u \in N(v)$, then remove $v$ from the graph, update the weight $w(u) := w(u) - w(v)$ for all $u \in N_G(v)$, and add $w(v)$ to $M_c$.*

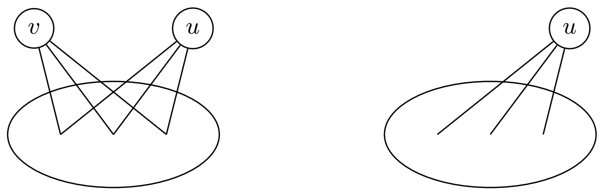

**Fig. 3.** (a) the graph $G$ that has a twin $\{u, v\}$; (b) the graph $G'$ after deleting $v$ from $G$ and updating the weight of $u$ by letting $w(u) = w(u) + w(v)$

An illustration of which is shown in Fig. 4.

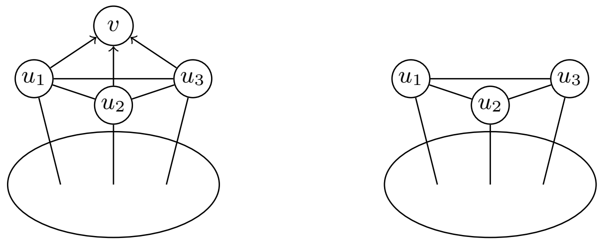

**Fig. 4.** (a) the graph $G$, where vertex $v$ has a clique neighborhood and the weight of $v$ is less than the weight of any neighbor of it; (b) the graph $G'$ after deleting $v$ from $G$ and updating the weight $w(u) := w(u) - w(v)$ for all $u \in N_G(v)$

**Heavy Vertices**. A vertex $v$ is called a *heavy vertex* if its weight is not less the weight of the maximum weighted independent set in subgraph induced by the open neighborhood of it, i.e.,

$$w(v) \geq \alpha(G[N(v)]).$$

We can see that for each heavy vertex, there is a maximum weighted independent set contain it. Note that if there is a maximum weighted independent set $S$ does not contain $v$, then we can replace $N(v) \cap S$ with $v$ in $S$ to get another maximum weighted independent set, where $w(N(v) \cap S) \leq w(v)$ since $v$ is a heavy vertex. Dealing heavy vertices is a simple but very efficient method to reduce the graph that have been used in some experimental algorithms [19, 30]. Whether a vertex is a heavy vertex can be checked in constant time if the degree of the vertex is bounded by a constant. In this paper, we will only check heavy vertices of degree bounded by 5. Note that degree-0 vertices are heavy vertices and we can reduce degree-0 vertices in this step.

**Reduction Rule 4 (R4)** *If there is a heavy vertex $v$ of degree at most 5, then delete $N[v]$ from the graph and add $w(v)$ to $M_c$.*

### 3.2 Reductions Based on Degree-2 Vertices

For unweighted graphs, we have good reduction rules to deal with all degree-2 vertices (see the reduction rule in [6]). However, for weighted graphs, it becomes much more complicated. Although we have several reduction rules, we can not deal with all degree-2 vertices.

Our first rule is generalized from the concept of folding degree-2 vertices in unweighted graphs introduced in [6], which has been also used in some experimental algorithms. A proof of the correctness can be found in [19, 30].

**Reduction Rule 5 (R5)** *If there is a degree-2 vertex $v$ with two neighbors $\{u_1, u_2\}$ such that $w(u_1) + w(u_2) > w(v) \geq max\{w(u_1), w(u_2)\}$, then delete $\{v, u_1, u_2\}$ from the graph $G$, introduce a new vertex $v'$ adjacent to $N_G(\{v, u_1, u_2\})$ with weight $w(v') := w(u_1) + w(u_2) - w(v)$, and add $w(v)$ to $M_c$.*

Fig. 5 gives an illustration of R5.

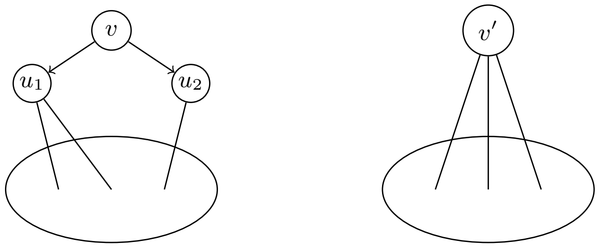

**Fig. 5.** (a) the graph $G$ having a degree-2 vertex $v$ with two neighbors $u_1$ and $u_2$; (b) the graph $G'$ after deleting $\{v, u_1, u_2\}$ and introducing the new vertex $v'$

The following two reductions are special cases of alternative sets introduced in [30]. We also use them to reduce some degree-2 vertices in the graph.

**Reduction Rule 6 (R6)** *If there is a path $v_1v_2v_3v_4$ such that $d_G(v_2) = d_G(v_3) = 2$ and $w(v_1) \geq w(v_2) \geq w(v_3) \geq w(v_4)$, then remove $v_2$ and $v_3$ from the graph, add an edge $v_1v_4$ if it does not exist, update the weight of $v_1$ by letting $w(v_1) := w(v_1) + w(v_3) - w(v_2)$, and add $w(v_2)$ to $M_c$.*

Fig. 6 gives an illustration of R6. A proof of the correctness of this reduction rule is given below.

**Lemma 4.** *Let $G'$ be a graph obtained from $G$ by applying R6, then $\alpha(G) = \alpha(G') + w(v_2)$.*

*Proof.* Let $S$ be a maximum weighted independent set in $G$. If $S$ contains $v_1$, we can assume that $S$ also contains $v_3$, because $S$ must contain one of $v_3$ and $v_4$ due to the maximality of $S$ and if $S$ contains $v_4$ we can replace $v_4$ with $v_3$ without

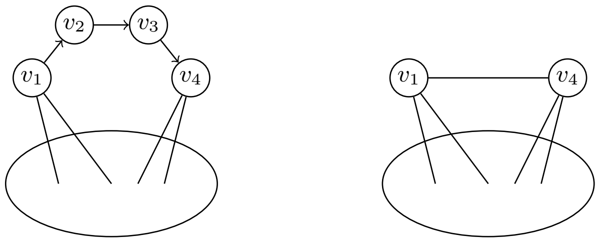

**Fig. 6.** (a) the graph $G$, where $v_2$ and $v_3$ are two adjacent degree-2 vertices; (b) the graph $G'$ after deleting $v_2$ and $v_3$ and adding the edge $v_1v_4$

decreasing the total weight of the independent set. For this case, $S' = S \setminus \{v_3\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) - w_G(v_2)$. If $S$ does not contain $v_1$, we can assume that $S$ contains $v_2$. For this case, $S' = S \setminus \{v_2\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) - w_G(v_2)$.

On the other hand, for any maximum weighted independent set $S'$ in $G'$, we can construct a weight independent set $S$ of $G$ such that $w_G(S) = w_{G'}(S') + w_G(v_2)$. If $v_1 \in S'$, then $S = S' \cup \{v_3\}$ is an independent set in $G$ with weight $w_G(S) = w_{G'}(S') + w_G(v_2)$. If $v_1 \notin S'$, then $S = S' \cup \{v_2\}$ is an independent set in $G$ with weight $w_G(S) = w_{G'}(S') + w_G(v_2)$.

**Reduction Rule 7 (R7)** *If there is a 4-cycle $v_1v_2v_3v_4$ such that $d_G(v_2) = d_G(v_3) = 2$ and $w(v_1) \geq w(v_2) \geq w(v_3)$, then remove $v_2$ and $v_3$, update the weight of $v_1$ by letting $w(v_1) := w(v_1) + w(v_3) - w(v_2)$, and add $w(v_2)$ to $M_c$.*

Fig. 7 gives an illustration of R7. A proof of the correctness of this reduction rule is given below.

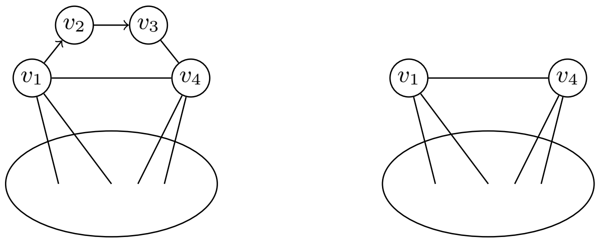

**Fig. 7.** (a) the graph $G$, where $v_2$ and $v_3$ are two adjacent degree-2 vertices in a 4-cycle $v_1v_2v_3v_4$; (b) the graph $G'$ after deleting $v_2$ and $v_3$ from $G$

**Lemma 5.** *Let $G'$ be a graph obtained from $G$ by applying R7, then $\alpha(G) = \alpha(G') + w(v_2)$.*

*Proof.* This lemma can be proved analogously with the proof of Lemma 4.

Let $S$ be a maximum weighted independent set in $G$. If $S$ contains $v_1$, we can assume that $S$ also contains $v_3$, because $S$ can not contain $v_1$ and $v_4$ now. For this case, $S' = S \setminus \{v_3\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) - w_G(v_2)$. If $S$ does not contain $v_1$, we can assume that $S$ contains $v_2$, since $S$ must contain at least one of $v_1$ and $v_2$ and $w(v_1) \geq w(v_2)$. For this case, $S' = S \setminus \{v_2\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) - w_G(v_2)$.

On the other hand, for any maximum weighted independent set $S'$ in $G'$, we can construct a weight independent set $S$ of $G$ such that $w_G(S) = w_{G'}(S') + w_G(v_2)$. If $v_1 \in S'$, then $S = S' \cup \{v_3\}$ is an independent set in $G$ with weight $w_G(S) = w_{G'}(S') + w_G(v_2)$. If $v_1 \notin S'$, then $S = S' \cup \{v_2\}$ is an independent set in $G$ with weight $w_G(S) = w_{G'}(S') + w_G(v_2)$.

Next, we introduce more rules for degree-2 vertices in some complicated structures.

**Reduction Rule 8 (R8)** *If there is a 4-path $v_1v_2v_3v_4v_5$ such that $d_G(v_2) = d_G(v_3) = d_G(v_4) = 2$ and $w(v_1) \geq w(v_2) \geq w(v_3) \leq w(v_4) \leq w(v_5)$, then remove $v_2$ and $v_4$, add edges $v_1v_3$ and $v_3v_5$, update the weight of $v_1$ by letting $w(v_1) := w(v_1) + w(v_3) - w(v_2)$ and the weight of $v_5$ by letting $w(v_5) := w(v_5) + w(v_3) - w(v_4)$, and add $w(v_2) + w(v_4) - w(v_3)$ to $M_c$.*

Fig. 8 gives an illustration of R8. The correctness of this reduction rule is based on the following lemma.

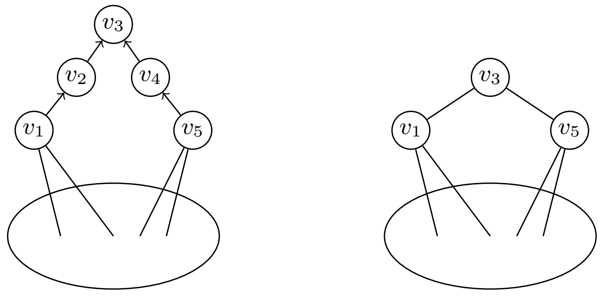

**Fig. 8.** (a) the graph $G$ with three adjacent degree-2 vertices $v_2, v_3$ and $v_4$ in a 4-path; (b) the graph $G'$ after deleting $v_2$ and $v_4$ and adding edges $v_2v_3$ and $v_3v_4$

**Lemma 6.** *Let $G'$ be a graph obtained from $G$ by applying R8, then $\alpha(G) = \alpha(G') + w(v_2) + w(v_4) - w(v_3)$.*

*Proof.* Let $S$ be a maximum weighted independent set in $G$. We consider the following four cases.

Case 1. $S$ contains both of $v_1$ and $v_5$: For this case, we can assume that $v_3$ is also in $S$ by the maximality of $S$. We can see that $S' = S \setminus \{v_3\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) + w_G(v_3) - w_G(v_2) + w_G(v_3) - w_G(v_4) - w_G(w_3) = w_G(S) + w_G(v_3) - w_G(v_2) - w_G(v_4)$.

Case 2. $S$ contains $v_1$ but not $v_5$: For this case, we can assume that $S$ also contains $v_4$, because $S$ must contain one of $v_3$ and $v_4$ due to the maximality of $S$ and $w(v_3) \leq w(v_4)$. For this case, $S' = S \setminus \{v_4\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) + w_G(v_3) - w_G(v_2) - w_G(v_4)$.

Case 3. $S$ contains $v_5$ but not $v_1$: For this case, we can assume that $S$ also contains $v_2$, because $S$ must contain one of $v_3$ and $v_2$ due to the maximality of $S$ and $w(v_3) \leq w(v_2)$. For this case, $S' = S \setminus \{v_2\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) + w_G(v_3) - w_G(v_2) - w_G(v_4)$.

Case 4. $S$ contains none of $v_1$ and $v_5$: For this case, we can assume that $S$ contains both of $v_2$ and $v_4$. For this case, $S' = S \cup \{v_3\} \setminus \{v_2, v_4\}$ is an independent set in $G'$ with weight $w_{G'}(S') = w_G(S) + w_G(v_3) - w_G(v_2) - w_G(v_4)$.

On the other hand, for any maximum weighted independent set $S'$ in $G'$, we can construct a weight independent set $S$ in $G$ such that $w_G(S) = w_{G'}(S') + w_G(v_2) + w_G(v_4) - w_G(v_3)$. If $S'$ does not contain any of $v_1$ and $v_5$, then it must contain $v_3$ due to the maximality of $S$. For this case, $S = S' \cup \{v_2, v_4\} \setminus \{v_3\}$ is an satisfied independent set in $G$. If $S'$ contains both of $v_1$ and $v_5$, then $S = S' \cup \{v_3\}$ is an satisfied independent set in $G$. If $S'$ contains $v_1$ but not $v_5$, then $S = S' \cup \{v_4\}$ is an satisfied independent set in $G$. If $S'$ contains $v_5$ but not $v_1$, then $S = S' \cup \{v_2\}$ is an satisfied independent set in $G$.

**Reduction Rule 9 (R9)** *For a 5-cycle $v_1v_2v_3v_4v_5$ such that $d_G(v_2) = d_G(v_3) = d_G(v_5) = 2$, $\min\{d(v_1), d(v_4)\} \geq 3$, and $w(v_1) \geq w(v_2) \geq w(v_3) \leq w(v_4)$,*

*(1) if $w(v_3) > w(v_5)$, then remove $v_5$, update the weight of $v_i$ by letting $w(v_i) := w(v_i) - w(v_5)$ for $i = 1, 2, 3, 4$, and add $2w(v_5)$ to $M_c$.*

*(2) if $w(v_3) \leq w(v_5)$, then remove $v_2$ and $v_3$, update the weight of $v_1$ by letting $w(v_1) := w(v_1) - w(v_2)$, the weight of $v_4$ by letting $w(v_4) := w(v_4) - w(v_3)$ and the weight of $v_5$ by letting $w(v_5) := w(v_5) - w(v_3)$, and add $w(v_2) + w(v_3)$ to $M_c$.*

Fig. 9 gives an illustration of R9. The correctness of this reduction rule is based on the following lemma.

**Lemma 7.** *Let $G'$ be the graph obtained from $G$ by applying R9. If (1) of R9 is applied, then*

$$\alpha(G) = \alpha(G') + 2w(v_5);$$

*If (2) of R9 is applied, then*

$$\alpha(G) = \alpha(G') + w(v_2) + w(v_3).$$

*Proof.* It is easy to observe that any maximum weighted independent set $S$ will contain exactly two non-adjacent vertices in the 5-cycle $v_1v_2v_3v_4v_5$. There are only five possible cases. If $\{v_3, v_5\} \subseteq S$, then $S \cup \{v_2\} \setminus \{v_3\}$ is another independent set in $G$ with the weight at least $w(S)$. So we can assume that $S$ contains one of $\{v_1, v_4\}$, $\{v_1, v_3\}$, $\{v_2, v_4\}$ and $\{v_2, v_5\}$.

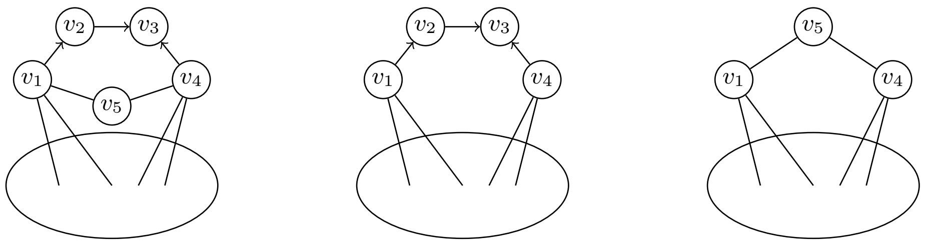

**Fig. 9.** (a) the graph $G$ containing a 5-cycle $v_1v_2v_3v_4v_5$ with three degree-2 vertices $v_2, v_3$ and $v_5$; (b) the graph $G'$ after applying (1) in R9 on $G$; (c) the graph $G'$ after applying (2) in R9 on $G$

Case (1): We assume that (1) of R9 is applied, where $w(v_3) > w(v_5)$. Let $S^*$ denote one of $\{v_1, v_4\}$, $\{v_1, v_3\}$ and $\{v_2, v_4\}$. If $S^* \subseteq S$, then $S' = S$ is an independent set in $G'$ such that $w_{G'}(S') = w_G(S \setminus S^*) + w_{G'}(S^*) = w_G(S) - 2w_G(v_5)$. Otherwise, $\{v_2, v_5\} \subseteq S$. For this case, $S' = S \setminus \{v_5\}$ is a weight independent set in $G'$ such that $w_{G'}(S') = w_G(S' \setminus \{v_2\}) + w_{G'}(v_2) = w_G(S) - 2w_G(v_5)$.

On the other hand, for any maximum weighted independent set $S'$ in $G'$, we can construct a weight independent set $S$ in $G$ such that $w_G(S) = w_{G'}(S') + 2w_G(v_5)$. We can assume that $S' \cap \{v_1, v_2, v_3, v_4\}$ is one of $\{v_1, v_3\}$, $\{v_2, v_4\}$, $\{v_1, v_4\}$ and $\{v_2\}$, because if $S'$ contains $v_1$ then $S'$ also contains one of $v_3$ and $v_4$ and if $S'$ does not contain $v_1$ then we can assume that $S'$ contains $v_2$.

Let $S^*$ denote one of $\{v_1, v_4\}$, $\{v_1, v_3\}$ and $\{v_2, v_4\}$. If $S^* \subseteq S'$, then $S = S'$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S' \setminus S^*) + w_G(S^*) = w_{G'}(S') + 2w_G(v_5)$. Otherwise, $v_2 \in S'$ but $v_4 \notin S'$. For this case, $S = S' \cup \{v_5\}$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S' \setminus \{v_2\}) + w_G(\{v_2, v_5\}) = w_{G'}(S') + 2w_G(v_5)$.

Case (2): We assume that (2) of R9 is applied, where $w(v_3) \leq w(v_5)$. Let $S^*$ denote one of $\{v_1, v_4\}$, $\{v_1, v_3\}$ and $\{v_2, v_4\}$. If $S^* \subseteq S$, then $S' = S \setminus \{v_2, v_3\}$ is an independent set in $G'$ such that $w_{G'}(S') = w_G(S \setminus (S^* \cup \{v_2, v_3\})) + w_{G'}(S^* \setminus \{v_2, v_3\}) = w_G(S) - w_G(\{v_2, v_3\})$. Otherwise, $\{v_2, v_5\} \subseteq S$. For this case, $S' = S \setminus \{v_2\}$ is an independent set in $G'$ such that $w_{G'}(S') = w_G(S' \setminus \{v_5\}) + w_{G'}(v_5) = w_G(S) - w_G(\{v_2, v_3\})$.

On the other hand, for any maximum weighted independent set $S'$ in $G'$, we can construct a weight independent set $S$ in $G$ such that $w_G(S) = w_{G'}(S') + w_G(\{v_2, v_3\})$. If $\{v_1, v_4\} \subseteq S'$, then $S = S'$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S' \setminus \{v_1, v_4\}) + w_G(\{v_1, v_4\}) = w_{G'}(S') + w_G(\{v_2, v_3\})$. If exactly one vertex of $v_1$ and $v_4$, say $v$ is in $S'$, then $S = S' \cup (\{v_2, v_3\} \setminus N(v))$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S' \setminus \{v\}) + w_G(\{v\} \cup \{v_2, v_3\} \setminus N(v)) = w_{G'}(S') + w_G(\{v_2, v_3\})$. Otherwise, $v_5 \in S'$. For this case, $S = S' \cup \{v_2\}$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S' \setminus \{v_5\}) + w_G(\{v_2, v_5\}) = w_{G'}(S') + w_G(\{v_2, v_3\})$.

**Reduction Rule 10 (R10)** *For a 6-cycle $v_1v_2v_3v_4v_5v_6$ such that $d_G(v_2) = d_G(v_3) = d_G(v_5) = d_G(v_6) = 2$, $w(v_1) \geq \max\{w(v_2), w(v_6)\}$, $w(v_4) \geq \max\{w(v_3), w(v_5)\}$, and $w(v_6) \geq w(v_5)$,*

*(1) if $w(v_2) \geq w(v_3)$, then remove $v_5$ and $v_6$, and update the weight of $v_2$ by letting $w(v_2) := w(v_2) + w(v_6)$ and the weight of $v_3$ by letting $w(v_3) := w(v_3) + w(v_5)$;*

*(2) if $w(v_2) < w(v_3)$, then remove $v_6$, add edge $v_1v_5$, and update the weight of $v_2$ by letting $w(v_2) := w(v_2) + w(v_6)$, the weight of $v_3$ by letting $w(v_3) := w(v_3) + w(v_5)$, and the weight of $v_5$ by letting $w(v_5) := w(v_6) + w(v_3) - \max\{w(v_2) + w(v_6), w(v_3) + w(v_5)\}$.*

Fig. 10 gives an illustration of R10. The correctness of this reduction rule is based on the following lemma.

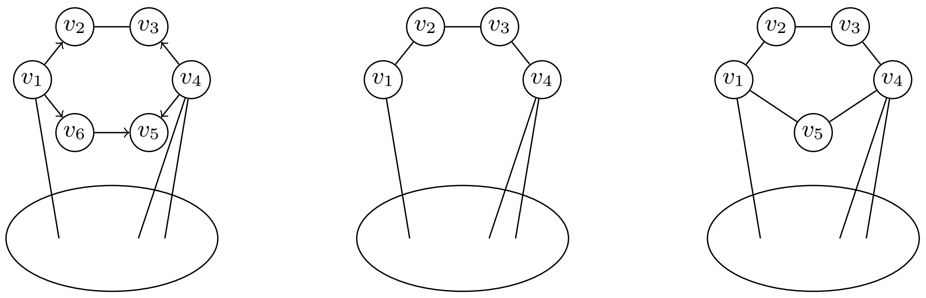

**Fig. 10.** (a) the graph $G$ containing a 6-cycle $v_1v_2v_3v_4v_5v_6$ with four degree-2 vertices $v_2, v_3, v_5$ and $v_6$; (b) the graph $G'$ after applying (1) in R10 on $G$; (c) the graph $G'$ after applying (2) in R10 on $G$

**Lemma 8.** *Let $G'$ be the graph obtained from $G$ by applying R10, then $\alpha(G) = \alpha(G')$.*

*Proof.* Case (1): We assume that (1) of R10 is applied, where $w(v_2) \geq w(v_3)$. Let $S$ be a maximum weighted independent set in $G$.

We show that, without loss of generality, we can assume that $S$ contains both of $v_2$ and $v_6$ or none of them (resp., both of $v_3$ and $v_5$ or none of them). The reason is based on the following observation. If $S$ contains $v_2$ but not $v_6$, then $S$ must contain $v_5$, otherwise $v_6$ should be added to $S$ directly by the maximality of $S$. For this case, we can replace $v_5$ with $v_6$ in $S$ to get another maximum weighted independent set since $w(v_6) \geq w(v_5)$. If $S$ contains $v_6$ but not $v_2$, then $S$ must contain $v_3$, otherwise $v_2$ should be added to $S$ directly by the maximality of $S$. For this case, we can replace $v_3$ with $v_2$ in $S$ to get another maximum weighted independent set since for this case we have that $w(v_2) \geq w(v_3)$. So we can assume that $S$ contains either both of $v_2$ and $v_6$ or none of them. If $S$ contains $v_3$ but not $v_5$, then $S$ must contain $v_6$, otherwise $v_5$ should be added to $S$ directly by the maximality of $S$. For this case, we can replace $v_3$ with $v_2$ in $S$ to get another maximum weighted independent set since $w(v_2) \geq w(v_3)$. Now the set $S$ contains both of $v_2$ and $v_6$ but none of $v_3$ and $v_5$. If $S$ contains $v_5$ but not $v_3$, then $S$ must contain $v_2$, otherwise $v_3$ should be added to $S$ directly by the maximality of $S$. For this case, we can replace $v_5$ with $v_6$ in $S$ to get another maximum weighted independent set since $w(v_6) \geq w(v_5)$. We also get a maximum weighted independent set that contains both of $v_2$ and $v_6$ but none of $v_3$ and $v_5$.

If $\{v_2, v_6\} \subseteq S$, then $S' = S \setminus \{v_6\}$ is a weight independent set in $G'$ with weight $w_{G'}(S') = w(S)$. If $\{v_3, v_5\} \subseteq S$, then $S' = S \setminus \{v_5\}$ is a weight independent set in $G'$ with weight $w_{G'}(S') = w(S)$.

On the other hand, for any maximum weighted independent set $S'$ in $G'$, we can construct a weight independent set $S$ in $G$ such that $w_G(S) = w_{G'}(S')$. If $v_2 \in S'$, then $S = S' \cup \{v_6\}$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S')$. If $v_3 \in S'$, then $S = S' \cup \{v_5\}$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S')$.

Case (2): We assume that (2) of R10 is applied, where $w(v_2) < w(v_3)$. Let $S$ be a maximum weighted independent set in $G$.

If $S$ contains $v_1$, then $S$ will contain either $v_4$ or both of $v_3$ and $v_5$ (after deleting $N[v_1]$ from the graph $\{v_3, v_5\}$ will be a twin). If $S$ does not contain $v_1$, then we can assume that $S$ contain $v_6$ since $w(v_6) \geq w(v_5)$. For this case, $S$ must contain one of $v_2$ and $v_3$ due to the maximality of $S$. Furthermore, if $S \cap \{v_1, v_2, v_3, v_4, v_5, v_6\} = \{v_2, v_6\}$, then we can replace $v_2$ with $v_3$ in $S$ to get another maximum independent set. So we can assume that $S_\Delta = S \cap \{v_1, v_2, v_3, v_4, v_5, v_6\}$ is one of $\{v_1, v_3, v_5\}$, $\{v_1, v_4\}$, $\{v_2, v_4, v_6\}$, and $\{v_3, v_6\}$.

If $S_\Delta$ is $\{v_1, v_3, v_5\}$ or $\{v_2, v_4, v_6\}$, then $S' = S \setminus \{v_5, v_6\}$ is an independent set in $G'$ such that $w_{G'}(S') = w_G(S' \setminus \{v_2, v_3\}) + w_{G'}(\{v_2, v_3\} \cap S') = w_G(S)$. If $S_\Delta = \{v_1, v_4\}$, then $S' = S$ is an independent set in $G'$ such that $w_{G'}(S') = w_G(S)$. Otherwise $S_\Delta = \{v_3, v_6\}$. For this case, $S' = (S \setminus \{v_3, v_6\}) \cup \{v^*, v_5\}$ is an independent set in $G'$, where $v^*$ is the vertex in $\{v_2, v_3\}$ with the larger weight in $G'$. We have that $w_{G'}(S') = w_G(S \setminus \{v_3, v_6\}) + w_{G'}(\{v^*, v_5\}) = w_G(S)$.

On the other hand, for any maximum weighted independent set $S'$ in $G'$, we can construct a weight independent set $S$ in $G$ such that $w_G(S) = w_{G'}(S')$. In $G'$, there is a 5-cycle $v_1v_2v_3v_4v_5$ and any maximum weighted independent set in $G'$ contains exactly two vertices in the 5-cycle. Recall that $v^*$ is the vertex in $\{v_2, v_3\}$ with the larger weight in $G'$. Let $\{v^\star, v^*\} = \{v_2, v_3\}$. We have that $w_{G'}(v^*) \geq w_{G'}(v^\star)$. If $S'$ contains $\{v^\star, v_5\}$, then we can replace $v^\star$ with $v^*$ in $S'$. So we can assume that $S'_\Delta = S' \cap \{v_1, v_2, v_3, v_4, v_5\}$ is one of the four cases $\{v_1, v_3\}$, $\{v_2, v_4\}$, $\{v_1, v_4\}$ and $\{v^*, v_5\}$.

If $S'_\Delta = \{v_1, v_3\}$, then $S = S' \cup \{v_5\}$ is an independent set in $G$ such that $w_G(S) = w_G(S' \setminus \{v_1, v_3\}) + w_G(\{v_1, v_3, v_5\}) = w_{G'}(S')$. If $S'_\Delta = \{v_2, v_4\}$, then $S = S' \cup \{v_6\}$ is an independent set in $G$ such that $w_G(S) = w_G(S' \setminus \{v_2, v_4\}) + w_G(\{v_2, v_4, v_6\}) = w_{G'}(S')$. If $S'_\Delta = \{v_1, v_4\}$, then $S = S'$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S')$. Otherwise, $S'_\Delta = \{v^*, v_5\}$. For this case,

$S = (S' \setminus \{v^*, v_5\}) \cup \{v_3, v_6\}$ is an independent set in $G$ such that $w_G(S) = w_{G'}(S' \setminus \{v, v_5\}) + w_G(\{v_3, v_6\}) = w_{G'}(S')$.

## 3.3 Reductions Based on Small Cuts

We also have some reduction rules to deal with vertex-cuts of size one or two, which can even be used to design a polynomial-time divide-and-conquer algorithm. However, a graph may not always have vertex-cuts of small size.

**Vertex-Cuts of Size One.** We first introduce the reduction rule based on vertex-cuts of size one.

**Lemma 9.** *Let $\{u\}$ be a vertex-cut of size one in $G$ and $G^*$ be a connected component in $G - u$.*

*(1) if $w(u) + \alpha(G^* - N[u]) \le \alpha(G^*)$, then $\alpha(G) = \alpha(G_1) + \alpha(G^*)$, where $G_1$ is the remaining graph after removing $G^*$ and $u$ from $G$;*
*(2) if $w(u) + \alpha(G^* - N[u]) > \alpha(G^*)$, then $\alpha(G) = \alpha(G_2) + \alpha(G^*)$, where $G_2$ is the remaining graph after removing $G^*$ from $G$ and updating the weight of $u$ by letting $w(u) := w(u) + \alpha(G^* - N[u]) - \alpha(G^*)$.*

*Proof.* Let $G' = G - G^* - N[u]$ and $G'' = G^* - N[u]$. Let $S$ be a maximum weighted independent set in $G$.

Case (1): $\alpha(G'') + w(u) \le \alpha(G^*)$. If $u \in S$, then we can see that $\alpha(G) = \alpha(G') + \alpha(G'') + w(u)$. Recall that we use $S(G)$ to denote a maximum independent set in $G$. We can replace $S \cap (\{u\} \cup V(G''))$ with $S(G^*)$ in $S$ to get another independent set without decreasing the total weight since $\alpha(G'')+w(u) \le \alpha(G^*)$. Furthermore, we can replace $S \cap V(G')$ with $S(G_1)$ in $S$ to get another independent set in $G$. Thus, $w(S \cap V(G')) = \alpha(G_1)$ and then $\alpha(G) = \alpha(G_1) + \alpha(G^*)$. Otherwise $u \notin S$ and it directly holds that $\alpha(G) = \alpha(G^*) + \alpha(G_1)$.

Case (2): $\alpha(G'') + w(u) > \alpha(G^*)$. If $u \in S$, then $S' = S \setminus V(G^*)$ is an independent set in $G_2$ with weight $w_{G_2}(S') = w_G(S' \setminus \{u\}) + w_{G_2}(u) = w_G(S' \setminus \{u\}) + w_G(u) + \alpha(G'') - \alpha(G^*) = \alpha(G) - \alpha(G^*)$. If $u \notin S$, then $S' = S \setminus V(G^*)$ is an independent set in $G'$ with $w_{G_2}(S') = w_G(S') = \alpha(G) - \alpha(G^*)$.

On the other hand, for any maximum weighted independent set $S'$ in $G_2$, we can construct an independent set $S_0$ in $G$ such that $w_G(S_0) = w_{G_2}(S')+\alpha(G^*) = \alpha(G_2) + \alpha(G^*)$. For the case that $u \in S'$, $S_0 = S' \cup S(G'')$ is an independent set in $G$. By $w_{G_2}(u) = w_G(u)+\alpha(G'')-\alpha(G^*)$, we get that $w_G(S_0) = w_G(S' \setminus \{u\})+ w_G(u) + \alpha(G'') = w_{G_2}(S') + \alpha(G^*) = \alpha(G_2) + \alpha(G^*)$. For the case that $u \notin S'$, $S_0 = S' \cup S(G^*)$ is an independent set in $G$ with $w_G(S_0) = w_{G_2}(S') + \alpha(G^*) = \alpha(G_2) + \alpha(G^*)$.

Fig. 11 shows a graph $G$ with a vertex-cut $\{u\}$. Lemma 9 provides a divide-and-conquer method based on vertex-cuts of size one. In our algorithm, we will only use it to deal with cuts that can split a connected component $G^*$ of bounded total vertex cost.

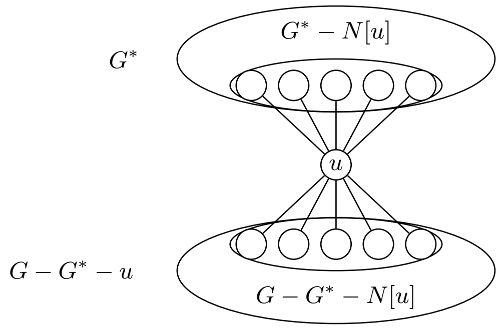

**Fig. 11.** A graph $G$ with a vertex-cut $\{u\}$

**Reduction Rule 11 (R11)** *For a vertex-cut $\{u\}$ with a connected component $G^*$ in $G-u$ such that $2\delta_3 - \delta_2 \le \sum_{v\in G^*} \delta_{d_G(v)} \le 10$,*

*(1) if $w(u) + \alpha(G^* - N[u]) \le \alpha(G^*)$, then remove $G^*$ and $\{u\}$ from $G$ and add $\alpha(G^*)$ to $M_c$;*
*(2) if $w(u) + \alpha(G^* - N[u]) > \alpha(G^*)$, then remove $G^*$ from $G$, update the weight of $u$ by letting $w(u) := w(u) + \alpha(G^* - N[u]) - \alpha(G^*)$, and add $\alpha(G^*)$ to $M_c$.*

**Vertex-Cuts of Size Two.** For vertex-cuts of size two, we have a similar result. However, it will become much more complicated.

**Lemma 10.** *Let $\{u, u'\}$ be a vertex-cut of size two in $G$ and $G^*$ be a connected component in $G - \{u, u'\}$, where we assume w.l.o.g. that $\alpha(G^* - N[u]) \ge \alpha(G^* - N[u'])$. We construct a new graph $G'$ from $G$ as follows: remove $G^*$; add three new vertices $\{v_1, v_2, v_3\}$ with weight $w(v_1) = \alpha(G^* - N[u']) - \alpha(G^* - N[\{u, u'\}])$, $w(v_2) = \alpha(G^* - N[u]) - \alpha(G^* - N[\{u, u'\}])$ and $w(v_3) = \alpha(G^*) - \alpha(G^* - N[u])$, and add five new edges $uv_1$, $v_1v_2$, $v_2u'$, $uv_3$ and $u'v_3$. It holds that*

$$\alpha(G) = \alpha(G') + \alpha(G^* - N[\{u, u'\}]).$$

*Proof.* Let $G'' = G^* - N[\{u, u'\}]$. Let $S$ be a maximum weight set in $G$.

First, we show that $G'$ can be obtained from $G$ with $\alpha(G') = \alpha(G) - \alpha(G'')$. If $u \in S$ but $u' \notin S$, then $S' = (S \setminus V(G^*)) \cup \{v_2\}$ is a weight independent set in $G'$ with weight $w_{G'}(S') = w_G(S \setminus V(G^*)) + w_{G'}(v_2) = \alpha(G) - \alpha(G'')$. If $u \notin S$ but $u' \in S$, then $S' = (S \setminus V(G^*)) \cup \{v_1\}$ is a weight independent set in $G'$ with weight $w_{G'}(S') = w_G(S \setminus V(G^*)) + w_{G'}(v_1) = \alpha(G) - \alpha(G'')$. If $u \in S$ and $u' \in S$, then $S' = (S \setminus V'(G^*))$ is a weight independent set in $G'$ with weight $w_{G'}(S') = w_G(S \setminus V(G^*)) = \alpha(G) - \alpha(G'')$. Otherwise, $u \notin S$ and $u' \notin S$. For this case, we let $S' = (S \setminus V') \cup \{v_2, v_3\}$, then $S'$ is a weight independent set in $G'$ with the weight $w_{G'}(S') = w_G(S \setminus V(G^*)) + w_{G'}(\{v_2, v_3\}) = \alpha(G) - \alpha(G'')$.

On the other hand, for any maximum weighted independent set $S'$ of $G'$, we can construct a weight independent set $S_0$ of $G$ such that $w_G(S_0) = w_{G'}(S') + \alpha(G'') = \alpha(G') + \alpha(G'')$. If $u \in S'$ but $u' \notin S'$, let $S_1$ be the maximum weighted independent set in $G^* - N_G[u]$, then $S_0 = (S_1 \cup S') \setminus \{v_1, v_2, v_3\}$ is a weight independent set in $G$ with weight $w_G(S_0) = w_G(S' \setminus \{v_2\}) + w_G(S_1) = \alpha(G') + \alpha(G'')$. If $u \notin S'$ but $u' \in S'$, let $S_2$ be the maximum weighted independent set in $G^* - N_G[u']$, then $S_0 = (S_2 \cup S') \setminus \{v_1, v_2, v_3\}$ is a weight independent set in $G$ with weight $w_G(S_0) = w_G(S' \setminus \{v_1\}) + w_G(S_3) = \alpha(G') + \alpha(G'')$. If $u \in S'$ but $u' \in S'$, let $S_3$ be the maximum weighted independent set in $G''$, then $S_0 = (S_3 \cup S') \setminus \{v_1, v_2, v_3\}$ is a weight independent set in $G$ with weight $w_G(S_0) = w_G(S' \setminus \{v_2\}) + w_G(S_1) = \alpha(G') + \alpha(G'')$. Otherwise, $u \notin S'$ but $u' \notin S'$. For this case, we let $S_4$ be the maximum weighted independent set in $G^*$, then $S_0 = (S' \cup S_4) \setminus \{v_1, v_2, v_3\}$ is a weight independent set in $G$ such that $w_G(S_0) = w_G(S' \setminus \{v_1, v_2, v_3\}) + w_G(S_4) = \alpha(G') + \alpha(G'')$.

Please see Fig. 12 for an illustration of the construction of $G'$ in Lemma 10. Based on Lemma 10, we have the following branching rule.

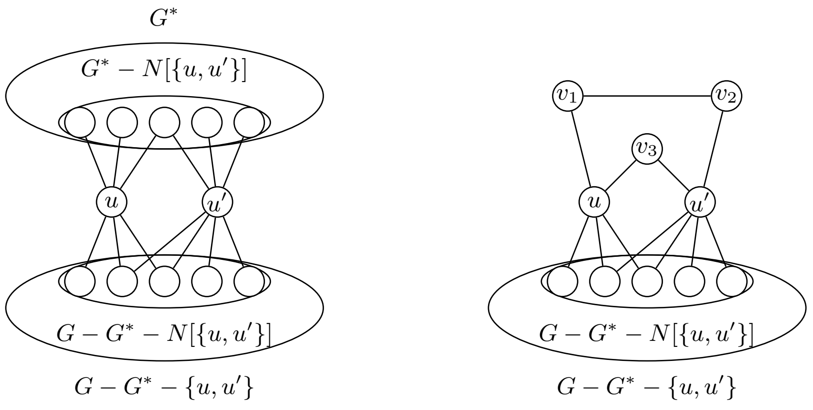

**Fig. 12.** (a) the graph $G$ with a vertex-cut $\{u, u'\}$ of size two; (b) the graph $G'$ constructed from $G$ by removing $G^*$ and introducing three new vertices $\{v_1, v_2, v_3\}$

**Reduction Rule 12 (R12)** *For a vertex-cut $\{u, u'\}$ of size two with a connected component $G^*$ in $G - \{u, u'\}$ such that $2\delta_3 + \delta_2 \leq \sum_{v \in G^*} \delta_{d_G(v)} \leq 10$, we construct the graph $G'$ in Lemma 10, replace $G$ with $G'$, and add $\alpha(G^* - N[\{u, u'\}])$ to $M_c$.*

## 3.4 Analyzing Reduction Rules

It is easy to see that all structures in our reduction rules are local structures that can be found in polynomial time. Each application of our reduction rules is to either remove some part of the graph or replace some part with a smaller structure, which can also be done in polynomial time. In this part, we mainly analyze how much the measure $p$ can be reduced in each reduction rule. Since we will apply these reduction rules in order, we assume without loss of generality that when one reduction rule is applied, no reduction rule with a smaller index can be applied.

**Lemma 11.** *Each application of our reduction rules can be executed in polynomial time.*

Each application of R1 to R4 will remove some vertices from the graph. The update of the vertex weight will not increase the measure $p$. Hence, we can get

**Lemma 12.** *Each application of R1 to R3 decreases the measure $p$ by at least $\delta_{d(v)} + \sum_{u \in N(v)} \delta^{<-1>}_{d(u)}$, where $v$ is the vertex on which the reduction rule is applied.*

**Lemma 13.** *Each application of R4 decreases the measure $p$ by at least $\sum_{u \in N[v]} \delta_{d(u)}$, where $v$ is the vertex on which the reduction rule is applied.*

For a degree-1 vertex $v$ with the unique neighbor $u$, if $w(v) < w(u)$ then we can apply R3 to reduce it, otherwise $w(v) \geq w(u)$ and we can apply R4 to reduce it. So we know that

**Lemma 14.** *If R1 to R4 can not be applied and the graph is not empty, then the minimum degree of the graph is at least 2.*

Let $C$ be a cycle that has a degree-2 vertex $v$. If $|C| = 3$, then either one of neighbors of $v$ is unconfined or $v$ can be reduced by R3. If $|C| = 4$ and there is a pair of nonadjacent degree-2 vertices, then R2 can be applied. So, we can get that

**Lemma 15.** *If R1 to R4 can not be applied and the graph is not empty, then there is no triangle containing degree-2 vertices and no 4-cycle containing two nonadjacent degree-2 vertices.*

**Lemma 16.** *If R1 to R5 can not be applied to $G$, then each connected component of the graph contains at least one vertex of degree $\geq 3$.*

*Proof.* By Lemma 14, we know that the graph has no degree-1 vertex. If there is a connected component containing only vertices of degree $\leq 2$, then the component can only be a cycle $C$ containing degree-2 vertices. By Lemma 15, we know that $|C| \geq 5$. Let $C = \{v_1, v_2, \ldots, v_{|C|}\}$. Since R5 can not be applied, we know that for each $1 \leq i \leq |C|$, it holds that $w(v_i) < \max\{w(v_{i+1}), w(v_{i-1})\}$, where $v_0 = v_{|C|}$ and $v_{|C|+1} = v_1$. However, it will not hold for the vertex $v^*$ with the maximum weight in the cycle, a contradiction. So the cycle $C$ containing only degree-2 vertices does not exist.

**Lemma 17.** *If R1 to R6 can not be applied to $G$, then each cycle $C$ contains at least two vertices of degree $\geq 3$.*

*Proof.* Let $C = \{v_0, v_1, v_2, \ldots, v_{|C|-1}\}$ be a cycle in the graph $G$. By Lemma 16, we know that $C$ must contain at least one vertex of degree $\geq 3$. Assume that there is exactly one vertex of degree $\geq 3$ in $C$, which is assumed to be $v_0$ without loss of generality. By Lemma 15, we know that $|C| \geq 5$. Since R5 can not be applied now, we know that $w(v_i) < \max\{w(v_{i+1}), w(v_{i-1})\}$ holds for all $1 \leq i < |C|$, where $v_{|C|} = v_0$. Then, we can get that $w(v_0) \geq \max\{w(v_1), w(v_{|C|-1})\}$. Thus, either $w(v_0) \geq w(v_1) \geq w(v_2) \geq w(v_3)$ or $w(v_{|C|}) \geq w(v_{|C|-1}) \geq w(v_{|C|-2}) \geq w(v_{|C|-3})$ holds, where implies that R6 can be applied. So $C$ must contain more than one vertices of degree $\geq 3$ and this lemma holds.

**Lemma 18.** *If R1 to R4 can not be applied, then each application of R5 to R8 decreases the measure p by at least* $2\delta_2$.

*Proof.* Assume that R1 to R4 can not be applied. By Lemma 14, we know that the minimum degree of $G$ is at least 2. We use $G'$ to denote the resulting graph after executing one application of R$i$ ($5 \leq i \leq 8$).

**Case 1:** R5 is applied. A degree-2 vertex $v$ with two neighbors $u_1$ and $u_2$ are deleted from the graph, and a new vertex $v'$ with degree at most $d(u_1) + d(u_2) - 2$ is introduced. For all other vertices, the degree will not increase. So the measure $p$ decreases by at least $\delta_2 + \delta_{d(u_1)} + \delta_{d(u_2)} - \delta_{d(u_1)+d(u_2)-2}$. Note that $\min\{d(u_1), d(u_2)\} \geq 2$. We have that $\delta_{d(u_1)} + \delta_{d(u_2)} - \delta_{d(u_1)+d(u_2)-2} \geq \delta_2 + \delta_2 - \delta_{2+2-2} = \delta_2$.

**Case 2:** R6 is applied. We replace a chain of two degree-2 vertices with an edge. Hence, the measure $p$ decreases by $p(G) - p(G') \geq 2\delta_2$.

**Case 3:** R7 is applied. In this case, we remove two degree-2 vertices from the graph. Hence, the measure $p$ decreases by at least $2\delta_2$.

**Case 4:** R8 is applied. In this case, two degree-2 vertices are replaced with an edge respectively. Hence, the measure $p$ decreases by $2\delta_2$.

**Lemma 19.** *If R1 to R6 can not be applied, then each application of R9 decreases the measure p by at least* $2\delta_3 - \delta_2$.

*Proof.* Let $v_1v_2v_3v_4v_5$ be the 5-cycle that R9 is applied to, where $v_2, v_3$ and $v_5$ are degree-2 vertices, as shown in Fig. 9. By Lemma 17, we know that $\min\{d(v_1), d(v_4)\} \geq 3$. In this reduction rule, we will remove either a degree-2 vertex $v_5$ or two adjacent degree-2 vertices $\{v_2, v_3\}$ from the graph. So, we can reduce the measure by at least $\delta_2$ from the removed vertices and at least $2\delta_3^{<-1>}$ from the neighbors of them.

**Lemma 20.** *If R1 to R6 can not be applied, then each application of R10(1) decreases the measure p by at least* $2\delta_3$ *and each application of R10(2) decreases the measure p by at least* $\delta_2$.

*Proof.* Let $v_1v_2v_3v_4v_5v_6$ be the 6-cycle that R10 is applied to, where $v_2, v_3, v_5$ and $v_6$ are degree-2 vertices, as shown in Fig. 10. By Lemma 17, we know that $\min\{d(v_1), d(v_4)\} \geq 3$. In this reduction rule, we will either remove two adjacent degree-2 vertices $\{v_5, v_6\}$ or replace a degree-2 vertex $v_6$ with an edge. For the first case, the measure will decrease by at least $2\delta_3$. For the second case, the measure will decrease by at least $\delta_2$.

**Lemma 21.** *Each application of R11 decreases the measure $p$ by at least $2\delta_3 - \delta_2$.*

*Proof.* At least a subgraph $G^*$ with total cost at least $2\delta_3 - \delta_2$ is deleted from the graph. So the measure decreases by at least $2\delta_3 - \delta_2$.

**Lemma 22.** *Each application of R12 will not increase the measure $p$.*

*Proof.* In an application of this rule, a subgraph $G^*$ will be replaced by three degree-2 vertices and the degree of the two vertices $u$ and $u'$ in the cut can increase by at most 1. So R12 decreases the measure by at least $\sum_{v \in G^*} \delta_{d_G(v)} - 2\delta_3 - \delta_2 \geq 0$.

Although an application of R12 may not decrease the measure directly, it will create a 5-cycle with exactly two vertices of degree $\geq 3$, which can be further reduced by applying other reduction rules. By putting all these together, we still can strictly decrease the measure.

**Lemma 23.** *Assume that R1 to R4 can not be applied. If there is a chain containing at least three degree-2 vertices, then we can apply reduction rules to decrease the measure $p$ by at least $2\delta_2$.*

*Proof.* Let $v_0v_1v_2v_3v_4$ be the chain such that $d(v_i) = 2$ for $i = 1, 2, 3$. If R5 can not be applied, then we know that for $i \in 1, 2, 3$, it holds that $w(v_i) \leq \max\{w(v_{i-1}), w(v_{i+1})\}$. So, either $w(v_0) \geq w(v_1) \geq w(v_2) \geq w(v_3)$ or $w(v_4) \geq w(v_3) \geq w(v_2) \geq w(v_1)$ or $w(v_0) \geq w(v_1) \geq w(v_2) \leq w(v_3) \leq w(v_4)$. Hence, one of R5 to R8 can be applied. By Lemma 18, we know that the measure will decrease by at least $2\delta_2$.

**Lemma 24.** *Let $G$ be a graph where R1 to R4 can not be applied. Let $G'$ be the graph after an application of one of R5 to R8. The minimum degree of $G'$ is at least 2.*

*Proof.* By Lemma 14, we know that the minimum degree of $G$ is at least 2.

**Case 1:** R5 is applied. We will delete a degree-2 vertex $v$ and its two neighbors $u_1$ and $u_2$, and introduce a new vertex $v'$ adjacent to all vertices in $N(N(v))$. If a degree-1 vertex $u$ is created, then the degree-1 vertex $u$ will be a vertex in $N(N(v))$. Thus, in $G$, $u$ is a degree-2 vertex, and $u$ and $v$ will form a twin. However, we have assumed that R2 can not be applied on $G$. So $G'$ can not contain a vertex of degree 1.

**Case 2:** one of R6 and R8 is applied. Here, we will reduce a chain to a smaller one and do not decrease the degree of the two endpoints of the chain. Hence, the minimum degree of $G'$ is still at least 2.

**Case 3:** R7 is applied. Let $v_1v_2v_3v_4$ be the cycle that R7 is applied to, as shown in Fig. 7. By Lemma 15, we know that $\min\{d(v_1), d(v_4)\} \geq 3$. Since we will delete $v_2$ and $v_3$ to obtain $G'$, the operation will only decrease the degree of $v_1$ and $v_2$ exactly 1 respectively. So, it holds that the minimum degree of $G'$ is at least 2.

**Lemma 25.** *Let $G$ be a graph where R1 to R4 can not be applied. If there is a 4-cycle $C$ containing exactly two degree-2 vertices, then the measure $p$ can be decreased by at least $5\delta_2$ by applying reduction rules.*

*Proof.* Let $C = v_1v_2v_3v_4$ be a 4-cycle in the graph. By Lemma 15, we know that the two degree-2 vertices must be adjacent. We assume that $v_2$ and $v_3$ are degree-2 vertices. If R5 and R6 can not be applied on the graph, then R7 must be applied. We consider the three cases. Let $G'$ be the resulting graph after executing one of R5 to R7. By Lemma 24, we know that the minimum degree of $G'$ is at least 2.

**Case 1:** R5 is applied on a degree-2 vertex $v$. If $v \in \{v_2, v_3\}$, then the structure of $G'$ is isomorphic to the graph obtained from $G$ by deleting $\{v_2, v_3\}$. So we have that $p(G) - p(G') \geq 2\delta_3 \geq 5\delta_2$. Otherwise, after applying R5, the 4-cycle $C$ is still left in $G'$. For this case, we have that $p(G) - p(G') \geq 2\delta_2$ by Lemma 18. We can further apply reduction rules on the graph since the 4-cycle $C$ still exists. If R1 to R4 can be applied to $G'$, then by Lemma 12 and Lemma 13, we can decrease the measure $p$ by at least $3\delta_2$ more. Otherwise, by induction, we know that the measure $p$ will eventually decrease by at least $5\delta_2$.

**Case 2:** R6 is applied on a path $v'_1v'_2v'_3v'_4$, where $v'_2$ and $v'_3$ are degree-2 vertices. The two degree-2 vertices $v'_2$ and $v'_3$ will be replaced with an edge $v'_1v'_4$ if it does not exist. If edge $v'_1v'_4$ exists, then the reduction rule simply deletes $v'_2$ and $v'_3$ from the graph and the $p$ will decrease by at least $2\delta_2+2\delta_3^{<-1>} = 2\delta_3 \geq 5\delta_2$. If edge $v'_1v'_4$ does not exist, then $v'_1v'_2v'_3v'_4$ is different from $v_1v_2v_3v_4$ and the 4-cycle $C$ is still left in $G'$. By induction, we know that the measure $p$ will eventually decrease by at least $5\delta_2$.

**Case 3:** R7 is applied on a path $v'_1v'_2v'_3v'_4$, where $v'_2$ and $v'_3$ are degree-2 vertices. We will remove two degree-2 vertices $\{v'_2, v'_3\}$ from $G$ to obtain $G'$. So, the measure $p$ will decrease by at least $2\delta_2 + 2\delta_3^{<-1>} = 2\delta_3 \geq 5\delta_2$.

**Lemma 26.** *Let $G$ be a graph where R1 to R4 can not be applied. If there is a cycle $C$ with $|C| \geq 5$ containing at most two vertices of degree at least 3, then applying reduction rules can decrease the measure $p$ by at least $2\delta_3 - \delta_2$.*

*Proof.* By lemma 14, we know that the minimum degree of $G$ is 2. Let $G'$ be the resulting graph after executing one of R5 to R10 on $G$ and let $G^*$ be the resulting graph after executing all the reduction rules on $G'$. We consider how many vertices of degree $\geq 3$ in the cycle $C$.

**Case 1:** $C$ contains no vertex of degree $\geq 3$. By lemma 16, we know that whole component $C$ will be reduced. So, $p(G) - p(G^*) \geq 5\delta_2 \geq 2\delta_3 - \delta_2$.

**Case 2:** $C$ contains exactly one vertex of degree $\geq 3$. Since $|C| \geq 5$, there is a chain that contains at least 3 vertices of degree 2. By Lemma 23, we know that one of R5 to R8 can be applied to some vertices of $C$ to obtained $G'$ and $p(G)-p(G') \geq 2\delta_2$. Since R5, R6 and R8 will reduce a chain to a smaller one with decreasing the length at most 2 and R7 will remove 2 adjacent vertices of degree 2 that are in a 4-cycle, there still exists a cycle $C'$ with at most 2 vertices of degree $\geq 3$ in $G'$ and $G'$ has a minimum degree at least 2. If R1 to R4 can be applied to $G'$, by Lemma 12 and Lemma 13, we know that $p(G')-p(G^*) \geq 3\delta_2$. Otherwise, by Lemma 15, we know that $|C'| \geq 4$. Now, we consider the cases of $|C'|$. If $|C'| = 4$, then by Lemma 25, we know that $p(G') - p(G^*) \geq 5\delta_2$. Otherwise, $|C'| \geq 5$. For this case, by induction, we know that $p(G') - p(G^*) \geq 2\delta_3 - \delta_2$.

**Case 3:** $C$ contains exactly two vertices of degree $\geq 3$. Clearly, one of R5 to R10 can be applied to $G$.

When one of R5 to R8 is applied to $G$, the discussion is similar to the above case.

When one of R9 to R10(1) is applied to $G$, by Lemma 19 and Lemma 20, we know that $p(G) - p(G') \geq 2\delta_3 - \delta_2$.

When R10(2) is applied to $G$ to obtain $G'$, by R10(2) and Lemma 17, we know that there is a 5-cycle that contains at most two vertices of degree $\geq 3$ and there is no cycle containing at most one vertex of degree $\geq 3$. If R1 to R4 can be applied to $G'$, then by Lemma 12 and Lemma 13, we know that $p(G') - p(G^*) \geq 2\delta_3 - \delta_2$. Otherwise, by induction, we can get $p(G') - p(G^*) \geq 2\delta_3 - \delta_2$.

Since $5\delta_2 \geq 2\delta_3 - \delta_2$, we know that this lemma holds.

**Definition 1.** *An instance is called* reduced*, if none of our reduction rules can be applied.*

**Lemma 27.** *In a reduced instance, any two degree-2 vertices in different chains have at most one common chain-neighbor of degree at least 3, and each cycle contains at least three vertices of degree $\geq 3$.*

*Proof.* Assume there are two degree-2 vertices in different chains that have two common chain-neighbors of degree at least 3. Then there is a cycle $C$ contains exactly two vertices of degree $\geq 3$. By Lemma 15 and Lemma 26, we know that there is no such cycle in a reduced instance.

**Lemma 28.** *For a triangle $C$ in a reduced instance, each vertex in $C$ is a vertex of degree at least 3 and it has a chain-neighbor of degree at least* 3 *not in $C$.*

*Proof.* By Lemma 15, we know that all the vertices in triangles are of degree at least 3. If a vertex in $C$ does not have a chain-neighbor of degree at least 3 in $C$, then there is a cycle containing at most two vertices of degree $\geq 3$, a contradiction to Lemma 27.

## 4 Branching Rules

Next, we introduce our branching rules, which will only be applied to reduced instances. After applying a branching rule, the algorithm will apply reduction rules as more as possible on each sub instance. In our analysis, we will consider the following applications of reduction rules together.

### 4.1 Two Branching rules

We use two branching rules. The first branching rule is to branch on a vertex $v$ by considering two cases: (i) there is a maximum weighted independent set in

$G$ which does not contain $v$; (ii) every maximum weighted independent set in $G$ contains $v$. For the former case, we simply delete $v$ from the graph. For the latter case, by Lemma 2 we know that we can include the set $S_v$ confining $v$ in the independent set. So we delete $N[S_v]$ from the graph.

**Branching Rule 1 (Branching on a vertex)** *Branch on a vertex $v$ to generate two sub instances by either deleting $v$ from the graph or deleting $N[S_v]$ from the graph and adding $w(S_v)$ to $M_c$.*

The following property of 4-cycles has been used to design an effective branching rule for unweighted versions [32], which also holds in weighted graphs.

**Lemma 29.** *Let $v_1v_2v_3v_4$ be a cycle of length 4 in the graph $G$. Then for any independent set $S$ in $G$, either $v_1, v_3 \notin S$ or $v_2, v_4 \notin S$.*

*Proof.* Since any independent set contains at most two non-adjacent vertices in a 4-cycle, we know that this lemma holds.

Based on Lemma 29, we get the second branching rule.

**Branching Rule 2 (Branching on a 4-cycle)** *Branch on a 4-cycle $v_1v_2v_3v_4$ to generate two sub instances by deleting either $\{v_1, v_3\}$ or $\{v_2, v_4\}$ from the graph.*

## 4.2 The analysis and some properties

The hardest part is to analyze how much we can decrease the measure in each sub-branch of a branching operation. Usually, we need to deeply analyze the local graph structure and use case-analysis. Here we try to summarize some common properties. The following notations will be frequently used in the whole paper.

Let $S$ be a vertex subset in a reduced graph $G$. We use $G_{-S}$ to denote the graph after deleting $S$ from $G$ and iteratively applying R1 to R4 until none of them can be applied. We use $R_S$ to denote the set of deleted vertices during applying R1 to R4 on $G - S$. Then $G_{-S} = G - (S \cup R_S)$. We also use $e_S$ to denote the number of edges between $S \cup R_S$ and $V \setminus (S \cup R_S)$ in $G$. We have the following lemmas for some bounds on $p(G) - p(G_{-S})$. Note that $G_{-S}$ may not be a reduced graph because of reduction rules from R5 to R12 and we may further apply reduction rules to further decrease the measure $p$.

**Lemma 30.** *It holds that*

$$p(G) - p(G_{-S}) = \sum_{u \in S \cup R_S} \delta_{d_G(u)} + e_S \delta_3^{<-1>}. \tag{6}$$

*Proof.* Since $S \cup R_S$ will be removed from $G$, we can reduce the measure by at least $\sum_{u \in S \cup R_S} \delta_{d_G(u)}$ from them. For each vertex in $N(S \cup R_S)$, it must be a vertex of degree $\geq 2$ in $G_{-S}$, otherwise R1 to R4 can be further applied on $G_{-S}$. Thus, deleting each edge between $S \cup R_S$ and $V \setminus (S \cup R_S)$ will decrease the measure $p$ by $\delta_3^{<-1>}$ from $N(S \cup R_S)$. In total, we can decrease the measure $p$ by $e_S \delta_3^{<-1>}$ from $N(S \cup R_S)$.

In some cases, we can not use the bound in (6) directly, since we may not know the vertex set $R_S$. So we also consider some special cases and relaxed bounds.

**Lemma 31.** *Let $S = \{v\}$ be a set of a single vertex of degree $\geq 3$. We have that*

$$p(G) - p(G_{-S}) \geq \delta_{d(v)} + \sum_{u \in N(v)} \delta_{d(u)}^{<-1>} + q_2 \delta_3^{<-1>},$$

*where $q_2$ is the number of degree-2 vertices in $N(v)$.*

*Proof.* After removing $v$, we can reduce the measure $p$ by $\delta_{d(v)}$ from the vertex $v$ itself and $\sum_{u \in N(v)} \delta_{d(u)}^{<-1>}$ from $N(v)$ since the degree of each vertex in $N(v)$ will decrease by 1. Furthermore, each degree-2 vertex $w \in N(v)$ in $G$ will become a degree-1 vertex in $G - v$. By Lemma 27, we know that $w$ has two different chain-neighbors of degree $\geq 3$. Let they be $v$ and $v'$. By Lemma 15, Lemma 17, Lemma 25 and Lemma 26, we know that there is no cycle that contains at most two vertices of degree $\geq 3$. So, we know that $v' \notin N(v)$. Furthermore, it holds that $d_{G-v}(v') = d_G(v')$. After iteratively applying R1 to R4 in $G - v$ to reduce degree-1 vertices, either $v'$ will be deleted or the degree of $v'$ will decrease by 1. Thus, we can further reduce $p$ by at least $\min\{\delta_{d(v')}, \delta_{d(v')}^{<-1>}\} = \delta_{d(v')}^{<-1>} = \delta_3^{<-1>}$. By Lemma 27, we know that for each degree-2 vertex $w \in N(v)$ in $G$, the other chain-neighbor $v'$ of degree $\geq 3$ is different. Thus, we can further reduce $p$ by at least $q_2 \delta_3^{<-1>}$ from these vertices.

**Lemma 32.** *If $S \cup R_S$ contains $N[v]$ for some vertex $v$ of degree $\geq 3$, then we have that*

$$p(G) - p(G_{-S}) \geq \sum_{u \in N[v]} \delta_{d(u)} + q_2 \delta_3^{<-1>},$$

*where $q_2$ is the number of degree-2 vertices in $N(v)$.*

*Proof.* The proof is similar to the proof of Lemma 31. At least $N[v] \subseteq S$ is removed and we can reduce the measure $p$ by $\sum_{u \in N[v]} \delta_{d(u)}$ from $N[v]$ directly. Each degree-2 vertex $w \in N(v)$ in $G$ has a different chain-neighbor $v' \neq v$ of degree $\geq 3$ by Lemma 27, which will be deleted or the degree of $v'$ will decrease by at least 1 after iteratively applying R1 to R4 to reduce degree-1 vertices. Thus, we can further reduce $p$ by at least $q_2 \delta_3^{<-1>}$.

Recall that we use $\mathcal{C}(G')$ to denote the set of connected components of the graph $G'$. We can easily obverse the following lemma, which will be used to prove several bounds on $p(G) - p(G_{-S})$.

**Lemma 33.** *Let $S$ be a vertex subset. Let $S'$ be a subset of $S \cup R_S$ and $R' = S \cup R_S \setminus S'$. The number of edges between $S \cup R_S$ and $V \setminus (S \cup R_S)$ is $e_S$, and the number of edges between $S'$ and $V \setminus S'$ is $k$. For any component $H \in \mathcal{C}(G[R'])$,* *the number of edges between $S'$ and $H$ is $l_H$ and the number of edges between $H$ and $N(S \cup R_S)$ is $r_H$. We have that*

$$k - e_S = \sum_{H \in \mathcal{C}(G[R'])} (l_H - r_H).$$

*Furthermore, for any component $H \in \mathcal{C}(G[R'])$ containing only degree-2 vertices, it holds that*

$$l_H - r_H = 0 \quad or \quad 2.$$

**Lemma 34.** *For any subset $S' \subseteq S \cup R_S$ with $k$ edges between $S'$ and $V \setminus S'$, it holds that*

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S \delta_3^{<-1>} + \begin{cases} 0, & k - e_S \leq 0 \\ \delta_3, & k - e_S = 1 \\ \delta_2, & k - e_S = 2 \\ \delta_3, & k - e_S = 3 \\ 2\delta_2, & k - e_S > 3. \end{cases}$$

*Proof.* By Inequality (6), we know that this lemma holds for $k - e_S \leq 0$. Next, we consider the cases where $k - e_S \geq 1$. Let $R' = S \cup R_S \setminus S'$. If $k - e_S = 1$, then by Lemma 33, we know that there is a component $H \in \mathcal{C}(G[R'])$ containing at least one vertex of degree $\geq 3$. If $k - e_S = 2$, then $R'$ is nonempty and contains at least one vertex of degree $\geq 2$. If $k - e_S = 3$, then $R'$ contains at least one vertex of degree $\geq 3$ similar to the case of $k - e_S = 1$. If $k - e_S > 3$, then $R'$ contains at least one vertex of degree $\geq 3$ or at least two vertices of degree 2. So, by Inequality (6), this lemma holds.

**Corollary 1.** *For any subset $S' \subseteq S \cup R_S$ with $k$ edges between $S'$ and $V \setminus S'$, it holds that either $p(G) - p(G_{-S}) > 10$ or*

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + \begin{cases} k\delta_3^{<-1>}, & \text{if } k \leq 4 \\ \delta_2 + 3\delta_3^{<-1>}, & \text{if } k = 5 \\ \delta_3 + 3\delta_3^{<-1>}, & \text{if } k = 6 \\ 2\delta_2 + 3\delta_3^{<-1>}, & \text{if } k > 6. \end{cases}$$

*Proof.* The instance $G$ is reduced. So R11 and R12 can not be applied. If $\sum_{v \in S \cup R_S} \delta_{d_G(v)} \leq 10$ and there are at most two edges between $S \cup R_S$ and $V \setminus (S \cup R_S)$, then the condition in R11 or R12 would hold. So we have either $p(G) - p(G_{-S}) > 10$ or $e_S \geq 3$. For the latter case, by Lemma 34, this corollary also holds.

**Lemma 35.** *Assume that a reduced graph $G$ has a maximum degree 3 and has no 3 or 4-cycles. For any subset $S' \subseteq S \cup R_S$ with $k$ edges between $S'$ and $V \setminus S'$, if the diameter of the induced graph $G[S']$ is 2, then it holds that either $p(G) - p(G_{-S}) > 10$ or*

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + 3\delta_3^{<-1>} + \begin{cases} 0, & k \leq 3 \\ \delta_3^{<-1>}, & k = 4 \\ 2\delta_2, & k = 5 \\ \delta_2 + \delta_3, & k = 6. \end{cases}$$

*Proof.* Let $R' = (S \cup R_S) \setminus S'$. Note that any vertex $v \in R'$ is adjacent to at most one vertex in $S'$, otherwise $v$ and some vertices in $S'$ would form a cycle of length at most 4 since the diameter of $G[S']$ is 2, a contradiction to the assumption that there is no cycle of length at most 4. Thus, any component $H \in \mathcal{C}(G[R'])$ contains at least $l_H$ vertices, and $R'$ contains at least $k - e_S$ vertices.

We first consider the value of $k - e_S$.

If $k - e_S = 1$, then by Lemma 33, we know that there is a vertex of degree $\geq 3$ in $R'$. So, by Inequality (6), we know that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S \delta_3^{<-1>} + \delta_3. \tag{7}$$

If $k - e_S = 2$, then $R'$ contains at least $k \geq 2$ vertices. So, by Inequality (6), we know that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S \delta_3^{<-1>} + 2\delta_2. \tag{8}$$

If $k - e_S = 3$, then $R'$ contains at least $k \geq 3$ vertices. By Lemma 33, we know that at least one vertex in $R'$ is of degree $\geq 3$. Hence, by Inequality (6), we know that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S \delta_3^{<-1>} + 2\delta_2 + \delta_3. \tag{9}$$

Next, we prove the lemma by considering the value of $k$.

**Case 1**: $k \leq 3$. This lemma holds by Corollary 1.

**Case 2**: $k = 4$. If $k - e_S = 0$, by Lemma 34, we get that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + 4\delta_3^{<-1>}.$$

If $k - e_S = 1, 2$ and 3, then we will get (7), (8), and (9), respectively. It is impossible that $e_S = 0$ and then it is impossible that $k - e_S = 4$. The worst case is that $k - e_S = 0$.

**Case 2**: $k = 5$. Similar to Case 2, we consider all possible values of $k - e_S$. The worst case is that $k - e_S = 2$, where by (8) we get that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + 3\delta_3^{<-1>} + 2\delta_2.$$

**Case 3**: $k = 6$. Similar to Case 2, we consider all possible values of $k - e_S$. The worst case is that $k - e_S = 2$, where by (9) we get that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + 4\delta_3^{<-1>} + 2\delta_2.$$

**Lemma 36.** *Assume that a reduced graph $G$ has a maximum degree 3, and each cycle $C$ in it contains at least five vertices, where at least four vertices are degree-3 vertices. For any subset $S' \subseteq S \cup R_S$ with $k$ edges between $S'$ and $V \setminus S'$, if each path $P$ in the induced graph $G[S']$ contains either at most three vertices or at most two degree-3 vertices, then it holds that either $p(G) - p(G_{-S}) > 10$ or*

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + \begin{cases} k\delta_3^{<-1>}, & k \leq 5 \\ \delta_3 + 2\delta_2 + 3\delta_3^{<-1>}, & k = 6. \end{cases}$$

*Proof.* The proof is similar to the proof of Lemma 35. Let $R' = S \cup R_S \setminus S'$. Any vertex $v \in R'$ is adjacent to at most one vertex in $S'$, otherwise $v$ would be in a 4-cycle or a cycle containing at most three degree-3 vertices since each path $P$ in $G[S']$ either contains at most three vertices or contains at most two degree-3 vertices. However, these cycles would not appear by the assumption. Thus, $R'$ contains at least $k - e_S$ vertices.

We first consider the value of $k - e_S$.

If $k - e_S = 1$, then by Lemma 33, we know that there is a vertex of degree $\geq 3$ in $R'$. So, by Inequality (6), we know that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S\delta_3^{<-1>} + \delta_3. \tag{10}$$

If $k - e_S = 2$, then $R'$ will contain at least two vertices of degree 3. The reason is below. If $R'$ contains at most one degree 3, then there will be a cycle containing at most three degree-3 vertices, a contradiction to the assumption. So, by Inequality (6), we know that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S\delta_3^{<-1>} + 2\delta_3. \tag{11}$$

If $k - e_S = 3$, then $R'$ contains at least $k \geq 3$ vertices. By Lemma 33, we know that at least one vertex in $R'$ is of degree $\geq 3$. Hence, by Inequality (6), we know that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + e_S\delta_3^{<-1>} + 2\delta_2 + \delta_3. \tag{12}$$

Next, we prove the lemma by considering the value of $k$.

**Case 1**: $k \leq 4$. This lemma holds by Corollary 1.

**Case 2**: $k = 5$. If $k - e_S = 0$, by Lemma 34, we get that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + 5\delta_3^{<-1>}.$$

If $k - e_S = 1, 2$ and 3, then we will get (10), (11), and (12), respectively. It is impossible that $e_S \leq 1$ since $G$ has no cut of size at most 2, and then it is impossible that $k - e_S \geq 4$. The worst case is that $k - e_S = 0$.

**Case 3**: $k = 6$. Similar to Case 2, we consider all possible values of $k - e_S$. The worst case is that $k - e_S = 3$, where by (8) we get that

$$p(G) - p(G_{-S}) \geq \sum_{u \in S'} \delta_{d_G(u)} + 3\delta_3^{<-1>} + 2\delta_2 + \delta_3.$$

## 5 The Algorithm

Now we are ready to describe the whole algorithm. The algorithm will first apply reduction rules and also solve connected components of size bounded by a constant (or the measure is bounded by a constant) directly.

Second, the algorithm will branch on vertices of degree $\geq 5$ if any. Note that even the input graph has no high-degree vertices, some reduction rules may create them during the algorithm. Third, the algorithm will deal with 4-cycles and degree-4 vertices. Last is to deal with degree-3 vertices, which is the most complicated part of the algorithm. When the algorithm executes one step, we assume that all previous steps can not be applied now.

**Step 1 (Applying Reductions)** *If the instance is not reduced, iteratively apply reduction rules in order, i.e., when one reduction rule is applied, no reduction rule with a smaller index can be applied on the graph.*

**Step 2 (Solving Small Components)** *If there is a connected component $G^*$ of $G$ such that $p(G^*) \leq 10$, solve the component $G^*$ directly and return $\alpha(G - G^*) + \alpha(G^*)$.*

Note that $p(G^*) \leq 10$, the number of vertices with degree $\geq 3$ is at most 10. We can enumerate all subsets of vertices with degree $\geq 3$ and let them in the independent set, and the remaining graph has a maximum degree at most 2, which can be solved in polynomial time by Lemma 16. So this step can be solved in polynomial time.

**Step 3 (Branching on Vertices of Degree $\geq 5$)** *If there is a vertex $v$ with degree $d(v) \geq 5$, then branch on $v$ with Branching Rule 1 by either excluding $v$ from the independent set or including $S_v$ in the independent set.*

**Lemma 37.** *Step 3 followed by applications of reduction rules creates a branching vector covered by*

$$[\delta_{d(v)} + d(v)\delta_3^{<-1>}, \delta_{d(v)} + d(v)\delta_3]. \tag{13}$$

*Proof.* For the first branching, a single vertex $v$ is deleted from the graph. By Lemma 31, we get $\delta_{d(v)} + d(v)\delta_3^{<-1>} + q_2\delta_2 \geq \delta_{d(v)} + d(v)\delta_3^{<-1>}$, where $q_2 \geq 0$ is the number of degree-2 vertices in $N(v)$.

For the second branching, $N[v] \subseteq N[S_v]$ is deleted from the graph. By Lemma 32, we know that the measure will be decreased by at least $\sum_{u \in N[v]} \delta_{d(u)} + q_2\delta_3^{<-1>}$, which is at least $\delta_{d(v)} + d(v)\delta_3$.

For the worst case that $d(v) = 5$, the branching vector (13) will become

$$[\delta_5 + 5\delta_3^{<-1>}, \delta_5 + 5\delta_3] = [5.368, 7.248].$$

Next, we assume that the maximum degree of the graph is at most 4.

**Step 4 (Branching on 4-Cycles with Chords)** *If there is a 4-cycle $C = v_1v_2v_3v_4$ with a chord $v_1v_3 \in E$, then branch on the 4-cycle with Branching Rule 2 by excluding either $\{v_1, v_3\}$ or $\{v_2, v_4\}$ from the independent set.*

Note that it is impossible that both of $v_1$ and $v_3$ are of degree 3, since otherwise one of $v_1$ and $v_3$ is an unconfined vertex and R1 should be applied. Then one of $v_1$ and $v_3$ is of degree 4. Without loss of generality, we assume that $v_1$ is a degree-4 vertex. Since $v_1$ and $v_3$ are adjacent, we know that none of $v_2$ and $v_4$ can be a degree-2 vertex, otherwise R1 to R4 can be applied.

**Lemma 38.** *Step 4 followed by applications of reduction rules creates a branching vector covered by one of*

$$[3\delta_4 + \delta_3^{<-1>}, 4\delta_4 + 2\delta_3^{<-1>}] = [5.496, 7.744] \quad \textit{and}$$

$$[4\delta_4, 2\delta_4 + 2\delta_3 + 2\delta_2] = [6.496, 6].$$

*Proof.* We use $\Delta_1$ and $\Delta_2$ to denote the amount of measure decreased in the two sub branches. We analyze $\Delta_1$ and $\Delta_2$ by considering several different cases.

**Case 1**: $v_2v_4 \in E$. Now $\{v_1, v_2, v_3, v_4\}$ from a clique. None of them can be a degree-3 vertex in $G$, otherwise the vertex would have a clique neighborhood and R4 should be applied. Thus, all of the four vertices in $C$ are degree-4 vertices. For each vertex in the cycle $C$, a neighbor not in $C$ is called an *out-neighbor*. We can see that the out-neighbor of $v_1$ is different from the out-neighbor of $v_3$ (resp., the out-neighbor of $v_2$ is different from the out-neighbor of $v_4$), otherwise one of $v_1$ and $v_3$ (resp., one of $v_2$ and $v_4$) would be unconfined.

In the first branching, $\{v_1, v_3\}$ is deleted from the graph. We can reduce the measure $p$ by $2\delta_4$ from $\{v_1, v_3\}$, $2\delta_4^{<-2>}$ from $\{v_2, v_4\}$, and at least $2\min\{\delta_2^{<-1>}, \delta_3^{<-1>}, \delta_4^{<-1>}\} = 2\delta_2$ from the out-neighbors of $\{v_1, v_3\}$. In total, it is at least $2\delta_4 + 2\delta_4^{<-2>} + 2\delta_2 = 4\delta_4$. In the second branching, the measure can be reduced by the same amount. We get a branching vector:

$$[4\delta_4, 4\delta_4].$$

Next, we assume that $v_2v_4 \notin E$.

**Case 2**: $d(v_3) = 3$ and at least one of $v_2$ and $v_4$, say $v_2$ is a degree-3 vertex. In the first branching, after deleting $\{v_1, v_3\}$, vertex $v_2$ will become a degree-1 vertex and will be removed by applying R1 to R4. Thus, by Corollary 1 (setting $S' = \{v_1, v_2, v_3\}$ with $k = 4$), we know that the measure $p$ decreases by at least $\sum_{u\in\{v_1,v_2,v_3\}} \delta_{d_u} + 4\delta_3^{<-1>} = 2\delta_3 + \delta_4 + 4\delta_3^{<-1>} = 4\delta_4 - \delta_2$.

In the second branching, all the four vertices in the cycle $C$ will be deleted after applying R1 to R4. By Corollary 1 (setting $S' = \{v_1, v_2, v_3, v_4\}$ with $k \geq 3$), we know that the measure $p$ decreases by at least $\sum_{u\in\{v_1,v_2,v_3,v_4\}} \delta_{d_u} + 3\delta_3^{<-1>} = 3\delta_3 + \delta_4 + 3\delta_3^{<-1>} = 4\delta_4$. We get a branching vector

$$[4\delta_4 - \delta_2, 4\delta_4].$$

**Case 3**: $d(v_3) = 3$ and $d(v_2) = d(v_4) = 4$. In the first branching of deleting $\{v_1, v_3\}$, we can reduce the measure $p$ by $\delta_4 + \delta_3$ from $\{v_1, v_3\}$, $2\delta_4^{<-2>}$ from $\{v_2, v_4\}$, and at least $\delta_2$ from the fourth neighbor of $v_1$. In total, we get $3\delta_4 + \delta_3^{<-1>}$.

The second branching is similar to the second branching in Case 2. all the four vertices in the cycle $C$ will be deleted in this branching. By Corollary 1 (setting $S' = \{v_1, v_2, v_3, v_4\}$ with $k \geq 3$), we know that the measure $p$ decreases by at least $\sum_{u\in\{v_1,v_2,v_3,v_4\}} \delta_{d_u} + 3\delta_3^{<-1>} = 4\delta_4 + 2\delta_3^{<-1>}$. We get a branching vector

$$[3\delta_4 + \delta_3^{<-1>}, 4\delta_4 + 2\delta_3^{<-1>}].$$

**Case 4**: $d(v_3) = 4$. We show that in the first branching of deleting $\{v_1, v_3\}$, the measure $p$ will decrease by at least $4\delta_4$. If both of $v_2$ and $v_4$ are degree-4 vertices, we reduce the measure $p$ by $2\delta_4$ from $\{v_1, v_3\}$, $2\delta_4^{<-2>}$ from $\{v_2, v_4\}$, and at least $2\delta_2$ from the two different out-neighbors of $\{v_1, v_3\}$. In total, it is $2\delta_4 + 2\delta_4^{<-2>} + 2\delta_2 = 4\delta_4$. Else at least one of $v_2$ and $v_4$, say $v_2$ is a degree-3 vertex. After deleting $\{v_1, v_3\}$, vertex $v_2$ will become a degree-1 vertex and will be removed by applying R1 to R4. Thus, by Corollary 1 (setting $S' = \{v_1, v_2, v_3\}$ with $k = 5$), we know that the measure $p$ decreases by at least $\sum_{u\in\{v_1,v_2,v_3\}} \delta_{d_u} + \delta_3 + 3\delta_3^{<-1>} = 4\delta_4 + \delta_3^{<-1>} > 4\delta_4$.

In the second branching of deleting $\{v_2, v_4\}$, we reduce the measure $p$ by at least $2\delta_3$ from $\{v_2, v_4\}$, $2\delta_4^{<-2>}$ from $\{v_1, v_3\}$, and at least $2\delta_2$ from two different out-neighbors of $\{v_2, v_4\}$. In total, it is $2\delta_4 + 2\delta_3 + 2\delta_2$. We get a branching vector

$$[4\delta_4, 2\delta_4 + 2\delta_3 + 2\delta_2].$$

Note that $2\delta_4 + 2\delta_3 + 2\delta_2 < 4\delta_4 - \delta_2 < 4\delta_4$. We know that all branching vectors will be covered by one of $[3\delta_4 + \delta_3^{<-1>}, 4\delta_4 + 2\delta_3^{<-1>}]$ and $[4\delta_4, 2\delta_4 + 2\delta_3 + 2\delta_2]$.

**Step 5 (Branching on Degree-4 Vertices)** *If there is a degree-4 vertex $v$, then branch on it with Branching Rule 1 by either excluding $v$ from the independent set or including $S_v$ in the independent set.*

We use $q_i$ to denote the number of degree-$i$ neighbors of $v$ in $G$. In the first branching of deleting $v$, by Lemma 31, we can reduce the measure $p$ by at least

$$\delta_4 + \sum_{i=2}^{4} q_i \delta_i^{<-1>} + q_2\delta_3^{<-1>} \geq \delta_4 + q_2\delta_3 + (1 - q_2)\delta_3^{<-1>}.$$

We consider the second branching of deleting $N[S_v]$. We can see that $N[v] \subseteq N[S_v]$ will be deleted. Note that each degree-2 vertex in $N(v)$ is adjacent to a vertex not in $N[v]$ otherwise $v$ would have a clique neighborhood and reduction rules can be applied. For the case that $q_2 = 4$, by Lemma 32, we know that the measure $p$ will decrease by at least $\delta_4 + 4\delta_2 + 4\delta_3^{<-1>} = \delta_4 + 4\delta_3$. For the case that $0 \le q_2 \le 3$, the number of edges between $N[v]$ and $V \setminus N[v]$ is at least 4 since otherwise there would be a vertex having a clique neighbor or a 4-cycle having a chord. By Corollary 1 (setting $S' = N[v]$ with $k \ge 4$), we know that the measure $p$ decreases by at least $\delta_4 + q_2\delta_2 + (4-q_2)\delta_3 + 3\delta_3^{<-1>} + \delta_2 = 4\delta_4 + (1-q_2)\delta_3 + (q_2+1)\delta_2$.

The above analysis gives the following lemma.

**Lemma 39.** *Step 5 followed by applications of reduction rules creates a branching vector covered by one of*

$$[\delta_4 + 4\delta_3, \delta_4 + 4\delta_3] = [5.624, 5.624] \quad (q_2 = 4),$$
$$[\delta_4 + 3\delta_3 + \delta_3^{<-1>}, 4\delta_4 - 2\delta_3 + 4\delta_2] = [5.248, 6] \quad (q_2 = 3),$$
$$[\delta_4 + 2\delta_3 + 2\delta_3^{<-1>}, 4\delta_4 - \delta_3 + 3\delta_2] = [4.872, 6.624] \quad (q_2 = 2),$$
$$[\delta_4 + 1\delta_3 + 3\delta_3^{<-1>}, 4\delta_4 + 2\delta_2] = [4.496, 7.248] \quad (q_2 = 1), \quad \textit{and}$$
$$[\delta_4 + 4\delta_3^{<-1>}, 4\delta_4 + \delta_3 + \delta_2] = [4.12, 7.872] \quad (q_2 = 0).$$

Next, we assume the maximum degree of the graph is 3.

**Step 6 (Branching on Other 4-Cycles)** *If there is a 4-cycle $C = v_1v_2v_3v_4$, then branch on the 4-cycle with Branching Rule 2 by excluding either $\{v_1, v_3\}$ or $\{v_2, v_4\}$ from the independent set.*

**Lemma 40.** *Step 6 followed by applications of reduction rules creates a branching vector covered by*

$$[6\delta_3 - 2\delta_2, 6\delta_3 - 2\delta_2] = [5.248, 5.248].$$

*Proof.* All the four vertices in the cycle $C$ are degree-3 vertices or degree-2 vertices since there are no vertices of degree $\ge 4$ now. We can see that there is at most one degree-2 vertex in the cycle $C$. If there are two nonadjacent degree-2 vertices in a 4-cycle, then there is a twin and R2 should be applied. If there are two adjacent degree-2 vertices in a 4-cycle, then either R5 or R7 can be applied. So there is at most one degree-2 vertex in the cycle $C$.

**Case 1**: one vertex in $C$, say $v_1$ is of degree 2 and all other vertices in $C$ are of degree 3.

In each branching, all the four vertices in the cycle $C$ will be deleted after applying R1 to R4. By Corollary 1 (setting $S' = \{v_1, v_2, v_3, v_4\}$ with $k = 3$), we know that the measure $p$ decreases by at least $\sum_{u \in \{v_1,v_2,v_3,v_4\}} \delta_{d_u} + 3\delta_3^{<-1>} = 3\delta_3 + \delta_2 + 3\delta_3^{<-1>} = 6\delta_3 - 2\delta_2$. We get a branching vector

$$[6\delta_3 - 2\delta_2, 6\delta_3 - 2\delta_2].$$

**Case 2**: all the four vertices in $C$ are degree-3 vertices. In each branching, all the four vertices in the cycle $C$ will be deleted after applying R1 to R4. By Corollary 1 (setting $S' = \{v_1, v_2, v_3, v_4\}$ with $k = 4$), we know that the measure $p$ decreases by at least $\sum_{u \in \{v_1,v_2,v_3,v_4\}} \delta_{d_u} + 4\delta_3^{<-1>} = 4\delta_3 + 4\delta_3^{<-1>} = 8\delta_3 - 4\delta_2$. Note that $8\delta_3 - 4\delta_2 > 6\delta_3 - 2\delta_2$. So it is covered by the above case.

From now on, we assume that the graph has a maximum degree 3 and there is no 4-cycle. Next, we will first consider triangles in the graph. For a triangle $C$ in the graph after Step 6, each vertex in $C$ is of degree 3 by Lemma 15 and is chain-adjacent to a degree-3 vertex not in $C$ by Lemma 28.

**Step 7 (Branching on Triangles)** *If there is a triangle $C = v_1v_2v_3$, where we assume without loss of generality that $w(v_1) \geq \max\{w(v_2), w(v_3)\}$ and $v_1$ is chain-adjacent to a degree-3 vertex $u \neq v_2, v_3$, then branch on $u$ with Branching Rule 1.*

**Lemma 41.** *Step 7 followed by applications of reduction rules creates a branching vector covered by one of*

$$[6\delta_3 - 3\delta_2, 7\delta_3 + \delta_2] = [4.872, 7.376] \quad \textit{and}$$

$$[6\delta_3 - 2\delta_2, 5\delta_3 + 2\delta_2] = [5.248, 5.752].$$

*Proof.* We analyze the branching vector by considering the length of the chain between $v_1$ and $u$.

**Case 1:** $v_1$ and $u$ are adjacent. In the first branching of excluding $u$ from the independent set, all vertices $C$ will be removed by applying R1 to R4 since $w(v_1) \geq \max\{w(v_2), w(v_3)\}$ and $v_2$ and $v_3$ will become unconfined vertices. By Corollary 1 (setting $S' = \{v_1, v_2, v_3, u\}$ with $k = 4$), we know that the measure $p$ decreases by at least $4\delta_3 + 4\delta_3^{<-1>}$.

In the second branching of including $u \in S_u$ in the independent set, we will delete $N[u]$ at least. By Corollary 1 (setting $S' = N[u]$ with $k = 6 - q_2$, where $q_2$ is the number of degree-2 neighbors of $u$), we know that the measure $p$ decreases by at least $6\delta_3 - 2\delta_2$. We get a branching vector

$$[8\delta_3 - 4\delta_2, 6\delta_3 - 2\delta_2].$$

**Case 2:** the chain between $v_1$ and $u$ is of length 2. We let $w$ be the degree-2 vertex in the chain. In the first branching of excluding $u$ from the independent set, $w$ will become a degree-1 vertex and then $w$ and $v_1$ will be deleted by applying R1 to R4. By Corollary 1 (setting $S' = \{v_1, w, u\}$ with $k = 4$), we know that the measure $p$ decreases by at least $2\delta_3 + \delta_2 + 4\delta_3^{<-1>} = 6\delta_3 - 3\delta_2$.

In the second branching of including $u \in S_u$ in the independent set, we will delete $N[u]$. Furthermore, $v_2$ and $v_3$ will become unconfined vertices after deleting $N[u]$ and all the three vertices in $C$ will be deleted by applying R1 to R4. Since there is no 4-cycle now, we know that $\{v_1, v_2, v_3\} \cap N[u] = \emptyset$. By Corollary 1 (setting $S' = \{v_1, v_2, v_3\} \cup N[u]$ with $k \geq 4$), we know that the measure $p$ decreases by at least $4\delta_3 + 3\delta_2 + (3\delta_3^{<-1>} + \delta_2) = 7\delta_3 + \delta_2$. We get a branching vector

$$[6\delta_3 - 3\delta_2, 7\delta_3 + \delta_2].$$

**Case 3:** the chain between $v_1$ and $u$ is of length 3. We let $w_1$ and $w_2$ be the two degree-2 vertices in the chain. In the first branching of excluding $u$ from the independent set, $w_1$ and $w_2$ will be removed by reducing degree-1 vertices, and

$v_1$ will also be removed since it will become a degree-2 vertex in a triangle. By Corollary 1 (setting $S' = \{v_1, w_1, w_2, u\}$ with $k = 4$), we know that the measure $p$ decreases by at least $2\delta_3 + 2\delta_2 + 4\delta_3^{<-1>} = 6\delta_3 - 2\delta_2$.

In the second branching of including $u \in S_u$ in the independent set, we will delete $N[u]$. Furthermore, $w_1$ and $v_1$ will also be removed. By Corollary 1 (setting $S' = \{v_1, w_1\} \cup N[u]$ with $k \geq 4$), we know that the measure $p$ decreases by at least $2\delta_3 + 4\delta_2 + (3\delta_3^{<-1>} + \delta_2) = 5\delta_3 + 2\delta_2$. We get a branching vector

$$[6\delta_3 - 2\delta_2, 5\delta_3 + 2\delta_2].$$

Note that in a reduced graph, the length of each chain is at most 3. So the above three cases cover all cases. Since $5\delta_3 + 2\delta_2 < 8\delta_3 - 4\delta_2$, we know that $[8\delta_3 - 4\delta_2, 6\delta_3 - 2\delta_2]$ is covered by $[6\delta_3 - 2\delta_2, 5\delta_3 + 2\delta_2]$.

Next, we assume that the maximum degree of the graph is at most 3 and all cycles in the graph have a length of at least 5. We still need to deal with some long cycles that contain exactly three degree-3 vertices.

**Step 8 (Branching on Cycles Containing Three Degree-3 Vertices)** *If there is a cycle $C$ containing exactly three degree-3 vertices $\{v_1, v_2, v_3\}$, where we assume without loss of generality that $v_1$ is chain-adjacent to a degree-3 vertex $u \neq v_2, v_3$, then branch on $u$ with Branching Rule 1.*

**Lemma 42.** *Step 8 followed by applications of reduction rules can create a branching vector covered by one of*

$$[6\delta_3 - 4\delta_2, 8\delta_3 - 2\delta_2] = [4.496, 7.248],$$

$$[6\delta_3 - 3\delta_2, 6\delta_3 - \delta_2] = [4.872, 5.624], \quad \textit{and}$$

$$[6\delta_3 - 2\delta_2, 6\delta_3 - 2\delta_2] = [5.248, 5.248].$$

*Proof.* Let $q_2$ denote the number of degree-2 vertices in $N(u)$. We first consider the branching of excluding $u$ from the independent set, where we delete $u$ from the graph. Let $S = \{u\}$. Recall that $R_S$ is the set of deleted vertices during applying R1-R4 on $G - S$. Let $G_{-S}$ be the graph obtained from $G$ by removing $S \cup R_S$ from $G$. We distinguish two cases by considering whether $v_1 \in R_S$ or not.

For the case that $v_1 \notin R_S$, we will have that $v_1 \in N(S \cup R_S)$ and $v_1$ is left as a degree-2 vertex in $G_{-S}$ since $v_1$ and $u$ are chain-adjacent. Furthermore, we can see that $S \cup R_S$ does not contain any vertex in the cycle $C$, otherwise all the vertices in $C$ (including $v_1$) should be included in $R_S$ by applying R1-R4. So the cycle $C$ is left in $G_{-S}$. First, by Lemma 31, we have that $p(G) - p(G_{-S}) \geq \delta_3 + q_2\delta_2 + 3\delta_3^{<-1>} = 4\delta_3 + (q_2 - 3)\delta_2$. Second, we consider the cycle $C$ in the remaining graph $G_{-S}$. Now the cycle $C$ contains at most two degree-3 vertices ($v_1$ becomes a degree-2 vertex). If it contains exactly two degree-3 vertices, then by Lemma 26, we can further reduce the measure by at least $2\delta_3 - \delta_2$ by further applying reduction rules on $G_{-S}$. If $C$ contains at most one degree-3 vertex, then all the vertices in the cycle will be reduced by further applying reduction rules. Thus, we can further reduce the measure by at least $\delta_3 + 4\delta_2 > 2\delta_3 - \delta_2$. In total, the measure $p$ will decrease by at least

$$6\delta_3 + (q_2 - 4)\delta_2.$$

For the case that $v_1 \in R_S$, we apply Corollary 1 by letting $S'$ be the set containing $u$, $v_1$, all degree-2 vertices in $N(u)$ and all vertices in the chain between $u$ and $v_1$, and $k = 4$. Now $S'$ contains exactly 2 degree-3 vertices and $q_2$ degree-2 vertices. We know that the measure $p$ decreases by at least

$$2\delta_3 + q_2\delta_2 + 4\delta_3^{<-1>} = 6\delta_3 + (q_2 - 4)\delta_2.$$

So, in the first branching, we can always decrease the measure $p$ by at least $6\delta_3 + (q_2 - 4)\delta_2$.

Next, we analyze the second branching of including $S_u$ in the independent set, where we will delete $N[u]$ at least. We let $S = N[S_u]$. We distinguish two cases by considering whether $v_1 \in S \cup R_S$ or not.

First, we consider the case that $v_1 \notin S \cup R_S$. Since $v_1 \notin S \cup R_S$ and $S = N[S_u]$, we know that $v_1$ is not adjacent to $u$, where $q_2 > 0$. As in the analysis for the first branching, we know that all vertices in the cycle $C$ are left in the graph $G_{-S}$, where $v_1$ will become a degree-2 vertex in $G_{-S}$. By Lemma 35, we know that $p(G) - p(G_{-S}) \geq (4 - q_2)\delta_3 + q_2\delta_2 + 3\delta_3^{<-1>} = 4\delta_3$. By further reducing the cycle $C$ in $G_{-S}$, the measure will further decrease by at least $2\delta_3 - \delta_2$. In total, the measure will decrease by at least

$$6\delta_3 - \delta_2.$$

Second, we consider the case that $v_1 \in S \cup R_S$. Let $S'$ be the set of vertices in $N[u]$ and vertices in the chain between $u$ and $v_1$.

If $u$ and $v_1$ are not adjacent, then $S'$ contains $5 - q_2$ degree-3 vertices and at least $q_2$ degree-2 vertices, and there are $k = 7 - q_2$ edges between $S'$ and $V \setminus S'$, where $1 \leq q_2 \leq 3$. By applying Corollary 1 with $S'$ and $k$, we know that the measure will decrease by at least

$$(5 - q_2)\delta_3 + q_2\delta_2 + 4\delta_3^{<-1>} \geq 6\delta_3 - \delta_2.$$

If $u$ and $v_1$ are adjacent, then $S'$ contains $4 - q_2$ degree-3 vertices and $q_2$ degree-2 vertices, and there are $k = 6 - q_2$ edges between $S'$ and $V \setminus S'$, where $0 \leq q_2 \leq 2$. Note that $S' = N[u]$ now. For this case, by applying Lemma 35 with $S'$ and $k$, we know that the measure will decrease by at least

$$(4 - q_2)\delta_3 + q_2\delta_2 + f(q_2),$$

where $f(q_2) = 4\delta_3^{<-1>}$ if $q_2 = 2$, $f(q_2) = 3\delta_3^{<-1>} + 2\delta_2$ if $q_2 = 1$, and $f(q_2) = 3\delta_3^{<-1>} + \delta_2 + \delta_3$ if $q_2 = 0$.

Thus, in this step, we can always branch with one of the following branching vectors

$$[6\delta_3 - 4\delta_2, 8\delta_3 - 2\delta_2] \quad (q_2 = 0),$$

$$[6\delta_3 - 3\delta_2, 6\delta_3 - \delta_2] \ \ (q_2 = 1),$$

$$[6\delta_3 - 2\delta_2, 6\delta_3 - 2\delta_2] \ \ (q_2 = 2), \ \ \text{and}$$

$$[6\delta_3 - \delta_2, 6\delta_3 - \delta_2] \ \ (q_2 = 3).$$

The last case is covered by the second case. The lemma holds.

After this step, we can see that the graph has a maximum degree 3 and a minimum degree 2. The length of any cycle in the graph is at least 5 and each cycle contains at least four degree-3 vertices. Next, we are going to eliminate degree-3 vertices in the graph according to the following order: first deal with degree-3 vertices with exactly two degree-2 neighbors (Step 9); then deal with the connected components containing both degree-3 vertices with three degree-2 neighbors and degree-3 vertices with at most one degree-2 neighbors (Step 10); last, all degree-3 vertices in each connected component are either having three degree-2 neighbors or having at most one degree-2 neighbor, and we deal with these two kinds of connected components separately (Step 11 and Step 12).

**Step 9 (Branching on Degree-3 Vertices with Two Degree-2 Neighbors)** *If there is degree-3 vertex $u$ having two degree-2 neighbors and one degree-3 neighbor $v$, then branch on $v$ with Branching Rule 1.*

**Lemma 43.** *Step 9 followed by applications of reduction rules creates a branching vector covered by one of*

$$[4\delta_3 - \delta_2, 8\delta_3 - \delta_2] = [3.624, 7.624],$$

$$[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496], \quad \textit{and}$$

$$[4\delta_3 + \delta_2, 6\delta_3] = [4.376, 6].$$

*Proof.* We use $q_2$ to denote the number of degree-2 vertices in $N(v)$, where $0 \leq q_2 \leq 2$.

In the first branching of excluding $v$ from the independent set, we let $S = \{v\}$. Recall that $R_S$ is the set of deleted vertices during applying R1 to R4 on $G - S$. We distinguish two cases by considering whether $u$ is in $R_S$ or not. If $u \in R_S$, we apply Corollary 1 with $S'$ being the set including $u$ and $v$ and all degree-2 neighbors of $u$ and $v$ (note that after deleting $u$ and $v$ all the degree-2 neighbors of them will become degree-1 vertices and will be deleted by applying R1 to R4). We know that the measure $p$ decreases by at least

$$2\delta_3 + (q_2 + 2)\delta_2 + 4\delta_3^{<-1>} = 6\delta_3 + (q_2 - 2)\delta_2.$$

Otherwise, $u$ is not in $R_S$ and then $u \in N(S \cup R_S)$. Let $G_{-S}$ be the graph obtained from $G$ by removing $S \cup R_S$ from $G$. For the worst case that $R_S = \emptyset$, by Lemma 31, we know that $p(G) - p(G_{-S}) \geq \delta_3 + q_2\delta_2 + 3\delta_3^{<-1>}$. Since $u$ is left in $G_{-S}$, we know that the two degree-2 neighbors of $u$ in $G$ are also left in $G_{-S}$. Thus, there is a chain containing at least three degree-2 vertices (including $u$)

in $G_{-S}$, by Lemma 23, we can further decrease the measure $p$ by at least $2\delta_2$ by applying reduction rules on $G_{-S}$. In total, the measure $p$ decreases by at least

$$\delta_3 + q_2\delta_2 + 3\delta_3^{<-1>} + 2\delta_2 = 4\delta_3 + (q_2 - 1)\delta_2.$$

Note that $6\delta_3 + (q_2 - 2)\delta_2 \geq 4\delta_3 + (q_2 - 1)\delta_2$. In this branching, we can always reduce the measure $p$ by at least $4\delta_3 + (q_2 - 1)\delta_2$.

In the second branching, $v$ is included in the independent set and at least $N[v]$ is deleted. First, we consider the case that $q_2 = 0$. We apply Lemma 36 with $S' = N[v]$ and $k = 6$. The measure $p$ decreases by at least $4\delta_3 + 3\delta_3^{<-1>} + \delta_3 + 2\delta_2 = 8\delta_3 - \delta_2$. For the case that $q_2 = 1$, we also apply Lemma 36 with $S' = N[v]$ and $k = 5$. The measure $p$ decreases by at least $3\delta_3 + \delta_2 + 5\delta_3^{<-1>} = 8\delta_3 - 4\delta_2$. For the case that $q_2 = 2$, we apply Corollary 1 with $S' = N[v] \cup N[u]$ and $k = 4$. Now $S'$ contains two degree-3 vertices and four degree-2 vertices. The measure $p$ decreases by at least $2\delta_3 + 4\delta_2 + 4\delta_3^{<-1>} = 6\delta_3$.

Therefore, we can get the three claimed branching vectors.

**Step 10 (Branching on Degree-3 Vertices of a Mixed Case)** *If a degree-3 vertex $u$ without degree-3 neighbors is chain-adjacent to a degree-3 vertex $v$ with exactly two degree-3 neighbors, then branch on $v$ with Branching Rule 1.*

**Lemma 44.** *Step 10 followed by applications of reduction rules creates a branching vector covered by*

$$[4\delta_3, 8\delta_3 - 2\delta_2] = [4, 7.248].$$

*Proof.* In the first branching of excluding $v$ from the independent set, we let $S = \{v\}$. Recall that $R_S$ is the set of deleted vertices during applying R1 to R4 on $G - S$. We distinguish two cases by considering whether $u$ is in $R_S$ or not. If $u \in R_S$, we apply Lemma 36 by letting $S'$ be the set of vertices $N[u]$ and all vertices in the chain from $u$ to $v$ (including $v$) and $k = 4$. Now $S'$ contains at most two degree-3 vertices and satisfies the conditions in Lemma 36. We know that the measure $p$ decreases by at least

$$2\delta_3 + 3\delta_2 + 4\delta_3^{<-1>} = 6\delta_3 - \delta_2.$$

Otherwise, $u$ is not in $R_S$ and then $u \in N(S \cup R_S)$. Let $G_{-S}$ be the graph obtained from $G$ by removing $S \cup R_S$ from $G$. For the worst case that $R_S = \emptyset$, by Lemma 31, we know that $p(G) - p(G_{-S}) \geq \delta_3 + 2\delta_3^{<-1>} + \delta_3 = 4\delta_3 - 2\delta_2$. Since $u$ is left in $G_{-S}$, we know that the two degree-2 neighbors of $u$ in $G$ are also left in $G_{-S}$. Thus, there is a chain containing at least three degree-2 vertices (including $u$) in $G_{-S}$, by Lemma 23, we can further decrease the measure $p$ by at least $2\delta_2$ by applying reduction rules on $G_{-S}$. In total, the measure $p$ decreases by at least

$$4\delta_3 - 2\delta_2 + 2\delta_2 = 4\delta_3.$$

Note that $6\delta_3 - \delta_2 \geq 4\delta_3$. In this branching, we can always reduce the measure $p$ by at least $4\delta_3$.

In the second branching, $v$ is included in the independent set and at least $N[v]$ is deleted. We let $S = N[v]$ and consider whether $u \in S \cup R_S$ or not. If $u \in S \cup R_S$, we apply Corollary 1 with $S'$ being the vertex set $N[v] \cup N[u]$ plus all the vertices in the chain between $u$ and $v$. Now $S'$ contains at least four degree-3 vertices and three degree-2 vertices and $k = 6$. The measure $p$ decreases by at least

$$4\delta_3 + 3\delta_2 + (3\delta_3^{<-1>} + \delta_2) = 7\delta_3 + \delta_2.$$

If $u \notin S \cup R_S$, then $u \in N(S \cup R_S)$. We apply Lemma 36 by letting $S' = N[v]$ and $k = 5$. Then $p(G) - p(G_{-S}) \geq 3\delta_3 + \delta_2 + 5\delta_3^{<-1>} = 8\delta_3 - 4\delta_2$. Furthermore, $u$ is left in a chain of length at least four in $G_{-S}$. by Lemma 23, we can further decrease the measure $p$ by at least $2\delta_2$ by applying reduction rules on $G_{-S}$. In total, the measure $p$ decreases by at least

$$8\delta_3 - 4\delta_2 + 2\delta_2 = 8\delta_3 - 2\delta_2.$$

Note that $7\delta_3 + \delta_2 \geq 8\delta_3 - 2\delta_2$. In this branching, we can always reduce the measure $p$ by at least $8\delta_3 - 2\delta_2$. We get the claimed branching vector.

**Lemma 45.** *Let $G$ be the graph after Step 10. For any connected component $H$ of $G$, all degree-3 vertices in $H$ either have no degree-3 neighbors or have at least two degree-3 neighbors.*

*Proof.* First, the graph $G$ has no degree-3 vertex with exactly one degree-3 neighbor since Step 9 could not be applied now. If there is a degree-3 vertex $u$ having no degree-3 neighbor and a degree-3 vertex $v$ having at least two degree-3 neighbors in a connected component $H$, then there is a path between $u$ and $v$. We can always choose $u$ and $v$ such that the path between $u$ and $v$ does not contain any degree-3 vertices, i.e., the path is a chain. Thus $u$ and $v$ is chain-adjacent, which means the condition of Step 10 holds, a contradiction to the fact that Step 10 can not be applied now.

**Step 11 (Branching on Degree-3 Vertices With At Least Two Degree-3 Neighbors)** *If there is a connected component $H$ containing a degree-3 vertex with at least two degree-3 neighbors, we let $u$ be the vertex of the maximum weight in $H$ and let $v$ be a degree-3 neighbor of $u$, and branch on $v$ with Branching Rule 1.*

Note that the vertex $u$ of the maximum weight in $H$ can not be a degree-2 vertex, otherwise R5 can be applied on the degree-2 vertex. So $u$ is a degree-3 vertex. By Lemma 45, we know that all degree-3 vertices in $H$ must have at least two degree-3 neighbors, and then we can find a degree-3 neighbor $v$ of $u$, where $v$ also has at least two degree-3 neighbors.

**Lemma 46.** *Step 11 followed by applications of reduction rules creates a branching vector covered by one of*

$$[4\delta_3 - \delta_2, 8\delta_3 - \delta_2] = [3.624, 7.624], \quad \textit{and}$$

$$[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496].$$

*Proof.* Let $q_2$ be the number of degree-2 neighbors of $v$. Then $q_2 = 0$ or 1.

In the first branching of excluding $v$ from the independent set, we let $S = \{v\}$. We distinguish two cases by considering whether $u$ is in $R_S$ or not. If $u \in R_S$, we apply Corollary 1 by letting $S' = \{u, v\}$ and $k = 4$. The measure $p$ decreases by at least $2\delta_3 + 4\delta_3^{<-1>} = 6\delta_3 - 4\delta_2$. If $u \notin R_S$, by Lemma 31, the measure $p$ decreases by $\delta_3 + (3 - q_2)\delta_3^{<-1>} + q_2\delta_3 = 4\delta_3 - (3 - q_2)\delta_2$. Furthermore, vertex $u$ is left as a degree-2 vertex. Since $u$ has the maximum weight in $H$, we know that R5 can be applied on $u$ to further decrease the measure $p$ by $2\delta_2$. Thus, in this branching, the measure $p$ decreases by at least $\min\{4\delta_3 - (1 - q_2)\delta_2, 6\delta_3 - 4\delta_2\} = 4\delta_3 - (1 - q_2)\delta_2$ for $q_2 = 0$ or 1.

In the second branching, $v$ is included in the independent set and at least $N[v]$ is deleted. For the case that $q_2 = 0$, we apply Lemma 36 by letting $S' = N[v]$ and $k = 6$. The measure $p$ decreases by at least $4\delta_3 + \delta_3 + 2\delta_2 + 3\delta_3^{<-1>} = 8\delta_3 - \delta_2$. For the case that $q_2 = 1$, we apply Lemma 36 by letting $S' = N[v]$ and $k = 5$. The measure $p$ decreases by at least $3\delta_3 + \delta_2 + 5\delta_3^{<-1>} = 8\delta_3 - 4\delta_2$. So, we get the two claimed branching vectors.

By Lemma 45, we know that after Step 12, no pair of degree-3 vertices are adjacent.

**Step 12 (Branching on Other Degree-3 Vertices)** *Pick up an arbitrary degree-3 vertex v and branch on it with Branching Rule 1.*

**Lemma 47.** *Step 12 followed by applications of reduction rules creates a branching vector covered by*

$$[4\delta_3 + 6\delta_2, 4\delta_3 + 6\delta_2] = [6.256, 6.256].$$

*Proof.* Let $\{u_1, u_2, u_3\}$ be the three degree-3 chain-neighbors of $v$. By Lemma 27 we know that the three degree-3 vertices are different.

In the first branch $S = \{v\}$ and in the second branch $S = S_v \supseteq N[v]$. Recall that we use $R_S$ to denote the set of deleted vertices during applying R1 to R4 on $G - S$ and let $G_{-S} = G - (S \cup R_S)$. In each branch, all vertices in $N[v]$ will be deleted in $G_{-S}$. If at least one vertex in $\{u_1, u_2, u_3\}$, say $u_1$ is deleted in $G_{-S}$, then we apply Corollary 1 by letting $S'$ being the vertex set $N[v] \cup N[u_1]$ together with all degree-2 vertices in the chain between $v$ and $u$. Then $S'$ contains at least two degree-3 vertices and five degree-2 vertices, and $k = 4$. The measure $p$ decreases by at least $\sum_{u \in S'} \delta_{d(u)} + 4\delta_3^{<-1>} = 2\delta_3 + 5\delta_2 + 4\delta_3^{<-1>} = 6\delta_3 + \delta_2$. If all the three vertices in $\{u_1, u_2, u_3\}$ are left in $G_{-S}$, then all of them will become degree-2 vertices in $G_{-S}$. For this case, we apply Corollary 1 by letting $S' = N[v]$ and $k = 3$. The measure $p$ decreases by at least $\sum_{u \in S'} \delta_{d(u)} + 3\delta_3^{<-1>} = 4\delta_3$. However, each vertex in $\{u_1, u_2, u_3\}$ has two degree-2 neighbors in $G_{-S}$. In $G_{-S}$, reduction rules on degree-2 vertices can be applied for at least three times to reduce chains of length $\geq 4$ (even when two vertices in $\{u_1, u_2, u_3\}$ are in the same chain and have a common degree-2 neighbor). So the measure $p$ can be further reduced by $6\delta_2$. In total, the measure $p$ will decrease by at least $4\delta_3 + 6\delta_2$. Note that $4\delta_3 + 6\delta_2 < 6\delta_3 + \delta_2$. We get the claimed branching vector.

It is easy to see that above steps cover all the cases. Among all the branching vectors, the bottleneck ones are $[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496]$ in Lemma 43, $[4\delta_3 + \delta_2, 6\delta_3] = [4.376, 6]$ in Lemma 43, and $[4\delta_3, 8\delta_3 - 4\delta_2] = [4, 6.496]$ in Lemma 46. All of them have a branching factor of 1.14427. So we get that

**Theorem 1.** Maximum Weighted Independent Set *can be solved in* $O^*(1.1443^p)$ *time and polynomial space.*

By Lemma 1 and Theorem 1, we get that

**Corollary 2.** Maximum Weight Independent Set *in graphs with average degree at most $x$ can be solved in* $O^*(1.1443^{(0.624x-0.872)n})$ *time and polynomial space.*

Let $x = 3$ in Lemma 1, we get that $p \leq n$ and the following result.

**Theorem 2.** Maximum Weight Independent Set *in graphs with the average degree at most three can be solved in* $O^*(1.1443^n)$ *time and polynomial space.*

## 6 Conclusion

In this paper, we design an exact algorithm for Maximum Weighted Independent Set. With the help of the measure-and-conquer technique, we analyze a nontrivial running time bound for the algorithm, which has a good performance on sparse graphs. For graphs with an average degree at most three, the running time bound is $O^*(1.1443^n)$, improving previous running time bounds for the problem in cubic graphs using polynomial space. Although the improvement is incremental, such improvements on classic problems have became harder and harder. Any further improvement may need new observations on the structural properties or new techniques to design and analyze the algorithms. Foe unweighted Maximum Independent Set on degree-3 graphs, the running time bound was improved for several times [7, 29, 5, 23, 28, 4, 32, 15]. Each improvement is small, but each improvement reveals new properties and new analysis. Our algorithm is analyzed by the measure-and-conquer technique. The framework of the analysis may also provide a way to analyze other related problems.

## References

1. Akiba, T., Iwata, Y.: Branch-and-reduce exponential/fpt algorithms in practice: A case study of vertex cover. Theoretical Computer Science **609**, 211–225 (2016)
2. Been, K., Daiches, E., Yap, C.K.: Dynamic map labeling. IEEE Transactions on Visualization and Computer Graphics **12**(5), 773–780 (2006)
3. Beigel, R.: Finding maximum independent sets in sparse and general graphs. In: Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 856–857 (1999)

4. Bourgeois, N., Escoffier, B., Paschos, V.T., van Rooij, J.M.M.: Fast algorithms for max independent set. Algorithmica **62**(1), 382–415 (2012)
5. Bourgeois, N., Escoffier, B., Paschos, V.T.: An O*($1.0977^{\mathrm{n}}$) exact algorithm for max independent set in sparse graphs. In: Parameterized and Exact Computation. pp. 55–65 (2008)
6. Chen, J., Kanj, I.A., Jia, W.: Vertex cover: Further observations and further improvements. Journal of Algorithms **41**(2), 280–301 (2001)
7. Chen, J., Kanj, I.A., Xia, G.: Labeled search trees and amortized analysis: Improved upper bounds for np-hard problems. Algorithmica **43**(4), 245–273 (2005)
8. Dahllöf, V., Jonsson, P.: An algorithm for counting maximum weighted independent sets and its applications. In: Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 292–298 (2002)
9. Dahllöf, V., Jonsson, P., Wahlström, M.: Counting models for 2sat and 3sat formulae. Theoretical Computer Science **332**(1-3), 265–291 (2005)
10. Fomin, F.V., Gaspers, S., Saurabh, S.: Branching and treewidth based exact algorithms. In: Proceedings of Seventeenth International Symposium on Algorithms and Computation. Lecture Notes in Computer Science, vol. 4288, pp. 16–25 (2006)
11. Fomin, F.V., Gaspers, S., Saurabh, S., Stepanov, A.A.: On two techniques of combining branching and treewidth. Algorithmica **54**(2), 181–207 (2009)
12. Fomin, F.V., Grandoni, F., Kratsch, D.: A measure & conquer approach for the analysis of exact algorithms. Journal of the ACM **56**(5), 25:1–25:32 (2009)
13. Fomin, F.V., Kratsch, D.: Exact Exponential Algorithms. Texts in Theoretical Computer Science. An EATCS Series, Springer (2010)
14. Fürer, M., Kasiviswanathan, S.P.: Algorithms for counting 2-satsolutions and colorings with applications. In: Proceedings of Third International Algorithmic on Aspects in Information and Management. pp. 47–57 (2007)
15. Issac, D., Jaiswal, R.: An $O^*(1.0821^n)$-time algorithm for computing maximum independent set in graphs with bounded degree 3. CoRR **abs/1308.1351** (2013)
16. Jian, T.: An $O(2^{0.304n})$ algorithm for solving maximum independent set problem. IEEE Transactions on Computers **35**(9), 847–851 (1986)
17. Karp, R.M.: Reducibility among combinatorial problems. In: Proceedings of a Symposium on the Complexity of Computer Computations. pp. 85–103. The IBM Research Symposia Series (1972)
18. Kneis, J., Langer, A., Rossmanith, P.: A Fine-grained Analysis of a Simple Independent Set Algorithm. In: Proceedings of IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science. Leibniz International Proceedings in Informatics (LIPIcs), vol. 4, pp. 287–298 (2009)
19. Lamm, S., Schulz, C., Strash, D., Williger, R., Zhang, H.: Exactly solving the maximum weight independent set problem on large real-world graphs. In: Proceedings of Algorithm Engineering and Experiments. pp. 144–158 (2019)
20. Liao, C.S., Liang, C.W., Poon, S.H.: Approximation algorithms on consistent dynamic map labeling. Theoretical Computer Science **640**, 84–93 (2016)
21. McConnell, R.M., de Montgolfier, F.: Linear-time modular decomposition of directed graphs. Discrete Applied Mathematics **145**(2), 198–209 (2005)
22. Niedermeier, R., Rossmanith, P.: On efficient fixed-parameter algorithms for weighted vertex cover. Journal of Algorithms **47**(2), 63–77 (2003)
23. Razgon, I.: Faster computation of maximum independent set and parameterized vertex cover for graphs with maximum degree 3. Journal of Discrete Algorithms **7**(2), 191–212 (2009)

24. Robson, J.M.: Algorithms for maximum independent sets. Journal of Algorithms **7**(3), 425 – 440 (1986)
25. Shachnai, H., Zehavi, M.: A multivariate framework for weighted FPT algorithms. Journal of Computer and System Science **89**, 157–189 (2017)
26. Tarjan, R.E., Trojanowski, A.E.: Finding a maximum independent set. SIAM Journal on Computing **6**(3), 537–546 (1977)
27. Wahlström, M.: A tighter bound for counting max-weight solutions to 2sat instances. In: Proceedings of Third International Workshop on Parameterized and Exact Computation. Lecture Notes in Computer Science, vol. 5018, pp. 202–213 (2008)
28. Xiao, M.: A simple and fast algorithm for maximum independent set in 3-degree graphs. In: WALCOM: Algorithms and Computation. pp. 281–292 (2010)
29. Xiao, M., Chen, J., Han, X.: Improvement on vertex cover and independent set problem for low-degree graphs. Chinese Journal of Computers **28(2)**, 153–160 (2005)
30. Xiao, M., Huang, S., Zhou, Y., Ding, B.: Efficient reductions and a fast algorithm of maximum weighted independent set. In: WWW'21: The Web Conference 2021. pp. 3930–3940 (2021)
31. Xiao, M., Nagamochi, H.: A refined algorithm for maximum independent set in degree-4 graphs. Journal of Combinatorial Optimization **34**(3), 830–873 (2017)
32. Xiao, M., Nagamochi, H.: Confining sets and avoiding bottleneck cases: A simple maximum independent set algorithm in degree-3 graphs. Theoretical Computer Science **469**, 92–104 (2013)
33. Xiao, M., Nagamochi, H.: An exact algorithm for maximum independent set in degree-5 graphs. Discrete Applied Mathematics **199**, 137–155 (2016)
34. Xiao, M., Nagamochi, H.: Exact algorithms for maximum independent set. Information and Computation **255**, 126–146 (2017)