Approximate Generalized Matching: $f$-Factors and $f$-Edge Covers

Dawei Huang
University of Michigan

Seth Pettie*
University of Michigan

Abstract

In this paper we present linear time approximation schemes for several generalized matching problems on nonbipartite graphs. Our results include $O(\epsilon m)$-time algorithms for $(1 - \epsilon)$-maximum weight $f$-factor and $(1 + \epsilon)$-approximate minimum weight $f$-edge cover. As a byproduct, we also obtain direct algorithms for the exact cardinality versions of these problems running in $O(m \sqrt{f(V)})$ time.

The technical contributions of this work include an efficient method for maintaining relaxed complementary slackness in generalized matching problems and approximation-preserving reductions between the $f$-factor and $f$-edge cover problems.

*Supported by NSF grants CCF-1217338, CNS-1318294, CCF-1514383, and CCF-1637546.
1 Introduction

Many combinatorial optimization problems are known to be reducible to computing optimal matchings in non-bipartite graphs [8, 7]. These problems include computing $b$-matchings, $f$-factors, $f$-edge covers, $T$-joins, undirected shortest paths (with no negative cycles), and bidirected flows, see [17, 13, 21, 9]. These problems have been investigated heavily since Tutte’s work in the 1950s [22, 20]. However, the existing reductions to graph matching are often inadequate: they blow up the size of the input [17], use auxiliary space [11], or piggyback on specific matching algorithms [11] like the Micali-Vazirani algorithm [18]. Moreover, most existing reductions destroy the dual structure of optimal solutions and are therefore not approximation preserving.

In this paper we design algorithms for computing $f$-factors and $f$-edge covers in a direct fashion, or through efficient, approximation-preserving reductions. Because our algorithms are based on the LP formulations of these problems (in contrast to approaches using shortest augmenting walks [11, 18]), they easily adapt to weighted and approximate variants of the problems. Let us define these problems formally. We assume graphs may have multiple parallel edges and loops.

**f-factor** An $f$-factor is a subset $F \subset E$ such that $\deg_F(v) \leq f(v)$. $F$ is perfect if the degree constraints hold with equality.

**f-edge cover** An $f$-edge cover is a subset $F \subset E$ such that $\deg_F(v) \geq f(v)$. It is perfect if all degree constraints hold with equality.

On unweighted graphs the $f$-factor objective is to maximize $|F|$, and for $f$-edge cover it is to minimize $|F|$. On weighted graphs it is to maximize/minimize $w(F)$, possibly subject to the additional constraint that $F$ is perfect.

### Classic Reductions.

The classical reduction from $f$-factor to standard graph matching uses the $b$-matching problem as a stepping stone. A $b$-matching is a function $\beta : E \to \mathbb{Z}_{\geq 0}$ (where $\beta(e)$ indicates how many copies of $e$ are in the matching) such that $\sum_{e \in E} \beta(e) \leq b(v)$, i.e., the number of matches edges incident to $v$, counting multiplicity, is at most $b(v)$. The $f$-factor problem on $G = (V, E, w)$ is reduced to $b$-matching by subdividing each edge $e = (u, v) \in E$ into a path $(u, u_e, v_e, v)$. Here $u_e, v_e$ are new vertices, all three edges have weight $w(u, v)$, and $b(u_e) = b(v_e) = 1$. The original vertices have $b(u) = f(u)$. This reduction blows up the number of vertices to $O(m)$ and is not approximation preserving. The $b$-matching problem is easily reduced to standard matching by replicating each vertex $u$ $b(u)$ times, and replacing each edge $(u, v)$ with a bipartite $b(u) \times b(v)$ clique on its endpoints’ replicas. This step of the reduction is approximation preserving, but blows up the number of vertices and edges. Both reductions together reduce $f$-factor to a graph matching problem on $O(m)$ vertices and $O(f_{\max}m)$ edges. Gabow [11] gave a method for solving $f$-factor in $O(m\sqrt{f(V)})$ time using black-box calls to single iterations of the Micali-Vazirani [18] algorithm.

Observe that $f$-factor and $f$-edge cover are complementary problems: if $C$ is an $f_C$-edge cover, the complementary edge set $F = E \setminus C$ is necessarily an $f_F$-factor, where $f_F(v) = \deg(v) - f_C(v)$. Complementarity implies that any polynomial-time algorithm for one problem solves the other in polynomial time, but it says nothing about the precise complexity of solving them exactly or approximately. Indeed, this phenomenon is very well known in the realm of NP-complete problems. For example, Maximum Independent Set and Minimum Vertex Cover are complementary problems.

---

[1] In the literature the term $f$-factor often refers to our definition of a perfect $f$-factor.
but have completely different approximation profiles. Gabow’s $O(m\sqrt{f_F(V)})$ cardinality $f$-factor algorithm \cite{11} implies that $f_C$-edge cover is computed in $O(m\sqrt{m - f_C(V)}) = O(m^{3/2})$ time, and says nothing about the approximability of $f$-edge cover. As far as we are aware, the fastest approximation algorithms for $f$-edge cover (see \cite{2}) treat it as a general weighted Set Cover problem on 2-element sets. Chvátal’s analysis \cite{3} shows the greedy algorithm is an $H(2)$-approximation, where $H(2) = 3/2$ is the 2nd harmonic number.

Our interest in the approximate $f$-edge cover problem is inspired by a new application to anonymizing data in environments where users have different privacy demands; see \cite{2, 1, 16}. Here the data records correspond to edges and the privacy demand of $v$ is measured by $f(v)$; the goal is to anonymize as few records to satisfy everyone’s privacy demands.

**New Results.** We give new algorithms for computing $f$-factors and $f$-edge covers approximately and exactly.

- We show that a folklore reduction from minimum weight 1-edge cover to maximum weight 1-factor (matching) is approximation-preserving, in the sense that any $(1 - \epsilon)$-approximation for matching gives a $(1 + \epsilon)$-approximation for edge cover. This implies that 1-edge cover can be $(1 + \epsilon)$-approximated in $O_{\epsilon}(m)$ time \cite{5}, and that one can apply any number of simple and practical algorithms \cite{4, 19, 5} to approximate 1-edge cover. This simple reduction does not extend to $f$-factors/$f$-edge covers.

- We give an $O_{\epsilon}(m)$-time $(1 + \epsilon)$-approximation algorithm for weighted $f_C$-edge cover, for any $f_C$. Our algorithm follows from two results, both of which are somewhat surprising. First, any approximate weighted $f_F$-factor algorithm that reports a $(1 \pm \epsilon)$-optimal dual solution can be transformed into a $(1 + O(\epsilon))$-approximate weighted $f_C$-edge cover algorithm. Second, such an $f_F$-factor algorithm exists, and its running time is $O_{\epsilon}(m)$. The first claim is clearly false if we drop the approximate dual solution requirement (for the same reason that an $O(1)$-approximate vertex cover does not translate into an $O(1)$-approximate maximum independent set), and the second is surprising because the running time is independent of the demand function $f_F$ and the magnitude of the weight function.

- As corollaries of these reductions, we obtain a new exact algorithm for minimum cardinality $f_C$-edge cover running in $O(m\sqrt{f_C(V)})$ time, rather than $O(m^{3/2})$ time (\cite{11}), and a direct algorithm for cardinality $f_F$-factor that runs in $O(m\sqrt{f_F(V)})$ time, without reduction \cite{11} to the Micali-Vazirani algorithm \cite{18}.

The blossom structure and LP characterization of $b$-matching is considerably simpler than the corresponding blossoms/LPs for $f$-factor and $f$-edge cover. In the interest of simplicity, one might want efficient code that solves (approximate) $b$-matching directly, without viewing it as an $f$-factor problem on a multigraph in which there is implicitly an infinite supply of each edge. We do not know of such a direct algorithm. Indeed, the structure of $b$-matching blossoms seems to rely on strict complementary slackness, and is *incompatible* with our main technical tool, relaxed complementary slackness\(^2\)

\(^2\)Using relaxed complementary slackness, matched and unmatched edges have *different* eligibility criteria (to be included in augmenting paths and blossoms) whereas $b$-matching blossoms require that all copies of an edge—matched and unmatched alike—are all eligible or all ineligible.
Structure of the Paper. In Section 3, we give an introduction to the LP-formulation of generalized matching problems and the structure of their blossoms and augmenting walks. In Section 4.1, we show that a folklore reduction from 1-edge cover to 1-factor is approximation-preserving and in Section 4.2, we reduce approximate $f$-edge cover to approximate $f$-factor. In Section 5, we give an $O(W m \epsilon^{-1})$-time algorithm for $(1 - \epsilon)$-approximate $f$-factor in graphs with weights in $[1, W]$ and then speed it up to $O(m \epsilon^{-1} \log \epsilon^{-1})$, independent of weight. Section 6 gives a linear time algorithm to compute a maximal set of augmenting walks; cf. [15, §8].

2 Preliminaries and Terminology

The input is a multigraph $G = (V, E)$. For $S \subseteq V$, let $\delta(S)$ and $\gamma(S)$ be set of edges with exactly one endpoint and both endpoints in $S$. For $T \subseteq E$, $\delta_T(S)$ denotes the intersection of $\delta(S)$ and $T$.

By definition, $\deg_T(S) = |\delta_T(S)|$.

3 Basis of $f$-factor and $f$-edge cover

This section extends basic algorithmic toolkits from matching to the context of $f$-factor and $f$-edge cover, e.g., blossoms, augmenting walks and LPs. This establishes basic tools for generalizing the Duan-Pettie algorithm [5] for Maximum Weight Matching to Maximum Weight $f$-factor and Minimum Weight $f$-edge cover.

3.1 LP formulation

The maximum weight $f$-factor problem can be expressed as maximizing $\sum_{e \in E} w(e) x(e)$, subject to the following constraints:

$$\sum_{e \in \delta(v)} x(e) \leq f(v), \text{ for all } v \in V$$

$$\sum_{e \in \gamma(B) \cup I} x(e) \leq \left\lfloor \frac{f(B) + |I|}{2} \right\rfloor, \text{ for all } B \subseteq V, I \subseteq \delta(B)$$

$$0 \leq x(e) \leq 1, \text{ for all } e \in E$$

Here, the blossom constraint $\sum_{e \in \gamma(B) \cup I} x(e) \leq \left\lfloor \frac{f(B) + |I|}{2} \right\rfloor$ is a generalization of blossom constraint $\sum_{e \in \gamma B} x(e) \leq \left\lfloor \frac{|B|}{2} \right\rfloor$ in ordinary matching. The reason that we have a subset $I$ of incident edges in the sum is that the subset allows us to distinguish between matched edges that have both endpoints inside $B$ with those with exactly one endpoint. Any basic feasible solution $x$ of this LP is integral [21, §33], and can therefore be interpreted as membership vector of an $f$-factor $F$. To ensure optimality of the solution, the algorithm works with the dual LP, which is:

$$\text{minimize } \sum_{v \in V} f(v) y(v) + \sum_{B \subseteq V, I \subseteq \delta(B)} \left\lfloor \frac{f(B) + |I|}{2} \right\rfloor z(B, I) + \sum_{e} u(e)$$

subject to $yz_F(e) + u(e) \geq w(e)$, for all $e \in E$

$y(v) \geq 0, z(B, I) \geq 0, u(e) \geq 0$
Here the aggregated dual \( y z_F : E \mapsto \mathbb{R}_{\geq 0} \) is defined as:

\[
y z_F(u,v) = y(u) + y(v) + \sum_{B, I: (u,v) \in \gamma(B) \cup I, I \subseteq \delta(B)} z(B, I).
\]

Unlike matching, each \( z \)-value here is associated with the combination of an odd vertex set \( B \) and a subset \( I \) of its incident edges.

The minimum weight \( f \)-edge cover problem can be expressed as minimizing \( \sum_{e \in E} w(e) x(e) \), subject to:

\[
\sum_{e \in \delta(v)} x(e) \geq f(v), \text{ for all } v \in V \\
\sum_{e \in \gamma(B) \cup \delta(B) \setminus I} x(e) \geq \left\lceil \frac{f(B) - |I|}{2} \right\rceil, \text{ for all } B \subseteq V \text{ and } I \subseteq \delta(B) \\
0 \leq x(e) \leq 1, \text{ for all } e \in E
\]

With the dual program being:

maximize \[
\sum_{v \in V} f(v) y(v) + \sum_{B \subseteq V, I \subseteq \delta(B)} \left\lceil \frac{f(B) - |I|}{2} \right\rceil z(B, I) - \sum_{e \in E} u(e)
\]

subject to \( y z_C(e) - u(e) \leq w(e), \text{ for all } e \in E \)

\[
y(v) \geq 0, z(B, I) \geq 0, u(e) \geq 0
\]

where \( y z_C(u,v) = y(u) + y(v) + \sum_{B, I: (u,v) \in \gamma(B) \cup \delta(B) \setminus \delta(I)} z(B, I) \)

Both of our \( f \)-factor and \( f \)-edge cover algorithms maintain a dynamic feasible solution \( F \subseteq E \) that satisfy the primal constraint. We call edges in \( F \) matched and all other edges unmatched (referred to as the type of an edge). A vertex \( v \) is saturated if \( \deg_F(v) = f(v) \). It is unsaturated/oversaturated if \( \deg_F(v) \) is smaller/greater than \( f(v) \). Given an \( f \)-factor \( F \), the deficiency \( \text{def}(v) \) of a vertex \( v \) is defined as \( \text{def}(v) = f(v) - \deg_F(v) \). Similarly, for an \( f \)-edge cover \( F \), the surplus of a vertex is defined as \( \text{surp}(v) = \text{deg}_F(v) - f(v) \).

### 3.2 Blossoms

We follow Gabow’s [12] definitions and terminology for \( f \)-factor blossoms, augmenting walks, etc. A blossom is a tuple \( (B, E_B, \beta(B), \eta(B)) \) where \( B \) is the vertex set, \( E_B \) the edge set, \( \beta(B) \subseteq B \) the base vertex and \( \eta(B) \subset \delta(B) \cap \delta(B), |\eta(B)| \leq 1 \) the base edge. We often refer to the blossom by referring to its vertex set \( B \). Blossoms can be defined inductively as follows.

**Definition 1.** A single vertex \( v \) forms a trivial blossom, or a singleton. Here \( B = \{v\}, E_B = \emptyset \), and \( \beta(B) = v, \eta(B) = \emptyset \).

Inductively, let \( B_0, B_1, \ldots, B_{l-1} \) be a sequence of disjoint singletons or nontrivial blossoms. Suppose there exists a closed walk \( C_B = \{e_0, e_1, \ldots, e_{l-1}\} \subseteq E \) starting and ending with \( B_0 \) such that

\[ \text{Note: We use } y z_F \text{ and } y z_C \text{ to denote the aggregated dual } y z \text{ for } f \text{-factor and } f \text{-edge cover respectively. We will omit the subscript if it is clear from the context.} \]
The vertex set $B = \bigcup_{i=0}^{l-1} B_i$ is identified with a blossom if the following are satisfied:

1. **Base Requirement:** If $B_0$ is a singleton, the two edges incident to $B_0$ on $C_B$, i.e., $e_0$ and $e_{l-1}$, must both be matched or both be unmatched.

2. **Alternation Requirement:** For non-base $B_i$, $i \neq 0$, if $B_i$ is a singleton, exactly one of $e_{i-1}$ and $e_i$ is matched. If $B_i$ is a nontrivial blossom, $\eta(B_i) \neq \emptyset$ and must be either $\{e_{i-1}\}$ or $\{e_i\}$.

The edge set of the blossom $B$ is $E_B = C_B \cup \bigcup_{i=1}^{l-1} E_{B_i}$. We define $\beta(B) = \beta(B_0)$. If $B_0$ is not a singleton, $\eta(B) = \eta(B_0)$. Otherwise, $\eta(B)$, if nonempty, is guaranteed to satisfy $\eta(B) \subseteq \delta(B) \cap \delta(\beta(B_0))$. It is also guaranteed that $\eta(B)$ is of different type to $e_0$ and $e_{l-1}$.

Blossoms are classified as **light/heavy**. If $B_0$ is a singleton, $B$ is light/heavy if $e_0$ and $e_{l-1}$ are both unmatched/matched. Otherwise, $B$ is light/heavy if $B_0$ is light/heavy. Blossoms in ordinary matching (1-factor) are always light.

The purpose of blossom is to identify the part of the graph that behaves as a whole when looking for an augmenting path. This property can be formally stated as follows:

**Lemma 2.** Let $v$ be an arbitrary vertex in $B$. There exists an even length alternating walk $P_0(v)$ and an odd length alternating walk $P_1(v)$ from $\beta(B)$ to $v$ using edges in $E_B$. Moreover, the terminal edge incident to $\beta(B)$ must have a different type than $\eta(B)$, if $\eta(B)$ exists.

**Proof.** We can prove this by induction, the base case is a blossom $B$ consists of all singletons $\langle v_1, v_2, ..., v_{2k+1} \rangle$ and $v = v_i$ for some $1 \leq i \leq 2k+1$. Then one of the two walks $\langle v_1, v_2, ..., v_i \rangle$ and $\langle v_1, v_{2k+1}, v_{2k}, ..., v_i \rangle$ must be odd and the other must be even.

Consider the cycle $C_B = \langle B_0, e_0, B_1, ..., e_{l-2}, B_{l-1}, e_{l-1}, B_0 \rangle$. Suppose the claim holds inductively for all nontrivial blossom in $B_0, B_1, ..., B_{l-1}$. Let $v$ be an arbitrary vertex in $B$. We use $P_{B_{i,j}}$ $(0 \leq i < l, j \in \{0,1\})$ to denote the walk $P_0$ and $P_1$ guarantee in blossom $B_i$. There are two cases:

**Case 1:** When $v$ is contained in a singleton $B_k$. We examine the two walks $\hat{P} = \langle e_0, e_1, ..., e_{k-1} \rangle$ and $\hat{P'} = \langle e_{i-1}, e_{i-2}, ..., e_k \rangle$. Notice that $\hat{P}$ and $\hat{P'}$ are walks in the graph obtained by contracting all subblossoms $B_0, B_1, ..., B_{l-1}$ of $B$. By the induction hypothesis, we can extend $\hat{P}$ and $\hat{P'}$ to $P$ and $P'$ in $G$ by replacing each $B_i$ on the walk with $P_{B_{i,j}}$ for suitable $j$ connecting the endpoints of $e_{i-1}$ and $e_i$. In particular, if $e_{i-1}$ and $e_i$ are of different types, we replace $B_i$ with the even length walk $P_{B_i,0}$. Otherwise, we replace with $P_{B_i,1}$. By the alternation requirement, one of $P$ and $P'$ must be odd and the other must be even.

**Case 2:** When $v$ is contained in a blossom $B_k$. We assume by the alternation requirement that $e_{k-1} = \eta(B_k)$. Consider the contracted walk $\hat{P} = \langle e_0, e_1, ..., e_{k-1} \rangle$. We extend $\hat{P}$ to an alternating walk $P$ in $E_B$ terminating at $e_{k-1}$ similar to Case 1. Then $P_0(v)$ or $P_1(v)$ is obtained by concatenating $P$ with the alternating walk $P_{B_k,0}(v)$ or $P_{B_k,1}(v)$ of the right parity.

Notice that in both cases, the base requirement in Definition [1] guarantees the starting edge of the alternating walk $P_1(v)$ and $P_0(v)$ alternates with the base edge $\eta(B)$.

The main difference between blossoms in generalized matching problems and blossoms in ordinary matching problem is that $P_0$ and $P_1$ are both meaningful in finding augmenting walks or blossoms. In ordinary matching, since each vertex has at most 1 matched edge incident to it, an alternating walk entering the blossom on the base vertex must leave the blossom on a nonbase vertex with an unmatched edge. As a result the subwalk inside the blossom is always even. In generalized
matching problems, this subwalk can be either even or odd. As a result, an alternating walk entering the blossom at a base edge can leave the blossom at any nonbase edge, whether matched or unmatched.

We also define the notion of maturity for blossoms, for both $f$-factor and $f$-edge cover. For simplicity let us focus on $f$-factor first. By complementary slackness, we can only assign a positive $z(B,I)$ for the pair $(B,I)$ if it satisfy the constraint $|F \cap (\gamma(B) \cup I)| \leq \lceil (f(B) + |I|)/2 \rceil$ with equality. This requirement can be captured in the following definition:

**Definition 3 (Mature Blossom).** A blossom is mature w.r.t an $f$-factor $F$ if it satisfies the following:

1. Every vertex $v \in B \setminus \{\beta(B)\}$ is saturated: $\deg_F(v) = f(v)$.
2. $\text{def}(\beta(B)) = 0$ or $1$. If $\text{def}(\beta(B)) = 1$, $B$ must be a light blossom and $\eta(B) = \emptyset$; If $\text{def}(\beta(B)) = 0$, $\eta(B) \neq \emptyset$.

This is motivated by two observations:

1. Complementary slackness: dual variables can be positive only if its primal constraint is satisfied with equality, i.e. a blossom can have a positive $z$-value only if $|F \cap (\gamma(B) \cup I(B))| = \lceil (f(B) + |I|)/2 \rceil$.
2. Topology of Augmenting Walks: Unsaturated heavy blossoms cannot start or end an augmenting walk since we cannot extend it to an augmenting walk in $G$ that starts with an unmatched edge.

Several remarks can be made here:

- A blossom that is not mature may contain an augmenting walk or augmenting cycle. Specifically, suppose $B$ is light and unsaturated. If any nonbase vertex $v \neq \beta(B)$ in $B$ is also unsaturated, the odd length alternating walk from $\beta(B)$ to $v$ satisfy the definition of an augmenting walk. Moreover, if $\beta(B)$ has deficiency more than 2, the alternating cycle $C_B$ is an augmenting cycle. We do not want to contract these blossoms since we might lose track of augmenting cycles.
- The maturity of an $f$-factor blossom $B$ will imply $|F \cap (\gamma(B) \cup I(B))| = \left\lceil \frac{f(B) + |I(B)|}{2} \right\rceil$. For $f$-edge cover, if $B$ is mature, we have $|F \cap (\gamma(B) \cup (\delta(B) \setminus I(B)))| = \left\lceil \frac{f(B) - |I(B)|}{2} \right\rceil$.
- Augmentation never destroys maturity. In particular, it never creates unsaturated heavy blossom.

Maturity for $f$-edge cover blossoms can be defined similarly and has similar properties.

**Definition 4 (Mature Blossom for $f$-edge cover).** A blossom is mature w.r.t an $f$-edge cover $F$ if it satisfies the following:

1. Every vertex $v \in B \setminus \{\beta(B)\}$ is saturated: $\deg_F(v) = f(v)$.
2. $\text{surp}(\beta(B)) = 0$ or $1$. If $\text{surp}(\beta(B)) = 1$, $B$ must be a heavy blossom and $\eta(B) = \emptyset$; If $\text{surp}(\beta(B)) = 0$, $\eta(B) \neq \emptyset$.
The algorithm keeps track of a laminar set $\Omega \subset 2^V$ of mature blossoms and maintains a non-negative $z$ value for each $B \in \Omega$ with one subset of $I(B) \subseteq \delta(B)$ of its neighborhood. The subset is defined as $I(B) = \delta_F(B) \oplus \eta(B)$, where $\oplus$ is the symmetric difference operator (XOR). All other subsets $I$ of $\delta(B)$ will have $z(B,I) = 0$. An important consequence of maturity is that if $B$ is a mature blossom, we have $|F \cap (\gamma(B) \cup I(B))| = \left\lfloor \frac{f(B) + |I(B)|}{2} \right\rfloor$ (or in $f$-edge cover: $|F \cap (\gamma(B) \cup (\delta(B) \setminus I(B)))| = \left\lceil \frac{(f(B) - |I(B)|)}{2} \right\rceil$).

### 3.3 Augmenting/Reducing Walk

Augmenting walks are the analogy for augmenting paths from ordinary matching. However, augmenting walks for $f$-factor can have a more complex topology than augmenting paths in ordinary matching. Issues arises from the fact that a blossom no longer behave like a single vertex after they are contracted. For example, in Figure 2, both before and after augmentation, the two edges, $(v_0, v_1)$ and $(v_4, v_6)$ that incident to blossom $\{v_1, v_2, v_4, v_5, v_3\}$ on the augmenting walk $\langle v_0, v_1, v_3, v_5, v_4, v_6 \rangle$ are of the same type, which can never happen in ordinary matching! Moreover, augmenting walks can begin and end at the same vertex and can visit the same vertex multiple times. Hence a naive contraction of a blossom into a single vertex does not preserve structures of augmenting walks and we need a more sophisticated definition for augmenting walks. The following definition takes this into account:

**Definition 5.** Let $\hat{G}$ be the graph obtained from $G$ by contracting a laminar set $\Omega$ of blossoms. Let $\hat{P} = \langle B_0, e_0, B_1, e_1, ..., B_{l-1}, e_{l-1}, B_l \rangle$ be a walk in $\hat{G}$. Here $\{e_i\}$ are edges and $\{B_i\}$ are blossoms or singletons, with $e_i \in B_i \times B_{i+1}$ for all $0 \leq i < l$. We say $\hat{P}$ is an augmenting walk with respect to the $f$-factor $F$ if the following requirements are satisfied:

---

Figure 1: Two example of contractable blossoms: Bold edges are matched and thin ones are unmatched. Blossoms are circled with border. Base edges are represented with arrow pointing away from its corresponding blossom.
Figure 2: An example for how a blossom changes with an augmentation: here the augmenting walk is \((v_0, v_1, v_3, v_5, v_4, v_6)\). Notice after rematching, the base edge of the blossom changes from \((v_0, v_1)\) to \((v_4, u_6)\), and the blossom turns from a heavy blossom to a light one.

1. **Terminal Vertices Requirement:** The terminals \(B_0\) and \(B_l\) must be unsaturated singletons or unsaturated light blossoms. If \(P\) is a closed walk \((B_0 = B_l)\), \(B_0\) must be a singleton and \(\text{def}(\beta(B_0)) \geq 2\). Otherwise \(B_0\) and \(B_l\) can be either singletons or blossoms and their deficiency must be positive.

2. **Terminal Edges Requirement:** If the terminal vertex \(B_0\) (\(B_l\)) is a singleton, the incident terminal edges \(e_0\) (\(e_{l-1}\)) must be unmatched. Otherwise they can be either matched or unmatched.

3. **Alternation Requirement:** Let \(B_i, 0 < i < l\), be an internal singleton or blossom. If \(B_i\) is a singleton, the exactly one of \(e_{i-1}\) and \(e_i\) is matched. If \(B_i\) is a nontrivial blossom, \(\eta(B_i) \neq \emptyset\) and one of \(e_{i-1}\) or \(e_i\) must be in \(\eta(B_i)\).

A natural consequence of the above definition is that an augmenting walk in \(\hat{G}\) can be extended to an augmenting walk in \(G\). This can be proved using argument in Lemma \[\text{Lemma 2}\]. Rematching \(F\) along an augmenting walk \(P\) means updating \(F\) to the symmetric difference \(F \oplus P\). This will decrease the total deficiency by 2. Moreover, after rematching along \(P\), every blossom \(B \in \Omega\) still satisfies the definition for blossoms (with different base vertices and edges).

In \(f\)-edge cover, the corresponding notion is called **reducing walk**. The definition of reducing walk can be naturally obtained from Definition \[\text{Definition 5}\] while replacing “unsaturated”, “deficiency” and “light” with “oversaturated”, “surplus” and “heavy”. We leave the details to readers. It is also worth pointing out that if an \(f\)-factor \(F\) and an \(f'\)-edge cover \(F'\) are complement to each other, i.e. \(F' = E \setminus F\) and \(f(v) + f'(v) = \text{deg}(v)\), and they have the same set of blossom \(\Omega\), then an augmenting walk \(\hat{P}\) for \(F\) is also a reducing walk for \(F'\).
3.4 Complementary Slackness

To characterize an (approximately) optimal solution, we maintain dual functions: \( y: V \mapsto \mathbb{R}_{\geq 0} \) and \( z: 2^V \mapsto \mathbb{R}_{\geq 0} \). Here \( z(B) \) is short for \( z(B, I(B)) \). We do not explicitly maintain the edge dual \( u: E \mapsto \mathbb{R}_{\geq 0} \) since its maximizing value can be explicitly given by \( u(e) = \max\{w(e) - yz(e), 0\} \).

For \( f \)-factor \( F \), the following property characterizes an approximate maximum weight \( f \)-factor:

**Property 1** (Approximate Complementary Slackness for \( f \)-factor). Let \( \delta_1, \delta_2 \geq 0 \) be nonnegative parameters. We say an \( f \)-factor \( F \), duals \( y, z \) and the set of blossoms \( \Omega \) satisfies \((\delta_1, \delta_2)\)-approximate complementary slackness if the following holds:

1. Approximate Domination. For each unmatched edge \( e \in E \setminus F \), \( yz(e) \geq w(e) - \delta_1 \).
2. Approximate Tightness. For each matched edge \( e \in F \), \( yz(e) \leq w(e) + \delta_2 \).
3. Blossom Maturity. For each blossom \( B \in \Omega \), \( |F \cap (\gamma(B) \cup I(B))| = \left\lfloor \frac{f(B) + |I(B)|}{2} \right\rfloor \).
4. Unsaturated Vertices Duals. For each unsaturated vertex \( v \), \( y(v) = 0 \).

The following lemma states Property 1 characterizes an approximately optimal solution.

**Lemma 6.** Let \( F \) be an \( f \)-factor in \( G \) along with duals \( y, z \) and let \( F^* \) be the maximum weight \( f \)-factor. If \( F, \Omega, y, z \) satisfy Property 1 with parameter \( \delta_1 \) and \( \delta_2 \), we have \( w(F) \geq w(F^*) - \delta_1|F^*| - \delta_2|F| \).

**Proof.** We first define \( u: E \mapsto \mathbb{R} \) as

\[
u(e) = \begin{cases} w(e) - yz(e) + \delta_2, & \text{if } e \in F, \\ 0, & \text{otherwise}. \end{cases}
\]

From Approximate Tightness, we have \( u(e) \geq 0 \) for all \( e \in E \). Moreover, \( yz(e) + u(e) \geq w(e) - \delta_1 \) for all \( e \in E \) and \( yz(e) + u(e) = w(e) + \delta_2 \) for all \( e \in F \). This gives the following:

\[
w(F) = \sum_{e \in F} w(e) = \sum_{e \in F} (yz(e) + u(e) - \delta_2) \\
= \sum_{v \in V} \deg_F(v)y(v) + \sum_{B \in \Omega} |F \cap (\gamma(B) \cup I(B))|z(B) + \sum_{e \in F} u(e) - |F|\delta_2
\]

By Property 1 (Unsaturated Vertices Duals, Blossom Maturity and definition of \( u \)), this is equal to

\[
= \sum_{v \in V} f(v)y(v) + \sum_{B \in \Omega} \left\lfloor \frac{f(B) + |I(B)|}{2} \right\rfloor z(B) + \sum_{e \in E} u(e) - |F|\delta_2 \\
\geq \sum_{v \in V} \deg_{F^*}(v)y(v) + \sum_{B \in \Omega} |F^* \cap (\gamma(B) \cup I(B))|z(B) + \sum_{e \in F^*} u(e) - |F|\delta_2 \\
= \sum_{e \in F^*} (yz(e) + u(e)) - |F|\delta_2 \\
\geq \sum_{e \in F^*} (w(e) - \delta_1) - |F|\delta_2 = w(F^*) - |F^*|\delta_1 - |F|\delta_2.
\]

\[\square\]
4 Connection Between $f$-Factors and $f$-Edge Covers

The classical approach for solving $f$-edge cover problem is reducing it to $f$-factor. Specifically, looking for a minimum weight $f$-edge cover $F$ can be seen as choosing edges that are not in $F$, which is a maximum weight $f'$-factor where $f'(u) = \operatorname{deg}(u) - f(u)$. There are two drawbacks to this reduction:

First it yields inefficient algorithms: Gabow’s algorithms [12] for solving maximum weight $f'$-factor scales linearly with $f'(V)$, which makes it undesirable when $f$ is small. Even for $f(V) = O(n)$, Gabow’s algorithm runs in $O(m^2 + mn \log n)$.

Moreover, this reduction is not approximation-preserving. In other words, the complement of an arbitrary $(1-\epsilon)$-approximate maximum weight $f'$-factor is not guarantee to be a $(1+\epsilon)$-approximate $f$-edge cover.

In this section we establish two results: First we prove that a folklore reduction from $1$-edge cover to matching is approximation preserving. This allows us to use an efficient approximate $f$-factor algorithm from our $(1-\epsilon)$ approximate maximum weight $f$-factor algorithm.

4.1 Approximate Preserving Reduction from 1-Edge Cover to 1-Factors

The edge cover problem is a special case of $f$-edge cover where $f$ is 1 everywhere. We can reduce the minimum weight edge cover to maximum weight matching by only reweighting edges [21]. Let $e(v)$ be an edge with minimum weight in $\delta(v)$ and let $\mu(v) = w(e(v))$. We define the new weight function $w'(u,v) = \mu(u) + \mu(v) - w(u,v)$. Schrijver [21] showed the following theorem:

**Theorem 7.** Let $M^*$ be a maximum weight matching with respect to weight function $w'$, and $C = M^* \cup \{e(v) \colon v \in V \setminus V(M)\}$. Then $C$ is a minimum weight edge cover with respect to weight function $w$.

We show this reduction also preserves approximation ratio:

**Theorem 8.** Let $M'$ be a $(1-\epsilon)$-maximum weight matching with respect to weight function $w'$, and $C' = M' \cup \{e(v) \colon v \in V \setminus V(M')\}$. Then $C'$ is a $(1+\epsilon)$-minimum weight edge cover with respect to weight function $w$.

**Proof.** Let $C^*$ and $M^*$ be the optimal edge cover and matching defined previously. By construction, we have

$$w(C) = w(M') + \mu(V \setminus V(M')) = \mu(V(M')) - w'(M') + \mu(V \setminus V(M')) = \mu(V) - w'(M')$$

Similarly, we have $w(C^*) = w(V) - w'(M^*)$. Then

$$w(C) = \mu(V) - w'(M) \leq \mu(V) - (1-\epsilon)w'(M^*) = w(C^*) + \epsilon w'(M^*) \leq (1+\epsilon)w(C^*)$$

The last inequality holds because we have $w'(u,v) = \mu(u) + \mu(v) - w(u,v) \leq w(u,v)$.

The reduction does not naturally extend to $f$-edge cover. In the next section we will show how to obtain a $(1+\epsilon)$-approximate $f$-edge cover algorithm from a $(1-\epsilon)$-approximate $f$-factor within the primal-dual framework.
4.2 From f-factor to f-edge cover

We show that a primal-dual algorithm computing a \((1-\epsilon)\)-approximate f-factor can be used to compute an \((1+\epsilon)\)-approximate f-edge cover. We start by giving the approximate complementary slackness for f-edge cover. Similar to f-factor, the property characterizes a good approximate f-edge cover.

Property 2 (Approximate Complementary Slackness for f-edge cover). Let \(\delta_1, \delta_2 \geq 0\) be positive parameters. We say an f-edge cover \(F\), with duals \(y, z\) and blossom family \(\Omega\) satisfies the \((\delta_1, \delta_2)\)-approximate complementary slackness if the following requirements holds:

1. Approximate Domination. For each unmatched edge \(e \in E \setminus F\), \(yz_C(e) \leq w(e) + \delta_1\).
2. Approximate Tightness. For each matched edge \(e \in F\), \(yz_C(e) \geq w(e) - \delta_2\).
3. Blossom Maturity. For each blossom \(B \in \Omega\), \(|F \cap (\gamma(B) \cup (\delta(B) \setminus I_{EC}(B)))| = \left[\frac{f(B) - |I_{EC}(B)|}{2}\right]\).
4. Oversaturated Vertices Duals. For each oversaturated vertex \(v\), \(y(v) = 0\).

Recall that we are using the aggregated duals \(yz_C\) for f-edge cover:

\[
yz_C(u, v) = y(u) + y(v) + \sum_{B: (u, v) \in \gamma(B) \cup (\delta(B) \setminus I_{EC}(B))} z(B)
\]

The proof for the following Lemma 9 is identical to Lemma 6.

Lemma 9. Let \(F\) be an f-edge cover with duals \(y, z, \Omega\) satisfying Property 2 with parameter \(\delta_1\) and \(\delta_2\), and let \(F^*\) be the minimum weight f-edge cover. We have \(w(F) \leq w(F^*) + \delta_1|F^*| + \delta_2|F|\).

The key observation here is if we have an \(f'\)-factor \(F'\), with blossoms \(\Omega\) and duals \(y, z\) satisfying Property 1, and an f-edge cover \(F\) which is \(F\)'s complement, then the same blossom set \(\Omega\) and same duals \(y, z\) can be used to show Property 2 for \(F\). Specifically, we have the following lemma,

Lemma 10. If the duals \(y, z, \Omega\) and an \(f'\)-factor \(F'\) satisfy Property 7 with parameter \(\delta_1', \delta_2'\), then the same duals \(y, z, \Omega\) and the complementary f-edge cover \(F = E \setminus F'\) satisfies Property 2 with parameter \(\delta_1 = \delta_2'\) and \(\delta_2 = \delta_1'\).

Proof. It is trivial to see Property 2(1,2) (Oversaturated Vertices Duals) and Property 1(4) (Unsaturated Vertices Duals) are equivalent to each other.

To show Property 2(1,2) is equivalent to Property 1(2,1) respectively, it suffices to show that the function \(yz_F\) for \(f'\)-factor \(F'\) agrees with the function \(yz_C\) for its f-edge cover complement \(F\). Recall that

\[
yz_C(u, v) = y(u) + y(v) + \sum_{B: (u, v) \in \gamma(B) \cup (\delta(B) \setminus I_{EC}(B))} z(B)
\]

\[
yz_F(u, v) = y(u) + y(v) + \sum_{B: (u, v) \in \gamma(B) \cup I_{F}(B)} z(B)
\]

Here \(I_C\) and \(I_F\) refers to the I-set of a blossom with respect to the f-edge cover \(F\) and the \(f'\)-factor \(F'\). It suffices to argue that \(I_F(B) = \delta(B) \setminus I_C(B)\):

\[
I_F(B) = \{\eta(B)\} \oplus \delta_F(B) = \{\eta(B)\} \oplus (\delta(B) \setminus \delta_C(B))
= \delta(B) \setminus (\{\eta(B)\} \oplus (\delta_C(B))) = \delta(B) \setminus I_C(B)
\]

11
Therefore, in $yz_F(e)$ and $yz_C(e)$, $z$-values are summed up over the same set of blossoms in $\Omega$. In other words, $yz_F(e) = yz_C(e)$ for each $e \in E$ and the claim follows.

For Blossom Maturity in Property 1 and Property 2, we argue that one equality implies the other.

Suppose $\eta(B) \subset F$. We have $I_C(B) = \delta_F(B) \setminus \eta(B)$. Therefore the edge set $F \cap (\gamma(B) \cup (\delta(B) \setminus I_C(B)))$ contains edges with both endpoints inside $B$ plus the edge $\eta(B)$. While $f(B) - |I_C(B)| = f(B) - \deg_F(B) + 1$ is an lower bound on how many endpoints this set of edges can have inside $B$. Equality $|F \cap (\gamma(B) \cup (\delta(B) \setminus I_C(B)))| = \left\lceil \frac{f(B) - |I_C(B)|}{2} \right\rceil$ implies $f(v) = \deg_F(v)$ for every $v \in B$. This means the $f'$-factor $F'$ also saturates every vertex $v \in B$ w.r.t. $f'$. Therefore, equality in Property 1.3 holds. If $\eta(B) \not\subset F$, then $I_C(B) = \delta_F(B)$, the edge set $F \cap (\gamma(B) \cup (\delta(B) \setminus I_C(B)))$ contains edges with both endpoints inside $B$ only. Similarly, $f(B) - |I_C(B)| = f(B) - \deg_F(B)$ is an lower bound on the total number of their endpoints. Equality $|F \cap (\gamma(B) \cup (\delta(B) \setminus I_C(B)))| = \left\lceil \frac{f(B) - |I_C(B)|}{2} \right\rceil$ again implies every vertex in $B$ is saturated so Property 1.3 is satisfied. Notice that the same proof also works for the case when $\eta(B) = \emptyset$. This complete the proof that Property 2.3 implies Property 1.3.

The other direction is symmetric.

5 Approximation Algorithms for $f$-Factor and $f$-Edge Cover

In this section, we prove the main result by giving an approximation algorithm for computing $(1 - \epsilon)$-approximate maximum weight $f$-factor. The crux of the result is an implementation of Edmonds’ search with relaxed complementary slackness as eligibility criterion. The notion of approximate complementary slackness is introduced by Gabow and Tarjan for both bipartite matching [14] and general matching [15]. Gabow gave an implementation of Edmonds’ search with exact complementary slackness for $f$-factor problem [12], which finds augmenting walks one at a time. The main contribution of this section is to adapt [12] to approximate complementary slackness to facilitate finding augmenting walks in batches.
To illustrate how this works, we will first give an approximation algorithm for $f$-factor in graph with small edge weights. Let $w(\cdot)$ be a positive weight function $w : E \mapsto \{1,\ldots,W\}$. The algorithm computes a $(1 - \epsilon)$-approximate maximum weight $f$-factor in $O(mW\epsilon^{-1})$ time, independent of $f$. We also show how to use scaling techniques to transform this algorithm to run in $O(m\epsilon^{-1}\log\epsilon^{-1})$ time, independent of $W$.

5.1 Approximation for small weight

The main procedure in our $O(mW\epsilon^{-1})$ time algorithm is called Edmonds’ search. In one iteration, Edmonds’ search finds a set of augmenting walks using eligible edges, creates and dissolve blossoms, and performs Dual Adjustment on $y$ and $z$ while maintaining the following Invariant:

**Invariant 1** (Approximate Complementary Slackness). Let $\delta > 0$ be some parameter such that $w(e)$ is a multiple of $\delta$, for all $e \in E$:

1. **Granularity.** $y$-values are multiples of $\delta/2$ and $z$-values are multiples of $\delta$.
2. **Approximate Domination.** For each unmatched edge and each blossom edge $e \in (E \setminus F) \cup \left(\bigcup_{B \in \Omega} E_B\right)$, $yz(e) \geq w(e) - \delta$.
3. **Tightness.** For each matched and each blossom edge $e \in F \cup \left(\bigcup_{B \in \Omega} E_B\right)$, $yz(e) \leq w(e)$.
4. **Blossom Maturity.** For each blossom $B \in \Omega$, $|F \cap (\gamma(B) \cup I(B))| = \left\lfloor \frac{f(B) + |I(B)|}{2} \right\rfloor$.
5. **Unsaturated Vertices.** All unsaturated vertices have the same $y$-value; their $y$-values are strictly less than the $y$-values of other vertices.

Notice that here we relax Property 1(4) to allow unsaturated vertices to have positive $y$-values. The purpose of Edmonds’ search is to decrease the $y$-values for all unsaturated vertices while maintaining Invariant 1. Following [15, 5, 6], we define the following eligibility criterion:

**Criterion 1.** An edge $(u, v)$ is eligible if it satisfies one of the following:

1. $e \in E_B$ for some $B \in \Omega$.
2. $e \notin F$ and $yz(e) = w(e) - \delta$.
3. $e \in F$ and $yz(e) = w(e)$.

A key property of this definition is that it is asymmetric for matched and unmatched edges. As a result, if we augment along an eligible augmenting walk $P$, all edges in $P$, except for those in contracted blossoms, will become ineligible.

Let $G_{\text{elig}}$ be the graph obtained from $G$ by discarding all ineligible edges, and let $\tilde{G}_{\text{elig}} = G_{\text{elig}}/\Omega$ be obtained from $G_{\text{elig}}$ by contracting all blossoms in $\Omega$. For initialization, we set $F = \emptyset$, $y = W/2$, $z = 0$, $\Omega = \emptyset$. Edmonds’ search repeatedly executes the following Augmentation, Blossom Formation, Dual Adjustment, and Blossom Dissolution steps until all unsaturated vertices have 0 $y$-values:

1. **Augmentation.** Find a maximal set $\tilde{\Psi}$ of augmenting path with a laminar set of reachable mature blossom $\Omega'$ in $\tilde{G}_{\text{elig}}$: $\Psi$ be the preimage of $\tilde{\Psi}$ in $G_{\text{elig}}$. Update $F \leftarrow F \oplus \bigcup_{P \in \Psi} P$ and $\Omega \leftarrow \Omega \cup \Omega'$. After this step, the new $\tilde{G}_{\text{elig}}$ contains no augmenting walk.
2. **Blossom Formation.** Find a maximal set of nested mature blossoms $\Omega'$ reachable from an unsaturated vertex. Set $\Omega ← \Omega \cup \Omega'$ and update $G_{\text{elig}}$. After this step, $G_{\text{elig}}$ contains no contractable blossom reachable from an unsaturated vertex.

3. **Dual Adjustment.** Let $\hat{S}$ be the set of vertices from $G_{\text{elig}}$ reachable from an unsaturated vertex via an eligible alternating walk. We classify vertices in $\hat{S}$ into $\hat{V}_{\text{in}}$, the set of inner vertices and $\hat{V}_{\text{out}}$, the set of outer vertices. Let $V_{\text{in}}$ and $V_{\text{out}}$ be the set of original vertices in $V$ represented by $\hat{V}_{\text{in}}$ and $\hat{V}_{\text{out}}$. Adjust the $y$ and $z$ values as follows:

$$y(v) ← y(v) - \delta/2, \text{ if } v \in V_{\text{out}}$$
$$y(v) ← y(v) + \delta/2, \text{ if } v \in V_{\text{in}}$$
$$z(B) ← z(B) + \delta, \text{ if } B \text{ is a root blossom in } \hat{V}_{\text{out}}$$
$$z(B) ← z(B) - \delta, \text{ if } B \text{ is a root blossom in } \hat{V}_{\text{in}}$$

4. **Blossom Dissolution.** After Dual Adjustment some root blossoms in $\Omega$ might have 0 $z$-value. Remove them from $\Omega$ until none exists. Update $\Omega$ and $G_{\text{elig}}$. Notice that root blossoms with 0 $z$-value can be generated because some root blossoms have their $z$-value decremented in Dual Adjustment step, or root blossoms formed in Augmentation step become unreachable after augmentation and thus do not get their $z$-value incremented.

To finish the description, we still need to define reachable vertices as well as inner/outer labelling of blossoms/singleton. We start by defining the notion of alternation that follows from Definition 5 of an augmenting walk. We say two edges $e$ and $e'$ that are both incident to a blossom/singleton $B$ alternates, if either $B$ is a singleton and $e$ and $e'$ are of different types, or $B$ is a nontrivial blossom, and $|\eta(B) \cap \{e, e'\}| = 1$. An alternating path is a path $P$ where every two adjacent edges alternates. Notice that the main difference between this and alternating paths in matching is that we have a different criterion of edge alternation for nontrivial blossoms, saying that two edges, even when they have the same type, alternates with each other if one of them is the base edge and the other is not.

The search forest $\hat{S}$ consists of blossoms and singletons that are reachable from an unsaturated blossom and singleton, via an eligible alternating path in $G_{\text{elig}}$. Furthermore we require the preimage if those paths in $G_{\text{elig}}$ start with an unsaturated vertex and an unmatched edge. Hence the roots of $\hat{S}$ can only be unsaturated singletons or unsaturated light blossoms. We label the root vertices outer. If $v$ is a nonroot vertex in the search forest let $\tau(v)$ be the edge in $\hat{S}$ pointing to the parent of $v$. The inner/outer status of vertices is defined as follows:

**Definition 11.** A vertex $v$ is outer if one of the following is satisfied:

1. $v$ is the root of a search tree.
2. $v$ is a singleton and $\tau(v) \in F$.
3. $v$ is a nontrivial blossom and $\tau(v) = \eta(v)$.

Otherwise, one the following holds and $v$ is classified as inner:

1. $v$ is a singleton and $\tau(v) \in E \setminus F$. 

2. \( v \) is a nontrivial blossom and \( \tau(v) \neq \eta(v) \).

An individual search tree in \( \hat{S} \), call it \( \hat{T} \), can be grown by repeatedly attaching a child \( v \) to its parent \( u \) using an edge \( (u, v) \) that is eligible for \( u \) in \( \hat{S} \). Let \( B_u \) denote the root blossom in \( \Omega \) containing \( u \). We say an edge \( (u, v) \in E \) is eligible for \( u \) or \( B_u \) if it is eligible and one of the following is satisfied:

1. \( u \) is an outer singleton and \( e \not\in F \).
2. \( B_u \) is an outer blossom and \( e \neq \eta(B_u) \).
3. \( u \) is an inner singleton and \( e \in F \).
4. \( B_u \) is an inner blossom and \( e = \eta(B_u) \).

Hence, \( \hat{S} \) consists of singletons and blossoms that are reachable from an unsaturated singleton or light blossom, via an eligible alternating path whose preimage starts with an unsaturated vertex and an unmatched edge. For simplicity, we call such blossoms and singletons reachable, and all other singletons and blossoms unreachable. We call a vertex \( v \) in original graph \( G \) eligible reachable (unreachable) if \( B_v \) is reachable (unreachable) in \( \hat{G}_{\text{elig}} \).

In our version of Edmonds’ Search, primal and dual are initialized in a way that Property 1(1) (Approximate Domination) is always satisfied, and Property 1(3) (Approximate Tightness) is vacuous (as the \( f \)-factor is initially empty) but Property 1(4) (Unsaturated Vertices) is not. For this reason, there is a large gap between primal and dual objective at the beginning of the algorithm, which can be evaluated by the following:

\[
yz(V) - w(F) = \sum_{v \in V} f(v)y(v) + \sum_{B \in \Omega} \left[ \frac{f(B) + |I(B)|}{2} \right] z(B) + \sum_{e \in E} u(e) - \sum_{e \in F} w(e)
= \sum_{v \in V} \text{def}(v)y(v).
\]

The goal of the algorithm can be seen as bridging the gap between the primal objective and dual objective while preserving all other complementary slackness properties. It can be achieve in two ways. Augmentations enlarge the \( f \)-factor by augmenting \( F \) along some augmenting walk \( P \). This will reduce the total deficiency on the vertex set \( V \). Dual Adjustments change the dual variables in a way that decreases the \( y \)-value on unsaturated vertices while maintaining other complementary slackness condition. In this algorithm, the progress of Edmonds’ Search is measured by the latter, i.e. the overall decrement on \( y \)-value for all unsaturated vertices.

The correctness of our algorithm reduces to showing that Augmentation, Blossom Formation, Blossom Dissolution, and in particular Dual Adjustment all preserve Invariant \( \hat{1} \).

**Lemma 12.** Augmentation preserves Invariant \( \hat{I} \).

**Proof.** We first show that after augmentation, the \( I \)-set of any blossom on \( B \) is invariant. As a result, the function \( yz(\cdot) \) is unchanged. We use \( I(B), \eta(B) \) and \( I'(B), \eta'(B) \) to denote the \( I \)-set and base edge of \( B \) before and after the augmentation. By Definition \( \hat{5} \) (augmenting walks), we have

\[
\delta_P(B) = \eta(B) \cup \eta'(B) = \eta(B) \oplus \eta'(B)
\]
Figure 4: An example of an eligible alternating search tree. Outer blossoms and singletons are labeled using solid boundaries while inner blossoms and singletons have dashed boundaries.
Let \( F \) and \( F' \) be the \( f \)-factor before and after augmentation, we have

\[
\delta_{F'}(B) = \delta_F(B) \oplus \delta_P(B)
\]

Combining both equations, we have

\[
\delta_{F'}(B) = \delta_F(B) \oplus (\eta(B) \oplus \eta'(B))
\]

Hence

\[
I'(B) = \delta_{F'}(B) \oplus \eta'(B) = \delta_F(B) \oplus \eta(B) = I(B)
\]

By Invariant \([1]\) any blossom edge \( e \in \bigcup_{B \in \Omega} E_B \) satisfies both Approximate Domination as well as Tightness, so it continues to satisfy these Invariants after augmentation. For any eligible edge not in \( E_B \) for any \( B \in \Omega \), by Criterion \([2,3]\) if \( e \in F \), \( yz(e) = w(e) - \delta \), thus after augmentation its duals satisfy Approximate Domination. If \( e \) is unmatched, \( yz(e) = w(e) \), so its duals satisfy Tightness after augmentation.

We also show that Augmentation preserves maturity of blossoms. For any vertex \( v \) in a non-terminal blossom \( B \), \( \deg_F(v) = \deg'_F(v) = f(v) \), so maturity is naturally preserved. If \( B \) is a terminal blossom, we have \( \deg_F(v) = f(v) - 1 \) for \( v = \beta(B) \) and \( \deg_F(v) = f(v) \) for all \( v \neq \beta(B) \). Moreover, \( B \) always has a base edge \( \eta(B) = \delta_P(B) \). Therefore, \( B \) is also mature after augmentation.

**Lemma 13.** Blossom Formation and Blossom Dissolution both preserve Invariant \([7]\).

**Proof.** In both steps, we are only creating or discarding blossoms of 0 \( z \)-values, both of which preserve the value of the \( yz \) function. In the Blossom Formation step, we guarantee every new blossom in \( \Omega' \) must be mature.

The crux of the proof is to show that Dual Adjustment also preserves Invariant \([1]\) in particular Approximate Domination and Tightness. Before proving the correctness of Dual Adjustment, we first prove the following parity lemma, which was first used in \([13]\); we generalize it to \( f \)-factor:

**Lemma 14 (Parity).** Let \( \widehat{S} \) be the search forest defined as above. Let \( S \) be the preimage of \( \widehat{S} \) in \( G \). The \( y \)-value of every vertex in \( S \) has the same parity, as a multiple of \( \delta/2 \).

**Proof.** The claim clearly holds after initialization as all vertices have the same \( y \)-values. Because every eligible edge \((u,v) \in \widehat{S} \) that straddles two singletons or blossoms must have \( yz(u,v) = y(u) + y(v) + \sum_{B: (u,v) \in (B) \cup I(B)} z(B) \) being a multiple of \( \delta \), and \( z \)-values are always multiples of \( \delta \), \( y(u) \) and \( y(v) \) will always share the same parity, as multiple of \( \delta/2 \). Therefore it suffices to show that every vertex in a blossom \( B \in \Omega \) has the same parity.

To prove this, we only need to show that Blossom Formation step only groups vertices with same parity together. This is because new blossoms \( B \) are formed with when edges in \( C_B \) are eligible because of Criterion \([2,3]\), this means their endpoints share the same parity. Hence by induction, all vertices in \( B \) also share the same parity. The Dual Adjustment step also preserves this property as vertices in a blossom will have the same inner/outer classification and thus have their \( y \)-values all incremented or decremented by \( \delta/2 \).

**Lemma 15.** Dual Adjustment preserves Invariant \([7]\).

17
Proof. We focus on (2)(Approximate Domination) and (3)(Tightness). Other invariants are not affected by Dual Adjustment.

There are much more cases to consider in f-factor compared to ordinary matching. For edge $(u, v)$, different cases can be generated by (1) inner/outer classification of both endpoints, (2) whether $(u, v)$ is matched, (3) whether $(u, v)$ is the base edges for its endpoints if they are blossoms, and (5) whether $(u, v)$ is eligible. Hence technically, the number of cases can explode to $4 \times 2 \times 2 \times 2 \times 2 = 64$. In the following analysis, we narrow down the meaningful cases to 8.

We consider an edge $e = (u, v)$. If $u$ and $v$ are both unreachable from any unsaturated vertex, or both are in the same root blossom, $yz(u, v)$ clearly remains unchanged after the dual adjustment. Therefore we can assume $B_u \neq B_v$ and at least one of them, say $B_u$, is reachable. Every reachable endpoint will contribute a change of $\pm \delta/2$ to $yz(u, v)$. This is the adjustment of $y(u)$, plus the adjustment of $z(B_u)$ if $e \in I(B)$. Let $\Delta(u)$ be the net change of the quantity $y(u) + \sum_{e \in I(B_u)} z(B_u)$. We omit $e$ and use $\Delta(u)$ when the edge $e$ we are considering is clear. By how we perform Dual Adjustment, we have the following scenarios:

1. $\Delta(u) = +\delta/2$: This occurs if $u$ is an inner singleton, or $B_u$ is an outer blossom with $e \in I(B_u)$, or an inner blossom with $e \not\in I(B_u)$. 
2. $\Delta(u) = -\delta/2$: This occurs if $u$ is an outer singleton, or $B_u$ is an inner blossom with $e \in I(B_u)$, or an outer blossom with $e \not\in I(B_u)$.

Then we consider the effect of a Dual Adjustment on edge $e = (u, v)$. First we consider the case when exactly one of $B_u$ and $B_v$, say $B_u$, is in $\hat{S}$. In this case only $u$ will introduce a change on $yz(u, v)$:

Case 1: $u$ is an inner singleton: Here $\Delta(u) = +\delta/2$. In this case Approximate Domination is preserved, so we only need to worry about approximate tightness and hence assume $e \in F$. Since $B_v$ is not in $\hat{S}$, $e$ cannot be eligible, or $B_v$ would have been included in $\hat{S}$ as a child of $B_u$. Hence $yz(e) < w(e)$. By Granularity, $yz(e) \leq w(e) - \delta/2$. Therefore we have $yz(e) \leq w(e)$ after the Dual Adjustment.

Case 2: $u$ is an outer singleton: Here $\Delta(u) = -\delta/2$. In this case Tightness is preserved and we only need to worry about approximate domination when $e \not\in F$. Similar to Case 1, $e$ must be ineligible and $yz(e) \geq w(e) - \delta/2$. After Dual Adjustment we have $yz(e) \geq w(e) - \delta$.

Case 3: $B_u$ is an inner blossom: We divide the cases according to whether $e$ is matched or not.

Subcase 3.1: $e \in F$. If $e \not\in \eta(B_u)$, then $e \in I(B_u)$ and $\Delta(u) = -\delta/2$. In this case Tightness is preserved. If $e \in \eta(B_u)$, then $e \not\in I(B_u)$ and $\Delta(u) = +\delta/2$. But $e$ cannot be eligible since otherwise $B_u$ must be in the search tree, so we have $yz(e) \leq w(e) - \delta/2$ and $yz(e) \leq w(e)$ after Dual Adjustment.

Subcase 3.2: $e \not\in F$. This is basically symmetric to Subcase 3.1. If $e \in \eta(B_u)$, then $e \in I(B_u)$ and $\Delta(u) = -\delta/2$. But $e$ cannot be eligible therefore $yz(e) \geq w(e) - \delta/2$ and $yz(e) \geq w(e) - \delta$ after Dual Adjustment. If $e \not\in \eta(B_u)$, then $e \not\in I(B_u)$ and $\Delta(u) = +\delta/2$, so approximate Domination is preserved.

Case 4: $B_u$ is an outer blossom:

Subcase 4.1: $e \in F$. If $e \in \eta(B_u)$, then $B_v$ must be the parent of $B_u$ in the search tree, contradicting the fact that $B_v \not\in \hat{S}$. Thus $e \not\in \eta(B_u)$, so $e \in I(B_u)$ and $\Delta(u) = +\delta/2$. Since
$B_u$ is not reachable, $e$ cannot be eligible, so $yz(u,v) \leq w(e) - \delta/2$ before Dual Adjustment and $yz(u,v) \leq w(e)$ afterward.

**Subcase 4.2:** $e \notin F$. Similarly, $e \notin \eta(B_u)$, so $e \notin I(B_u)$ and $\Delta(u) = -\delta/2$. Similarly $B_v$ is not reachable so $e$ cannot be eligible. Therefore we have $yz(u,v) \geq w(e) - \delta/2$ and $yz(u,v) \geq w(e) - \delta$ after Dual Adjustment.

This completes the case when exactly one of $e$’s endpoints is reachable. The following part will complete the argument for when both endpoints are reachable. We argue that three scenarios can happen: either $\Delta(u)$ and $\Delta(v)$ are of opposite signs and cancel each other out, or $\Delta(u)$ and $\Delta(v)$ are of the same sign and the sign aligns with the property we wish to keep, or if both cases does not hold, we use Lemma 14 (Parity) to argue that there is enough room for Dual Adjustment not to violate Approximate Domination or Tightness.

We first examine tree edges in $\hat{S}$. In this case we assume $B_u$ is the parent of $B_v$ and $e$ is the parent edge of $B_v$. Hence $e$ must be eligible for $B_u$. We argue by the sign of $\Delta(u)$.

**Case 5:** $\Delta(u) = +\delta/2$:
There are three cases here: $u$ is an inner singleton, $B_u$ is an outer blossom with $e \in I(B_u)$, or $B_u$ is an inner blossom with $e \notin I(B_u)$. We first observe that in all three cases, $e \in F$. This is straightforward when $u$ is an inner singleton. If $B_u$ is an outer blossom with $e \in I(B_u)$, we know that since $B_u$ is outer, $e \notin \eta(B_u)$, so therefore $e \in F$. If $B_u$ is an inner singleton with $e \notin I(B_u)$, since $B_u$ is inner, $e \notin \eta(B_u)$, so combined with the fact that $e \notin I(B_u)$ we have $e \in F$.

Notice that since $B_u$ is the parent of $B_v$, and $e \in F$, $v$ can be an outer singleton, or $B_v$ is an outer blossom with $e \in \eta(B_v)$, or $B_v$ is an inner blossom with $e \notin \eta(B_v)$. In the second case $e \notin I(B_v)$ and in the third case $e \in I(B_v)$. In all three cases we have $\Delta(v) = -\delta/2$, and $yz(e)$ remains unchanged.

**Case 6:** $\Delta(u) = -\delta/2$: Case 6 is symmetric to Case 5. $B_u$ can either be an outer singleton, an inner blossom with $e \in I(B_u)$ or an outer blossom with $e \notin I(B_u)$. In all cases, the fact that $e$ must be eligible for $B_u$ implies $e \notin F$, and $B_u$ can only be an inner singleton, an outer blossom with $e \in I(B_u)$ or an inner blossom with $e \notin I(B_u)$. Hence we have $\Delta(v) = +\delta/2$ so $yz(e)$ still remains constant.

Now suppose $B_u$ and $B_v$ are both in $\hat{S}$ but $(u,v)$ is not a tree edge. We still break the cases according to the sign of $\Delta(u)$ and $\Delta(v)$. Here we only need to consider when $\Delta(u) = \Delta(v)$, since otherwise they cancel each other and $yz(e)$ remains constant.

**Case 7:** $\Delta(u) = \Delta(v) = \delta/2$. In this case $yz(e)$ is incremented by $\delta$. Therefore we only need to worry about Tightness when $e \in F$. Notice that $B_u$ can only be an inner singleton, an outer blossom with $e \in I(B_u)$ or an inner blossom with $e \notin I(B_u)$. When $B_u$ is an outer blossom, $e \notin \eta(B_u)$. When $B_u$ is an inner blossom, since $e \in F$ and $e \notin I(B_u)$, $e \notin \eta(B_u)$. The same holds for the other endpoint $B_v$.

It is easy to verify that in all cases, $e$ is eligible for $B_u$ (or $B_v$) if and only if $e$ is eligible. But notice that after Augmentation and Blossom Formation steps, there is no augmenting walk or reachable blossom in $\hat{G}_{elig}$, i.e., there cannot be an edge $(u,v)$ that is eligible for both endpoints $B_u$ and $B_v$ since otherwise you can find an augmenting walk or a new reachable blossom. Thus $e$ is ineligible and $yz(e) < w(e)$. But by Invariant 1 (Granularity) and Lemma 14 (Parity), both $w(e)$ and $yz(e)$ must be multiples of $\delta$. Therefore we have $yz(e) \leq w(e) - \delta$. This implies $yz(e) \leq w(e)$ after Dual Adjustment.
Case 8: $\Delta(u) = \Delta(v) = -\delta/2$. Here $yz(e)$ is decremented by $\delta$. Similar to the case above, we can assume $e \not\in E$ and only focus on Approximate Domination. $B_u$ can be an outer singleton, inner blossom with $e \in I(B_u)$, or outer blossom with $e \not\in I(B_u)$. Since $e \not\in E$, $e \in I(B_u)$ if and only if $e \in \eta(B_u)$. Therefore if $e$ is eligible, $e$ must be eligible for both $B_u$ and $B_v$. But similar to Case 7, $e$ being eligible for both endpoints will lead to the discovery of additional blossom or augmenting walk in $G_{\text{eligible}}$, which is impossible after Augmentation and Blossom Formation. Therefore we conclude in this case $e$ is ineligible and $yz(e) > w(e) - \delta$. By Lemma 14(Parity), we have $yz(e) \geq w(e)$ before Dual Adjustment, so Approximate Domination still holds after Dual Adjustment.

**Theorem 16.** A $(1 - \epsilon)$-approximate $f$-factor can be computed $O(Wm\epsilon^{-1})$.

**Proof.** By setting $\delta = 1/\lceil \epsilon^{-1} \rceil \leq \epsilon$, $y$-values of unsaturated vertices takes $(W/2)/(\delta/2) = O(W\epsilon^{-1})$ iterations to reach $0$ and thus satisfying Property $[1]$ with $\delta_1 = \delta, \delta_2 = 0$. By invoking Lemma 6, we have

$$w(F) \geq w(F^*) - |F^*|\delta \geq w(F^*) - w(F^*)\delta \geq (1 - \epsilon)w(F^*)$$

For arbitrary $f$-factor $F^*$.

For the running time, each iteration of Augmentation, Blossom Formation, Dual Adjustment and Blossom Dissolution can be implemented in linear time. We defer the detail implementation to Section 6. There are a total of $W/\delta = O(W\epsilon^{-1})$ iterations, so the running time is $O(Wm\epsilon^{-1})$.

As a result of Lemma 10 and Lemma 9, we also obtain the following result:

**Corollary 17.** A $(1 + \epsilon)$-approximate $f$-edge cover can be computed $O(Wm\epsilon^{-1})$.

### 5.2 A Scaling Algorithm for General Weight

In this section, we can modify the $O(Wm\epsilon^{-1})$ weighted $f$-factor algorithm to work on graphs with general weights. The modification is based on the scaling framework in [3]. The idea is to divide the algorithm into into $L = \log W + 1$ scales that execute Edmonds’ search with exponentially diminishing $\delta$. The goal of each scale is to use $O(\epsilon^{-1})$ Edmonds’ searches to halve the $y$-value of all unsaturated vertices while maintaining a more relaxed version of approximate complementary slackness. By manipulating the weight function, Approximate Domination, which is weak at the beginning, is strengthened over scales, while Approximate Tightness is weakened in exchange. We define $\delta_i, 0 \leq i < L$ be the error parameter for each scale, $\delta_0 = O(W\epsilon)$ and $\delta_i = \delta_{i-1}/2$ for $0 < i < L$. Each scale works with a new weight function $w_i$ which is the old weight function rounded down to the nearest multiple of $\delta_i$, i.e, $w_i(e) = \delta_i \lceil w(e)/\delta_i \rceil$. In the last scale $W_L = w$. We maintain a scaled version of Invariant $[1]$ at each scale:

**Invariant 2** (Scaled approximate complementary slackness with positive unsaturated vertices). At scale $i = 0, 1, ..., L = \log W$, we maintain the $f$-factor $F$, blossoms $\Omega$ and duals $y, z$ to satisfy the following invariant:

1. **Granularity.** All $y$-values are multiple of $\delta_i/2$, and $z$-values are multiple of $\delta_i$.
2. **Approximate Domination.** For each $e \not\in E$ or $e \in E_B$ for any $B \in \Omega$, $yz(e) \geq w_i(e) - \delta_i$
3. **Approximate Tightness.** For each $e \in F$ or $e \in E_B$ for any $B \in \Omega$, let $j$ be the index of the earliest scale that $e$ became matched ($e$ can be unmatched and rematched afterwards). We have $yz(e) \leq w_i(e) + 2\delta_j - 2\delta_i$. 


4. **Mature Blossoms.** For each blossom \( B \in \Omega \), \( |F \cap (\gamma(B) \cup I(B))| = \left\lfloor \frac{f(B) + I(B)}{2} \right\rfloor \).

5. **Unsaturated Vertices Duals.** The \( y \)-values of all free vertices are the same and will always be smaller or equal to the \( y \)-values of other vertices.

Based on Invariant 2, Edmonds’ search will use the following Eligibility criterion:

**Criterion 2.** At scale \( i \), an edge \( e \in E \) is eligible if one of the following holds

1. \( e \in E_B \) for some \( B \in \Omega \).
2. \( e \notin F \) and \( yz(e) = w_i(e) - \delta_i \).
3. \( e \in F \) and \( yz(e) - w_i(e) \) is a nonnegative multiple of \( \delta_i \).

Before the start of scale 0, the algorithm initializes \( F, \Omega, y, z \) similar to the algorithm for small edge weights: \( y(u) \leftarrow W/2, \Omega \leftarrow \emptyset, F \leftarrow \emptyset \). At scale \( i \), the duals of unsaturated vertices start at \( W/2^{i+1} \). We execute \( (W/2^{i+2})/\delta_i = O(\epsilon^{-1}) \) iterations of Edmonds’ search with parameter \( \delta_i \) and Criterion 2. The scale terminates when the \( y \)-values of unsaturated vertices are decreased to \( W/2^{i+2} \), or in the last iteration, terminates as they reach 0.

Notice that although the invariant and the eligibility criterion are changed, the fact that Edmonds’ search preserves the complementary slackness invariant, in particular that Dual Adjustment preserves approximate domination and approximate tightness still holds here. By the proof of Lemma 15 as long as the definition of eligibility guarantees the following parity property:

**Lemma 18.** At any point of scale \( i \), let \( S \) be the set of vertices in \( G_{\text{elig}} \) reachable from an unsaturated vertices using eligible edges. The \( y \)-value of any vertex \( v \in V \) with \( B_v \in S \) has the same parity as multiple of \( \delta_i/2 \).

Dual adjustment never changes the \( yz \)-values of edges inside any blossom \( B \in \Omega \), while it will have the following effect on edge \( e \) if its endpoints are lying in different blossoms:

1. If \( e \notin F \) and is ineligible, \( yz(e) \) might decrease but will never exceed the threshold for eligibility, i.e., it will not drop below \( w_i(e) - \delta_i \).
2. If \( e \notin F \) and is eligible, \( yz(e) \) will never decrease.
3. If \( e \in F \) and is ineligible, \( yz(e) \) might increase but will never exceed the threshold for eligibility, i.e., it will not raise above \( w_i(e) + 2\delta_j - 2\delta_i \).
4. If \( e \in F \) and is eligible, \( yz(e) \) will never increase.

In other words, Dual Adjustment will not destroy Approximate Domination and Approximate Tightness. Therefore Edmonds’ search within scale \( i \) will preserve Invariant 2.

We also need to manipulate the duals between different scales to ensure Invariant 2. Formally, after completion of scale \( i \), we can increment all the \( y \)-values by \( \delta_{i+1} \). This will ensure both approximate domination and approximate tightness holds at scale \( i + 1 \): \( w_{i+1}(e) \leq w_i(e) + \delta_{i+1} \), so increasing \( y \)-values by \( \delta_{i+1} \) ensures Approximate Domination. For Approximate Tightness, we have \( yz(e) - w_{i+1}(e) \leq 2\delta_j - 2\delta_i + 2\delta_{i+1} = 2\delta_j - 2\delta_{i+1} \), since \( \delta_{i+1} = \delta_i/2 \).

The algorithm terminates when the \( y \)-values of all unsaturated vertices reaches 0. It terminates with an \( f \)-factor \( F \) and its corresponding duals \( y, z \) and \( \Omega \) satisfying the following property:

**Property 3** (Scale Complementary Slackness).
1. Approximate Domination. For all $e \notin F$ or $e \in E_B$ for any $B \in \Omega$, $yz(e) \geq w(e) - \delta_L$.

2. Approximate Tightness. For all $e \in F$ or $e \in E_B$ for any $B \in \Omega$, let $j$ be the index of the earliest scale that $e$ becomes matched (e can be unmatched and rematched afterwards). We have $yz(e) \leq w(e) + 2\delta_j$.

3. Blossoms Maturity. For all blossoms $B \in \Omega$, $|F \cap (\gamma(B) \cup I(B))| = \left\lfloor \frac{f(V) + I(B)}{2} \right\rfloor$.

4. Unsaturated Vertices Duals. The $y$-values of all free vertices are 0.

This implies domination and tightness are satisfied within some factor $1 + O(\epsilon)$. For Approximate Domination this is easy to see since $w(e) \geq 1$ and $\delta_L = O(\epsilon)$, thus $yz(e) \geq (1 - O(\epsilon))w(e)$ if $e \notin F$.

For Approximate Tightness, we observe that we can lower bound the weight of $e$ if $e$ first enters $F$ at scale $j$, by noticing $e$ must satisfy the eligibility criterion at scale $i$ first for unmatched edges, i.e., $w_i(e) = yz(e) - \delta_i$. But since $y$-values at scale $i$ must be at least $W/2^{i+2}$, so $w_i(e)$ is at least $W/2^{i+2}$. Since $\delta_i = 2W/2^{i+2}, yz(e) \leq (1 + O(\epsilon))w(e)$ if $e \in F$.

With a similar proof to Lemma 6 we can show this characterizes an $(1 - O(\epsilon))$-approximate minimum weight $f$-factor.

The running time of the algorithm is $O(m\epsilon^{-1}\log W)$ because there is a total of $\log W + 1$ scales, and each scale consists of $O(\epsilon^{-1})$ iterations of Edmonds’ search that can be implemented in linear time. Thus, we have

**Theorem 19.** A $(1 - \epsilon)$-approximate maximum weight $f$-factor can be computed in $O(m\epsilon^{-1}\log W)$ time.

**Corollary 20.** A $(1+\epsilon)$-approximate minimum weight $f$-edge cover can be computed in $O(m\epsilon^{-1}\log W)$ time.

### 5.3 A Linear Time Algorithm

We also point out that by applying techniques in §3.2, the algorithm can be modified to run in time independent of $W$. The main idea is we can further restrict our algorithm to ignore an edge $e$ for all but $O(\log \epsilon^{-1})$ scales. Let $\mu_i = W/2^{i+1} + \delta_i$ be the maximum possible weight of an edge, matched or unmatched, being eligible during scale $i$. Let $\text{scale}(e)$ be the $i$ such that $w(e) \in [\mu_i, \mu_{i-1})$. We notice that we can ignore $e$ in any scale $j < \text{scale}(e)$. Moreover, we will also forcibly ignore $e$ at scale $j > \text{scale}(e) + \lambda$ where $\lambda = \log \epsilon^{-1} + O(1)$. Ignoring an otherwise eligible edge might cause violation of approximate tightness and approximate domination. However, since the $y$-values of free vertices are $O(\epsilon w(e))$ at this point, this violation will only amount to $O(\epsilon w(e))$.

To see this, notice that $\mu_i$ is also an upper bound to the amount of change to $yz(e)$ caused by all Dual Adjustment after scale $i$. Hence, after scale $\text{scale}(e) + \lambda$, the total amount of violation to approximate tightness and approximate domination on $e$ can be bounded by $\mu_{\text{scale}(e) + \lambda} = O(\epsilon)\mu_{\text{scale}(e)} = O(\epsilon)w(e)$, which guarantees we still get an $(1 - O(\epsilon))$ approximate solution.

Therefore, every edge take part in at most $O(\log \epsilon^{-1})$ scales, with $O(\epsilon^{-1})$ operation per scale. The total running time is $O(m\epsilon^{-1}\log \epsilon^{-1})$.

**Theorem 21.** A $(1 - \epsilon)$-approximate maximum weight $f$-factor and a $(1+\epsilon)$-approximate minimum weight $f$-edge cover can be computed in $O(m\epsilon^{-1}\log \epsilon^{-1})$ time, independent of the weight function.
6 A Linear Time Augmenting Walk Algorithm

This section presents the linear time augmenting walk algorithm used in the Augmentation step of Edmonds’ search. The goal here is to find a maximal set $\Psi$ of edge disjoint augmenting walks in the graph $G_{elg}$.

The main idea is to apply the modified depth first search of [15, §8], which finds a maximal set of vertex disjoint augmenting walks for ordinary matching in linear time. There are several difficulties in adapting algorithm in [15] for $f$-factor:

1. Instead of vertex disjoint, we are looking for a maximal set of edge disjoint augmenting path, which might lead to intersecting search trees.

2. The algorithm needs to be able to identify augmenting cycles, which are not present in the ordinary matching.

3. The depth first search tree branches on both outer and inner and outer singletons, as well as outer blossoms.

The input is a contracted graph $\hat{G} = G/\Omega$ with $f$-factor $F$. At any moment, we maintain a search forest, which is an ordered set of depth first search trees $\hat{S} = \langle \hat{T}_1, \hat{T}_2, \ldots, \hat{T}_k \rangle$ on minor of $G$. The minor is defined by a dynamic laminar set of blossom $\Omega'$ on $\hat{G}$, where vertices on $\hat{T}_i$ are singletons or maximal blossoms of $\Omega' \cup \Omega$. For a vertex $v$ in $G$, we use $B_v$ to denote the largest blossom in $\Omega \cup \Omega'$ that contains $v$. We use $T_i$ to refers to the set of vertices contained in blossoms of $\hat{T}_i$.

Since search trees are rooted at unsaturated vertices, we can use Definition [11] to define inner/outer vertices in the search forest. This also gives us the set of edges that are eligible for a vertex in the search forest.

Similar to depth first search, $\hat{T}_1, \hat{T}_2, \ldots, \hat{T}_{k-1}$ are search trees from previous terminated searches. Each $\hat{T}_i$ can either be a successful search tree, which contains an augmenting walk $P_i$, or an exhausted search tree, which is a maximal search tree that does not lead to any augmenting walk. We set $P_i = \emptyset$ in this case. We let $\Psi = \bigcup_i P_i$. The final output of the algorithm is $\Psi$ along with the newly discovered blossoms in $\Omega'$, such that $\Psi$ is a maximal set of augmenting walks in $\Omega' \cup \Omega$.

**Invariant 3.** For each vertex $u$ in $T_i \setminus P_i$ where $1 \leq i < k$, there is no edge $(u, v)$ that is eligible for $u$, and $v$ is in some later search tree $T_j$, $j > i$.

In other words, each vertex $u$ in $V(T_i) \setminus V(P_i)$, $i < k$ is exhausted, that is, every edge that is eligible for $u$ is scanned by the depth first search procedure. Notice that a vertex can go from being already exhausted to unexhausted because its inner/outer classification changes and the change introduces a new set of eligible edges. Therefore, we say a vertex $v$ is outer(inner)-exhausted if it is currently outer(inner) and all edges eligible for $v$ are scanned.

Notice that walks in $\Psi$ are edge disjoint but not necessarily vertex disjoint. Suppose we scan the neighborhood of $v$ in the order given by its adjacency list $\langle e_1, e_2, ..., e_{\text{deg}(v)} \rangle$. During the algorithm, each edge in $\delta(v)$ is in one of the following three state:

---

4 As in Edmonds Algorithm, $\Omega'$ is the set of blossoms we discovered during the search for augmenting walk

5 In matching, this corresponds to the property that there do not exist an unmatched edge connecting two outer vertices from different search trees.
• **Explored**, indicating the edge is explored from $v$ when building some search tree $\hat{T}_i$, but either is included in $\hat{T}_i$ and does not lead to any augmenting path, or is not included in $\hat{T}_i$ because the other endpoint is already in some previous search tree.

• **Augmented**, indicating the edge is explored from $v$ in some $\hat{T}_i$ and is part of $P_i$.

• **Unexplored**, indicating it has not been explored in any $\hat{T}_i$ yet.

When the algorithm terminates all the unexplored edge in $\delta(v)$ will be declared unreachable, i.e. after removing all the edges in $\Psi$, the edge cannot be reach via an alternating path starting from any unsaturated vertex. This will be stated in Lemma 22.

The main complication of our algorithm is the alternating structure of the search tree, which necessitates the maintenance of a maximal set of blossoms. Maintaining a maximal set of blossoms allows us to completely decide for each vertex $v$, whether there is an even and odd length alternating path from an unsaturated vertex to $v$. We can reduce it to maintaining the following Invariant:

**Invariant 4.** If $(u,v)$ is an edge such that $(u,v)$ is scanned from both $u$ and $v$, $u$ and $v$ must be in the same blossom.

The last search tree in the search forest $\hat{T}_k$, referred to as the active search tree, is the search tree that the algorithm is currently growing. The algorithm grows the search tree in a depth first manner by extending the active walk, $P_{active}$, i.e. a walk in $\hat{T}_k$ consisting of all vertices that we have not finish exploring yet. The rest of the Invariant states the topology of the search tree as well as its depth first property.

**Invariant 5.**

1. The sequence $\hat{S} = \langle \hat{T}_1, \hat{T}_2, ..., \hat{T}_k \rangle$ consists of edge disjoint alternating trees in the graph $\hat{G} = G/(\Omega \cup \Omega')$ that are rooted at unsaturated singletons/light blossoms.

2. Edges in $\hat{T}_1$ are all explored. If an edge $(u,v)$ is explored but not augmented, then all edges $(v,w)$ eligible for $v$ must also be explored. If $(u,v)$ is augmented, then its parent edge $\tau(B_u)$ must also be augmented.

3. Vertices in $\hat{T}_i$ are singletons or blossoms of $\Omega \cup \Omega'$. For each tree edge $(u,v) = \tau(v)$, $(u,v)$ is eligible for $u$.

4. The active walk $P_{active}$ consists of a walk from root to a $\hat{T}_k$ descendant.

A search tree initially starts with a single unsaturated vertex as its root and the only vertex in the active path. Suppose a blossom $B$ is the last blossom on the active path. In each iteration, we scan the next unexplored edge $(u,v)$, $u \in B$ that is eligible for $B$. Depending on where the other endpoint $v$ is located, we do one of the following:

1. **Augmentations**: If $B_v$ is not saturated, and $(u,v)$ is unmatched, extend the active walk to $v$, then terminate the search return the active walk $P_{active}$ as an augmenting walk. Notice that this step can return an augmenting cycle when $B_v$ is already in the search tree.

2. **DFS extensions**: If the exploration of $(u,v)$ does not lead to an Augmentation step, and either $B_v$ is a nontrivial blossom not in any search tree, or $B_v$ is a singleton, extend the active walk to $v$ and put $v$ into $\hat{T}_k$ as $B_u$’s child. This corresponds to a grow step in Edmonds’ Search.
3. **Blossom Formation**: If \( B_v \) is a descendant of \( B \), and edge \((u,v)\) is also eligible for \( B_v \), we form a blossom consisting of blossoms and singletons from the \( T_k \) path from \( B \) to \( B_v \). Contract this blossom, call it \( B' \), and place the blossom in the active path. Label all these vertices now as outer and ready to be scanned. Note that all previously inner vertices will go from exhausted to not exhausted as an outer vertex now.

4. **DFS retractions**: If every edge \((u,v)\) eligible for \( B \) is already scanned, mark \( B \) as exhausted and remove it from the active walk. If \( B \) is the only vertex in the active walk, terminate the search.

After the search terminates and returns with a search tree, we update the deficiency function (in case we discover an augmenting walk/cycle, and some unsaturated vertices become saturated) and start a new search at an unsaturated vertex that has not been an exhausted search tree root, as long as it exists. Notice that the root of a successful search tree can be reused as long as it is unsaturated after all augmentations so far. The algorithm terminates when all unsaturated vertices are exhausted, where each vertex is either saturated, or is unsaturated but we cannot discover any augmenting path starting from it.

We first show after the search terminates, all unexplored edges are not reachable from any alternating path starting from an unsaturated vertex. This guarantees we did not ignore any edges that might lead to a new augmenting path.

**Lemma 22.** When the search terminates, each unexplored edges \((u,v)\) is unreachable from any alternating path that does not contain any edge in \( \Psi \).

**Proof.** Suppose an unexplored edge \((u,v)\) is reachable from an alternating path consisting of explored but not augmented edges. Without loss of generality let \((u,v)\) be the first reachable unexplored edge in \( \delta(u) \) and let \((w,u)\) be the edge that precede \((u,v)\) in the alternating path. Since \((w,u)\) is explored but not augmented, all unexplored edges must be explored from \( u \), leading to a contradiction. \(\square\)

We show then all operations preserves Invariant 3, 4, and 5.

**Lemma 23.** All operations preserves Invariant 3, 4, and 5.

**Proof.** Invariant 3 is a natural consequence of depth first search. Invariant 5 (1) is immediate from how we maintain blossoms and active walk. Invariant 5 (2) holds because if \((u,v)\) is explored but none of the search on \((v,w)\) eligible for \( v \) returned with an augmenting path, then every edges eligible for \( v \) must also be explored before the search retract from \( u \).

To show Invariant 4 notice by the depth first nature of the search tree, \( B_u \) and \( B_v \) must be ancestor of one another (otherwise, if \( B_u \) is first explored while \( B_v \) is not in the search tree, DFS extension will put \( B_v \) into the search tree as \( B_u \) child). Suppose \( B_u \) is the ancestor of \( B_v \), when we scan the edge \((u,v)\) from \( B_u \), if \((u,v)\) is not eligible for \( v \) and no blossom step can be performed, the edge \((u,v)\) must be made eligible for \( B_v \) in a later blossom step from \( B_u \) or an ancestor of \( B_u \), to \( B_v \) or a descendant of \( B_v \). This puts \( u \) and \( v \) into the same blossom. \(\square\)

With these invariants, we can show the following lemma:

**Lemma 24.** Any augmenting walk \( P' \) must intersect some augmenting walk \( P \in \Psi \) at an edge.
Proof. Suppose we have an augmenting walk \( P' = \langle u_0, u_1, \ldots, u_{2k-2}, u_{2k-1}, u_{2k} \rangle \) that does not intersect \( \Psi \) at any edge. By Lemma 22 we can assume all edges \((u_i, u_{i+1})\) in \( P' \) must be explored or ineligible for \( u_i \). By termination of the algorithm, \( u_0 \) must be an exhausted unsaturated vertex so must be a search tree root. And by definition of augmenting walk \((u_1, u_2)\) must be eligible for \( u_0 \) so must be explored. We show by induction that for \( 0 \leq i \leq 2k - 1 \), \((u_i, u_{i+1})\) must be explored and \( u_{i+1} \) must be in the search forest \( S \).

For \( i = 0 \), by definition \( u_0 \) is a search tree root and \((u_0, u_1)\) is explored from \( u_0 \) and is eligible for \( u_0 \). This means \( u_1 \) must also be in the search tree. If \( u_i \) is in the search tree and \((u_i, u_{i+1})\) is explored from \( u_i \). Consider vertex \((u_{i+1})\), since it is in the augmenting walk, by the alternation requirement for augmenting walk one of \((u_i, u_{i+1})\) and \((u_{i+1}, u_{i+2})\) must be eligible for \( u_{i+1} \), if \((u_{i+1}, u_{i+2})\) is eligible for \( u_{i+1} \), \((u_{i+1}, u_{i+2})\) must be explored and we are done. Otherwise, \((u_i, u_{i+1})\) is eligible for both of its endpoints and thus \((u_i, u_{i+1})\) must be in the same blossom. In this case, Blossom Formation step makes \((u_{i+1}, u_{i+2})\) eligible for \( u_{i+1} \).

Since all \( u_i \)'s is in the search tree and the preceding edge is eligible for its predecessor, \( u_i \)'s, in particular \( u_{2k} \) can not be a unsaturated inner singleton or an unsaturated light inner blossom because otherwise an Augmentation step will be performed. This contradicts the fact that \( P' \) is an augmenting walk. \( \square \)

**Theorem 25.** A maximal set of augmenting walks can be found in linear time.

**Proof.** The algorithm runs in linear time because each edge is explored at most twice, i.e. once in each direction. Moreover, blossom structures here can also be maintained in linear time using incremental-tree disjoint set data structure in [10]. \( \square \)

We say a set of nested blossoms \( \Omega' \) is maximal if after contracting blossoms in \( \Omega' \), no more contractible blossom that is reachable from an unsaturated vertex can be found. Since all augmenting walk has been eliminated after Augmentation, this is equivalent to saying that no edge \((u,v)\) is eligible for both \( B_u \) and \( B_v \). Thus, to implement the blossom step for Edmonds’ search, it suffices to run the linear time FIND-AW once again (which will not find any augmenting walk) and return the blossom set \( \Omega' \) discovered at the Blossom Formation step. By Invariant [5](4), \( \Omega' \) is maximal.

**Corollary 26.** A maximal set of nested blossom can be found in linear time.

**Corollary 27.** One iteration of Edmonds’ Search with Criterion[1] (Approximate Complementary Slackness) can be implemented in linear time.

### 7 Algorithms for unweighted \( f \)-factor and \( f \)-edge cover

In this section we will give an \( O(\sqrt{|V|}m) \) algorithm for both maximum cardinality \( f \)-factor and minimum cardinality \( f \)-edge cover. This is a direct consequence of the \( O(Wm^{-1}) \) algorithm for the weighted problem. This algorithm matches the running time of [11] but does not rely on reduction to Micali-Vazirani Algorithm. Moreover, it is much simpler to state and to analyze.

For illustration purposes we focus on maximum cardinality \( f \)-factor. The algorithm consists of two phases. The first phase, referred to as batch augmentation, finds an \( f \)-factor \( F \) that is close to optimal using an instance of the \( O(Wm^{-1}) \) algorithm. After \( F \) is close to optimum, we discard all dual variables \( y \) and \( z \), dissolve all the blossoms in \( \Omega \) and uses our linear time augmenting walk algorithm to increase the cardinality of \( F \) until \( F \) becomes optimum.

This is stated formally in the following theorem:
Theorem 28. A maximum cardinality $f$-factor can be computed in $O(\sqrt{f(V)m})$ time.

Proof. We can view an maximum cardinality $f$-factor problem as an maximum weight problem with weight function $w(e) = 1$. Choose $\epsilon = 1/\sqrt{f(V)}$, by Theorem 16, we can compute a $(1 - 1/\sqrt{f(V)})$-approximate maximum cardinality matching $F$ in $O(\sqrt{f(V)m})$ time. If $F^*$ is the maximum cardinality $f$-factor, we have

$$|F| \geq (1 - \frac{1}{\sqrt{f(V)}})|F^*| > |F^*| - \frac{1}{\sqrt{f(V)}} \frac{f(V)}{2} > |F^*| - \sqrt{f(V)/2}$$

This means $F$ is only $O(\sqrt{f(V)})$ augmentations away from optimal. Hence we can then discard blossom structure $\Omega$ with duals $y$ and $z$ from the approximate $f$-factor and run the linear time augmenting walk algorithm of Lemma 25 in $G$ with respect to $F$ until $F$ is optimal. By the discussion above, $O(\sqrt{f(V)})$ iterations suffice. The total running time of the algorithm is $O(\sqrt{f(V)m})$. □

Theorem 29. A minimum cardinality $f$-edge cover can be computed in $O(\sqrt{f(V)m})$ time.

Proof. This is similar to Theorem 28. We first use the $O(Wm\epsilon^{-1})$ algorithm for $f$-edge cover given in Section 4 to find an $(1 + \sqrt{f(V)^{-1}})$-approximate minimum cardinality $f$-edge cover $F$. We have $|F| \leq |F^*| + \sqrt{f(V)^{-1}}|F^*|$. Notice that we always have $|F^*| \leq f(V)$ because taking $f(v)$ arbitrary incident edges for each $v$ is a trivial $f$-edge cover with cardinality at most $f(V)$. We have $|F| \leq |F^*| + \sqrt{f(V)}$, which means at most $O(\sqrt{f(V)})$ reduction is needed before $F$ reaches optimal. The total running time of the algorithm is still $O(\sqrt{f(V)m})$. □


References

[1] K. Choromanski, T. Jebara, and K. Tang. Adaptive anonymity via b-matching. In Proceedings of the 26th International Conference on Neural Information Processing Systems, pages 3192–3200, 2013.

[2] K. M. Choromanski, A. Khan, A. Pothen, and T. Jebara. Large scale robust adaptive anonymity via b-edge cover and parallel computation. 2015.

[3] V. Chvatal. A greedy heuristic for the set-covering problem. Mathematics of Operations Research, 4(3):233–235, 1979.

[4] D. E. Drake and S. Hougardy. A simple approximation algorithm for the weighted matching problem. Information Processing Letters, 85(4):211–213, 2003.

[5] R. Duan and S. Pettie. Linear time approximation for maximum weight matching. J. ACM, 61(1), 2014.

[6] R. Duan, S. Pettie, and H.-H. Su. Scaling algorithms for weighted matching in general graphs. In Proceedings 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 781–800, 2017.

[7] J. Edmonds. Maximum matching and a polyhedron with 0,1 vertices. J. of Res. the Nat. Bureau of Standards, 69 B:125–130, 1965.

[8] J. Edmonds. Paths, trees and flowers. Canadian Journal of Mathematics, pages 449–467, 1965.

[9] J. Edmonds and E. Johnson. Matching: A Well-Solved Class of Integer Linear Programs. In Combinatorial Optimization - Eureka, You Shrink!, pages 27–30. 2003.

[10] H. N. Gabow. A linear-time algorithm for a special case of disjoint set union.

[11] H. N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing, pages 448–456, 1983.

[12] H. N. Gabow. Data structures for weighted matching and extensions to b-matching and f-factors. CoRR, abs/1611.07541, 2016.

[13] H. N. Gabow and P. Sankowski. Algebraic algorithms for b-matching, shortest undirected paths, and f-factors. In 2013 IEEE 54th Annual Symposium on Foundations of Computer Science, pages 137–146, 2013.

[14] H. N. Gabow and R. E. Tarjan. Faster scaling algorithms for network problems. SIAM Journal on Computing, 18(5):1013–1036, 1989.

[15] H. N. Gabow and R. E. Tarjan. Faster scaling algorithms for general graph matching problems. J. ACM, 38(4):815–853, 1991.

[16] A. Khan and A. Pothen. A new 3/2-approximation algorithm for the b-edge cover problem. In 2016 Proceedings of the Seventh SIAM Workshop on Combinatorial Scientific Computing, pages 52–61.
[17] E. Lawler. *Combinatorial Optimization: Networks and Matroids*. Dover Books on Mathematics Series. 2001.

[18] S. Micali and V. V. Vazirani. An $O(\sqrt{|V|E})$ algorithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science*, pages 17–27, 1980.

[19] S. Pettie and P. Sanders. A simpler linear time $2/3 - \epsilon$ approximation for maximum weight matching. *Inf. Process. Lett.*, 91(6):271–276, Sept. 2004.

[20] W. Pulleyblank. *Faces of matching polyhedra*. PhD thesis, University of Waterloo, Ontario, Canada, 1973.

[21] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Algorithms and Combinatorics. Springer, 2002.

[22] W. T. Tutte. On the problem of decomposing a graph into $n$ connected factors. *Journal of the London Mathematical Society*, s1-36(1):221–230, 1961.