MATHEMATICAL ENGINEERING TECHNICAL REPORTS

A Blossom Algorithm for Maximum Edge-Disjoint T-Paths

Satoru IWATA and Yu YOKOI

METR 2019–16 August 2019

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: https://www.keisu.t.u-tokyo.ac.jp/research/techrep/

A Blossom Algorithm for Maximum Edge-Disjoint T-Paths *

Satoru Iwata † Yu Yokoi ‡

August 20, 2019

Abstract

Let $G = (V, E)$ be a multigraph with a set $T \subseteq V$ of terminals. A path in G is called a T-path if its ends are distinct vertices in T and no internal vertices belong to T. In 1978, Mader showed a characterization of the maximum number of edge-disjoint T-paths. The original proof was not constructive, and hence it did not suggest an efficient algorithm.

In this paper, we provide a combinatorial, deterministic algorithm for finding the maximum number of edge-disjoint T-paths. The algorithm adopts an augmenting path approach. More specifically, we introduce a novel concept of augmenting walks in auxiliary labeled graphs to capture a possible augmentation of the number of edge-disjoint T-paths. To design a search procedure for an augmenting walk, we introduce blossoms analogously to the blossom algorithm of Edmonds (1965) for the matching problem, while it is neither a special case nor a generalization of the present problem. When the search procedure terminates without finding an augmenting walk, the algorithm provides a certificate for the optimality of the current edge-disjoint T-paths. Thus the correctness argument of the algorithm serves as an alternative direct proof of Mader's theorem on edge-disjoint T-paths. The algorithm runs in $O(|V| \cdot |E|^2)$ time, which is much faster than the best known deterministic algorithm based on a reduction to the linear matroid parity problem.

*Supported by JST CREST, No. JPMJCR14D2.

†Department of Mathematical Informatics, University of Tokyo, Tokyo 113-8656, Japan. Supported by Grant-in-Aid for Scientific Research, No. 17H01699 from JSPS.

‡National Institute of Informatics, Tokyo 101-8430, Japan. Supported by Grant-in-Aid for Scientific Research, No. 18K18004 from JSPS.

1 Introduction

Let $G = (V, E)$ be a multigraph without selfloops. For a specified set $T \subseteq V$ of terminals, a path in G is called a T-path if its ends are distinct vertices in T and no internal vertices belong to T. In 1978, Mader [12] showed a characterization of the maximum number of edge-disjoint T-paths. The theorem naturally extended a previously known min-max theorem on the inner Eulerian case due to Cherkassky [1] and Lovász [10]. Unlike this preceding result, the original proof was not constructive. It did not suggest an efficient algorithm for finding the maximum number of edge-disjoint T-paths.

Subsequently, Mader [13] extended his theorem to the problem of maximum number of openly disjoint T-paths. Lovász [11] then introduced an equivalent variant, called disjoint $\mathcal{S}$-paths, to provide an alternative proof via the matroid matching theorem. See also [18] for a minor correction. Schrijver [16] provided a short alternative proof for Mader's theorem on disjoint $\mathcal{S}$-paths. The proof was again nonconstructive, and it did not lead to an efficient algorithm.

Schrijver [17] described a reduction of the disjoint $\mathcal{S}$-paths problem to the linear matroid parity problem. Consequently, one can use efficient linear matroid parity algorithms [2, 5, 14, 15] for finding the maximum number of disjoint $\mathcal{S}$-paths (or openly disjoint T-paths). The current best running time bound is $O(n^\omega)$, where n is the number of vertices and ω is the exponent of the fast matrix multiplications. This bound is achieved by the randomized algebraic algorithm of Cheung, Lau, and Leung [2]. The best deterministic running time bound due to Gabow and Stallmann [5] is $O(mn^\omega)$, where m is the number of edges. Without using the reduction to linear matroid parity, Chudnovsky, Cunningham, and Geelen [3] devised a combinatorial algorithm that runs in $O(n^5)$ time. When applying these methods to the edge-disjoint T-paths problem, one has to deal with the line graph of G. Thus the best known randomized and deterministic running time bounds for the maximum edge-disjoint T-paths are $O(|E|^\omega)$ and $O(|E|^{\omega+2})$, respectively.

In this paper, we provide a combinatorial, deterministic algorithm for finding the maximum number of edge-disjoint T-paths. The algorithm adopts an augmenting path approach. More specifically, we introduce a novel concept of augmenting walks in auxiliary labeled graphs to capture a possible augmentation of the number of edge-disjoint T-paths. To design a search procedure for an augmenting walk, we introduce blossoms analogously to the matching algorithm of Edmonds [4], although the present problem is neither a special case nor a generalization of the matching problem. When the search procedure terminates without finding an augmenting walk, the algorithm provides a certificate for the optimality of the current edge-disjoint T-paths. Thus the correctness argument of the algorithm serves as an alternative direct proof of Mader's theorem on edge-disjoint T-paths. The algorithm runs in $O(|V| \cdot |E|^2)$ time. This is definitely faster than the above mentioned deterministic algorithm and is comparable with the randomized algebraic algorithm.

A natural generalization of the present setting is to think of finding a maximum edge-disjoint T-paths of minimum total cost, where the cost is defined to be the sum of the costs of the included edges. Karzanov [8] gave a min-max theorem and described a combinatorial algorithm for this problem. The detailed proof of correctness, given in an technical report of more than 60 pages [7], has remained unpublished. Keijsper, Pendavingh, and Stougie [9] provided a dual pair of linear programs whose optimal value coincides with the maximum number of edge-disjoint T-paths. Giving an efficient separation procedure for this linear program, they showed that one can find maximum edge-disjoint T-paths in polynomial time via the ellipsoid method. A recent paper of Hirai and Pap [6] dealt with a weighted maximization of edge-disjoint T-paths, where the weight is given by a metric on the terminal set T. They clarified that this problem with edge costs can be solved in polynomial time if the weight is given by a tree metric and that it is NP-hard otherwise. Mader's edge-disjoint T-paths problem corresponds to the case with a tree metric that comes from a star. They adopted a novel polyhedral approach to prove a min-max theorem that extends Mader's theorem on edge-disjoint T-paths. Their algorithm, however, depends on the ellipsoid method. Our algorithm may serve as a prototype of possible combinatorial algorithms for this generalization.

The rest of this paper is organized as follows. Section 2 describes the statement of Mader's theorem on edge-disjoint T-paths. In Section 3, we introduce augmenting walks in auxiliary labeled graphs, and provide a procedure to increase the number of edge-disjoint T-paths using an augmenting walk. Section 4 presents a procedure to find an augmenting walk, whose correctness is verified in Section 5. Finally in Section 6, we analyze the complexity of the whole algorithm.

2 Mader's theorem

For a multigraph $G = (V, E)$ and $T \subseteq V$, a collection $\mathcal{X}$ of mutually disjoint subsets $X_s \subseteq V$ indexed by $s \in T$ is called a T-subpartition if $X_s \cap T = \{s\}$ holds for each $s \in T$. For any $X \subseteq V$, we denote by $\delta(X)$ the set of edges in E between X and $V \setminus X$. We also denote $d(X) := |\delta(X)|$.

Since each T-path between $s, t \in T$ contains at least one edge in $\delta(X_s)$ and at least one edge in $\delta(X_t)$, the number of edge-disjoint T-paths is at most $\frac{1}{2}\sum_{s\in T} d(X_s)$. This is not a tight bound. A more detailed analysis, however, leads to a tighter upper bound as follows.

For a T-subpartition $\mathcal{X}$, let $G \setminus \mathcal{X}$ denote the graph obtained from G by deleting all the vertices in $\bigcup_{s\in T} X_s$ and incident edges. A connected component of $G \setminus \mathcal{X}$ is said to be odd if its vertex set K has odd $d(K)$. Then $\mathrm{odd}(G \setminus \mathcal{X})$ denotes the number of odd components in $G \setminus \mathcal{X}$.

Lemma 2.1 *The number of edge-disjoint T-paths in G is at most*

$$\kappa(\mathcal{X}) := \frac{1}{2}\left[\sum_{s\in T} d(X_s) - \mathrm{odd}(G \setminus \mathcal{X})\right]$$

for any T-subpartition $\mathcal{X}$.

Proof. Suppose that there are k edge-disjoint T-paths in G. Let $\mathcal{K}$ be the collection of vertex sets of connected components of $G \setminus \mathcal{X}$. A T-path contains an edge between two distinct components of $\mathcal{X}$ or passes through a member K of $\mathcal{K}$. In the latter case, the T-path must contain two edges in $\delta(K)$. Therefore, we have $k \leq d(\mathcal{X}) + \sum_{K\in\mathcal{K}} \left\lfloor \frac{d(K)}{2} \right\rfloor$, where $d(\mathcal{X})$ is the number of edges between distinct components of $\mathcal{X}$. Since $d(\mathcal{X}) = \frac{1}{2}\left[\sum_{s\in T} d(X_s) - \sum_{K\in\mathcal{K}} d(K)\right]$, this implies $k \leq \frac{1}{2}\left[\sum_{s\in T} d(X_s) - \sum_{K\in\mathcal{K}} \left(d(K) - 2\left\lfloor \frac{d(K)}{2} \right\rfloor\right)\right]$, and the right-hand side equals $\kappa(\mathcal{X})$. ■

Mader's edge-disjoint T-paths theorem asserts that this upper bound is tight.

Theorem 2.2 (Mader [12]) *The maximum number of edge-disjoint T-paths equals the minimum of $\kappa(\mathcal{X})$ among all the T-subpartitions $\mathcal{X}$.*

3 Augmentation

Given k edge-disjoint T-paths $\mathcal{P} = \{P_1, \ldots, P_k\}$ in a multigraph $G = (V, E)$ without selfloops, we intend to characterize when $k + 1$ edge-disjoint T-paths exist in G. We now define an auxiliary labeled graph $\mathcal{G}(\mathcal{P}) = ((V, E \cup L), \sigma_V, \sigma_E, \sigma_L)$, by adding selfloops to G and assigning symbols to edges and vertices as labels. For $j = 1, \ldots, k$, we attach a selfloop at each internal vertex of P_j. A vertex has multiple selfloops if it belongs to multiple T-paths. We denote by L the set of those selfloops. For an edge $e \in E \cup L$, we denote by ∂e the set of its end-vertices. If $e \in E$ with $\partial e = \{u, v\}$ belongs to a T-path P_j from s to t, and s, u, e, v, t appear in this order along P_j, we

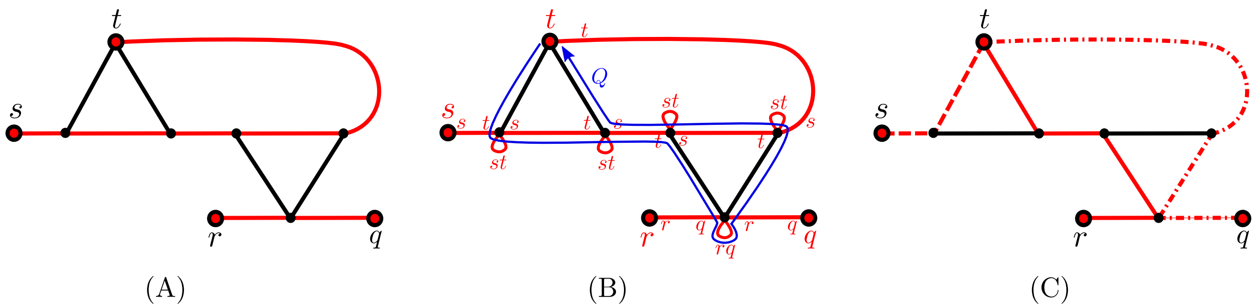

Figure 1: (A) A graph $G = (V, E)$ with terminals $T = \{s, t, r, q\}$. Red and black edges represent labeled and free edges, respectively, i.e., red edges form T-paths $\mathcal{P}$. (B) The auxiliary labeled graph $\mathcal{G}(\mathcal{P})$ and an augmenting walk Q, where $\gamma(Q) = tststrqtstst$. (C) Three edge-disjoint T-paths in G.

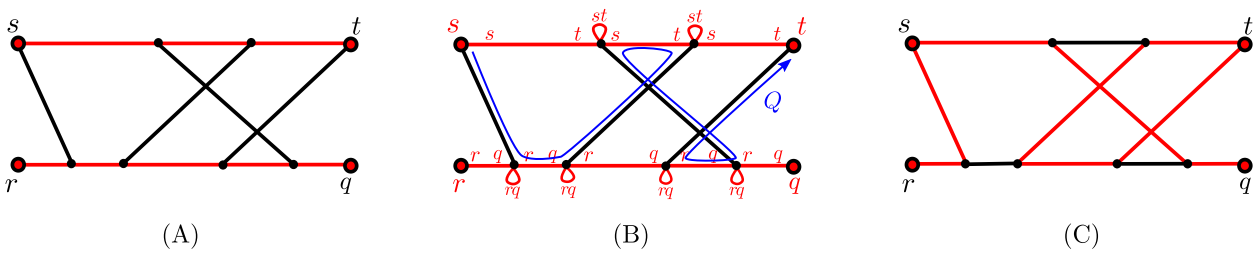

Figure 2: (A) A graph $G = (V, E)$ with terminals $T = \{s, t, r, q\}$, and T-paths $\mathcal{P}$. (B) The auxiliary labeled graph $\mathcal{G}(\mathcal{P})$ and an augmenting walk Q, where $\gamma(Q) = srqtsqrt$. (C) Red edges represent the symmetric difference of Q and the T-paths $\mathcal{P}$. This can not be decomposed into three edge-disjoint T-paths

assign symbols $\sigma_E(e, u) := s$ and $\sigma_E(e, v) := t$. Symbols are not assigned for edges in E that are not used in $\mathcal{P}$. An edge $e \in E$ is called *labeled* or *free* depending on whether it is assigned symbols or not. Each selfloop $e \in L$ that comes from a T-path with terminals s, t is assigned $\sigma_L(e) := st$. (One can choose st or ts arbitrarily. Once the order is chosen, it is fixed afterwards.) Furthermore, any terminal vertex $t \in T$ is assigned $\sigma_V(t) := t$, other vertices $v \in V \setminus T$ have no symbols.

A walk in $\mathcal{G}(\mathcal{P}) = ((V, E \cup L), \sigma_V, \sigma_E, \sigma_L)$ is a sequence $Q = (v_0, e_1, v_1, \ldots, e_\ell, v_\ell)$ of vertices $v_i \in V$ and edges $e_i \in E \cup L$ such that $\partial e_i = \{v_{i-1}, v_i\}$ for $i = 1, \ldots, \ell$. For a walk Q, we associate a string $\gamma(Q)$ to be the sequence of symbols that appear as labels in Q: If e_i is a labeled edge, then e_i is assigned $\sigma_E(e_i, v_{i-1})\sigma_E(e_i, v_i)$. If $e_i \in L$, then e_i is assigned $\sigma_L(e_i)$. If $v_i = t \in T$, then v_i is assigned $\sigma_V(v_i) = t$. Free edges and non-terminal vertices are assigned no symbols. We call Q an *augmenting walk* in $\mathcal{G}(\mathcal{P})$ if it satisfies the following three conditions:

(A1) $v_0, v_\ell \in T$ and all other vertices in Q are not in T. ($v_0 = v_\ell$ is allowed.)

(A2) $\gamma(Q)$ has no consecutive appearance of a symbol.

(A3) In Q, each free edge and selfloop appears at most once, and each labeled edge appears at most twice (at most once in each direction).

In Figure 1, we provide an example of an augmenting walk, which suggests the significance of selfloops and double use of labeled edges. In this case, we can augment the number of edge-disjoint T-paths by taking the symmetric difference between the augmenting walk and the union of the original T-paths. This simple operation does not always work. Figure 2 gives an example for which the symmetric difference gives an edge set that cannot be decomposed into edge-disjoint T-paths. By considering a more careful procedure, however, we obtain the following statement.

Proposition 3.1 *If there exists an augmenting walk in the labeled graph $\mathcal{G}(\mathcal{P})$, then there exist $k+1$ edge-disjoint T-paths in G.*

We provide an augmentation procedure to construct $k+1$ edge-disjoint T-paths using an augmenting walk. Without loss of generality, we can suppose that we have an augmenting walk Q that has no *redundant* selfloops, where a selfloop e is called redundant if deleting e from Q preserves (A2). We introduce some notations needed to describe the procedure.

For distinct vertices u, v in a T-path $P_j \in \mathcal{P}$, we denote by $P_j(u, v)$ the subpath of P_j from u to v. We also denote by $P_j(u, u)$ the selfloop at u that comes from P_j. Let s and t be the end-vertices of P_j. The subpath $P_j(u, v)$ is called st-directed if $\gamma(P_j(u, v))$ starts with s and ends with t. Otherwise, it is called ts-directed.

For a walk $Q = (v_0, e_1, v_1, \ldots, e_\ell, v_\ell)$ and indices a, b with $0 \leq a < b \leq \ell$, we denote by $Q[a, b]$ its subsequence $(v_a, e_{a+1}, \ldots, e_b, v_b)$. We call $Q[a, b]$ a P_j*-segment* of Q if all the edges in $Q[a, b]$ are edges or selfloops of P_j and e_a, e_{b+1} are not. Because Q has no redundant selfloops, any P_j-segment coincides with some subpath of P_j or some selfloop on P_j. An P_j-segment is st- or ts- directed if the corresponding $P_j(u, v)$ is so. We call $Q[a, b]$ an $\mathcal{P}$-segment if it is an P_j-segment for some $P_j \in \mathcal{P}$. We denote by $\mu_{\mathcal{P}}(Q)$ the number of $\mathcal{P}$-segments in Q.

Lemma 3.2 *If there is an augmenting walk Q with $\mu_{\mathcal{P}}(Q) = 0$ in $\mathcal{G}(\mathcal{P})$, then Q includes a T-path that is edge-disjoint from the T-paths in $\mathcal{P}$. Hence, there are $k+1$ edge-disjoint T-paths.*

Proof. As $\mu_{\mathcal{P}}(Q) = 0$, all edges in Q are free edges. By (A1), then $\gamma(Q)$ contains only the symbols assigned to the first and last vertices, which are distinct by (A2). Therefore, the walk Q includes a T-path. As it contains only free edges, it is edge-disjoint from the T-paths in $\mathcal{P}$. ∎

Our procedure repeatedly updates $\mathcal{P}$ and Q to decrease $\mu_{\mathcal{P}}(Q)$. We introduce two operations.

The first one is a *shortcut operation* for an augmenting walk Q. Suppose that $Q[a, b]$ is an st-directed P_j-segment and $Q[c, d]$ is another P_j-segment with $b < c$ (whose direction is not specified). A shortcut operation for Q is applicable to such $Q[a, b]$ and $Q[c, d]$ if

- v_d is on the subpath $P_j(v_a, t)$,
- the first symbol in $Q[d, \ell]$ is not t, and
- there is no P_j-segment in $Q[0, a]$ or $Q[d, \ell]$ that includes $P_j(v_a, v_d)$ or its subpath.

The shortcut operation replaces $Q[a, d]$ in Q with the subpath $P_j(v_a, v_d)$ if $v_a \neq v_d$. In case $v_a = v_d$, the shortcut operation replaces $Q[a, d]$ with the selfloop $P_j(v_a, v_a)$ if the last symbol in $Q[0, a]$ and the first symbol in $Q[d, \ell]$ are the same, and otherwise it just removes $(e_{a+1}, \ldots, e_d, v_d)$. By this definition, we have the following observation.

Observation 3.3 *An applicable shortcut operation for an augmenting walk Q yields another augmenting walk Q' with $\mu_{\mathcal{P}}(Q') < \mu_{\mathcal{P}}(Q)$.*

The next one is *uncrossing operation*, which is applied to a (non-augmenting) walk that contains repeated subsequences. Let $Q[a, b]$ and $Q[c, d]$ be repeated subsequences in Q, i.e., $b \leq c$ and they consists of the same edges and vertices in the same order. The uncrossing operation for these repeated subsequences is to replace the subsequence $Q[a, d]$ with the inversion of $Q[b, c]$. Then, $Q[a, b]$ and $Q[c, d]$ are removed and the interval between them are attached with reversed order. The obtained sequence Q' is again a walk as $v_a = v_c$ and $v_b = v_d$.

Using these operations, we describe Procedure Augment to augment T-paths. For walks Q_1 and Q_2 such that the last vertex of Q_1 and the first vertex of Q_2 are the same, we denote by $Q_1 + Q_2$ the walk obtained by connecting them. The complexity of Procedure Augment will be analyzed in Section 6.

Procedure Augment$(\mathcal{P}, Q)$

Input: k edge-disjoint T-paths $\mathcal{P} = \{P_1, \dots, P_k\}$ in G and an augmenting walk Q in $\mathcal{G}(\mathcal{P})$.
Output: $k+1$ edge-disjoint T-paths in G.

1. If $\mu_{\mathcal{P}}(Q) = 0$, then take a T-path R included in Q and return $\mathcal{P} \cup \{R\}$.
2. Take $P_j \in \mathcal{P}$ and $s, t \in T$ such that the first $\mathcal{P}$-segment in Q is an st-directed P_j-segment.
   (a) If Q contains a pair of P_j-segments to which a shortcut is applicable, then update Q by applying the shortcut operation and go back to 1.
   (b) Otherwise, let $Q[a, b]$ be the first st-directed P_j-segment in Q and do the following:
   - Let P'_j be a path included in $Q[0, a] + P_j(v_a, s)$.
   - Let $Q' := P_j(t, v_b) + Q[b, \ell]$ (v_ℓ is the last vertex of Q). While there exists a subpath of $P_j(t, v_b)$ that appears twice in Q', take the maximal one that is the closest to v_b on $P_j(t, v_b)$ and update Q' by uncrossing the corresponding repeated subsequences.

   Update $\mathcal{P}$ by replacing P_j with P'_j, update Q to Q', and go back to 1.

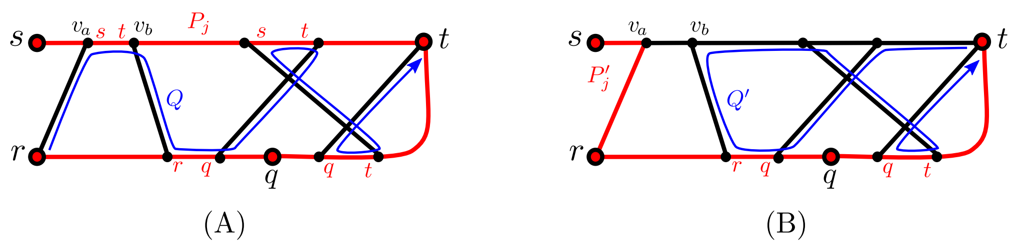

Figure 3: A demonstration of Step 2 (b) of Procedure Augment. The figure (A) represents the T-paths and the augmenting walk just before Step 2 (b), and (B) represents those just after Step 2 (b).

To show Proposition 3.1, we prove the correctness of Procedure Augment. As we have Lemma 3.2 and Observation 3.3, it suffices to show that Step 2 (b) decreases $\mu_{\mathcal{P}}(Q)$ preserving that $\mathcal{P}$ consists of edge-disjoint T-paths and Q is an augmenting walk (Lemma 3.5). We prepare the following lemma.

Lemma 3.4 *For an augmenting walk Q and a T-path P_j connecting $s, t \in T$, there is no pair of P_j-segments to which a shortcut operation is applicable if and only if the following conditions hold:*

(i) If P_j-segments $Q[a, b]$ and $Q[c, d]$ with $b < c$ are both st-directed (resp., both ts-directed), then $v_a \neq v_d$ and $s, Q[c, d], Q[a, b], t$ (resp., $t, Q[c, d], Q[a, b], s$) appears on P_j in this order.

(ii) If P_j-segments $Q[a, b]$ and $Q[c, d]$ with $b < c$ are st-directed and ts-directed, respectively (resp., ts-directed and st-directed, respectively) and v_d is on $P_j(v_a, t)$ (resp., on $P_j(v_a, s)$), then the first symbol in $Q[d, \ell]$ is t (resp., is s).

Proof. Suppose that there are P_j-segments $Q[a, b]$ and $Q[c, d]$ with $b < c$ to which a shortcut is applicable. Without loss of generality, let $Q[a, b]$ be st-directed. By the definition of an applicable shortcut, v_d is on $P_j(v_a, t)$ and the first symbol in $Q[d, \ell]$ is not t. Then (i) is violated if $Q[c, d]$ is st-directed, and otherwise (ii) is violated. Thus, the "if" part is shown. To see the other direction, suppose that (i) or (ii) fails. Among pairs of P_j-segments violating (i) or (ii), let $Q[a, b]$ and $Q[c, d]$ the one that minimizes the length of $P_j(v_a, v_d)$. To them, shortcut operation is applicable. ∎

Lemma 3.5 *When the procedure updates $\mathcal{P}$ and Q at Step 2 (b), $\mathcal{P}$ remains to consist of k edge-disjoint T-paths, Q remains to be an augmenting walk, and $\mu_{\mathcal{P}}(Q)$ becomes smaller than before.*

Proof. Because all edges in $Q[0,a]$ are free, only the initial vertex $v_0 \in T$ has a symbol in $Q[0,a]$. As $Q[a,b]$ is st-directed, we have $v_0 \neq s$ by (A2). Then, $Q[0,a] + P_j(v_a, s)$ includes a T-path P'_j with distinct terminals v_0 and s. As $Q[0,a]$ and P_j are disjoint from the T-paths in $\mathcal{P} \setminus \{P_j\}$, $\mathcal{P}' := \mathcal{P} \setminus \{P_j\} \cup \{P'_j\}$ consists of k edge-disjoint T-paths.

We now show that Q' defined in Step 2 (b) is indeed an augmenting walk in $\mathcal{G}(\mathcal{P}')$. By the algorithm, when Step 2 (b) is applied, there is no pair of P_j-segments to which a shortcut is applicable. Hence we have (i) and (ii) in Lemma 3.4.

Let $Q^* := P_j(t, v_b) + Q[b,\ell]$, i.e., Q' before uncrossing operations. We use $\gamma(\cdot)$ and $\gamma'(\cdot)$ to denote the string of a walk in $\mathcal{G}(\mathcal{P})$ and $\mathcal{G}(\mathcal{P}')$, respectively. Note that $\gamma'(P_j(t,v_b)) = t$ because all edges on $P_j(t,v_b)$ have no symbols in $\mathcal{G}(\mathcal{P}')$. Then $\gamma'(Q^*) = t \cdot \gamma'(Q[b,\ell])$.

We first show that $\gamma'(Q^*) = t \cdot \gamma'(Q[b,\ell])$ satisfies (A2). Since $Q[a,b]$ is st-directed, the first symbol in $Q[b,\ell]$ is not t, which implies together with (A2) for $\gamma(Q)$ that $t \cdot \gamma(Q[b,\ell])$ satisfies (A2). By replacing P_j with P'_j, we newly assign symbols to the edges in $P'_j \setminus P_j$, while we delete symbols and selfloops on $P_j(v_a, t)$. The new symbols on $P'_j \setminus P_j$ do not change $\gamma(Q[b,\ell])$ because those edges are not used in $Q[b,\ell]$. On the other hand, deletion of symbols and selfloops changes $\gamma(Q[b,\ell])$. To see whether (A2) is maintained, it suffices to care about every P_j-segment $Q[c,d]$ in $Q[b,\ell]$ whose symbols are all deleted. By (i) of Lemma 3.4, such $Q[c,d]$ is ts-directed. Hence the last symbol in $\gamma(Q[0,c])$ is not t and the first symbol in $\gamma(Q[d,\ell])$ is t by Lemma 3.4 (ii). Therefore, deletion of symbols on $Q[c,d]$ preserves (A2). Thus, $\gamma'(Q^*) = t \cdot \gamma'(Q[b,\ell])$ satisfies (A2).

We can also observe that $Q^* := P_j(t,v_b) + Q[b,\ell]$ satisfies (A1) and the conditions in (A3) for labeled edges and selfloops. However, the condition in (A3) for free edges may be violated in $\mathcal{G}(\mathcal{P}')$ because the edges in $P_j(t,v_b)$, which are free in $\mathcal{G}(\mathcal{P}')$, may appear twice in Q^*: first in $P_j(t,v_b)$ and secondly in $Q[b,\ell]$. We now intend to prove that Step 2 (b) deletes such invalid duplications. By Lemma 3.4 (i), all P_j-segments in $Q[b,\ell]$ that contain edges in $P_j(t,v_b)$ are ts-directed and mutually vertex-disjoint. Among subpaths of $P_j(t,v_b)$ that appear twice Q', take a maximal one that is the closest to v_b on $P_j(t,v_b)$. Let $Q^*[c,d]$ and $Q^*[g,h]$ be its first and second appearance in Q'. We apply uncrossing operation to them. As $P_j(t,v_b)$ has no symbols, the last symbol in $Q^*[0,c]$ is the terminal t and the first symbol in $Q^*[d,\ell']$ is not t, where ℓ' is the index of the last vertex of Q^*. Since $Q^*[g,h]$ corresponds to a ts-directed P_j-segment of Q (or its subsequent), the last symbol of $Q^*[0,e]$ is not t. The first symbol in $Q^*[f,\ell']$ is t by Lemma 3.4 (ii). These imply that after uncrossing operation, $\gamma'(Q^*)$ still satisfies (A2), while $Q^*[c,d]$ and $Q^*[g,h]$ are removed. By applying uncrossing operation to all repeated subpaths of $P_j(t,v_b)$ from v_b-side to t-side, we can delete all invalid duplications preserving (A2). (Note that by Lemma 3.4 (i) these subpaths are arranged on P_j from v_b-side to t-side corresponding to the order of appearance in Q^*.) Thus, we finally obtain an augmenting walk Q' with $\mu_{\mathcal{P}'}(Q') < \mu_{\mathcal{P}}(Q)$. ■

4 Search for an Augmenting Walk

As we have seen, we can increase the number of T-paths if we find an augmenting walk in the labeled graph $\mathcal{G}(\mathcal{P})$. We now design a procedure to find an augmenting walk by extending a search forest from the root set T. When a cyclic structure with a particular condition is found, the procedure shrinks it and applies a recursion to the resulting smaller graph. Terminal vertices are not involved in shrinking, and hence the vertex set includes T at any point of the procedure.

In a recursive call, the procedure is given a labeled graph $\mathcal{G} = ((V, E \cup L), \sigma_V, \sigma_E, \sigma_L)$ and a forest F, which is a subgraph of $(V, E \cup L)$ and includes T as a root set. That is, F is a collection of vertex-disjoint $|T|$ trees, each of which is rooted at some $t \in T$. (A single vertex is also regarded as a tree.) The forest F represents the history of a search process. We denote by $V(F)$ the vertex set of F. For each $v \in V(F)$, there is a unique path from some vertex in T to v. We denote this path by $P_F(v)$. If $v \in T$, then $P_F(v)$ is a single vertex. Every vertex in $P_F(v)$ is called an *ancestor* of v. For an ancestor u of v, we denote by $P_F(u,v)$ the subwalk of $P_F(v)$ from u to v. We write $\overline{P_F(v)}$ and $\overline{P_F(u,v)}$ for the reversed sequences of $P_F(v)$ and $P_F(u,v)$, respectively.

In the given labeled graph $\mathcal{G}$, symbols are assigned to some edges in E, all selfloops in L, all terminals in T, as before. In addition, a symbol $*$ is assigned to all pseudo-vertices that have been created by shrinking operations. For each $v \in V(F) \setminus T$, the last edge of the path $P_F(v)$ is denoted by $\mathsf{stalk}_F(v) \in E$. For each $v \in V(F)$, the last symbol appearing in $P_F(v)$ is denoted by $\mathsf{mark}_F(v) \in T \cup \{*\}$.

We now extend the definition of an augmenting walk for a labeled graph that may contain pseudo-vertices. For a pair $(\mathcal{G}, F)$ of a labeled graph $\mathcal{G}$ and a forest F in $\mathcal{G}$, we say that a walk $Q = (v_0, e_1, v_1, \ldots, e_\ell, v_\ell)$ in $(V, E \cup L)$ is a *shrunk augmenting walk* in $(\mathcal{G}, F)$ if Q satisfies (A1), (A3), and an extended version of (A2):

(A2') $\gamma(Q)$ has no consecutive appearance of a symbol of T (while $*$ is allowed to repeat),

and additional two conditions below:

(A4) Every pseudo-vertex appears at most once. If v_i is a pseudo-vertex, $\mathsf{stalk}_F(v_i) \in \{e_i, e_{i+1}\}$.

(A5) For every vertex v_i with $i \in \{1, \ldots, \ell-1\}$, if $\mathsf{stalk}_F(v_i) \notin \{e_i, e_{i+1}\}$, then $\mathsf{mark}_F(v_i)$ belongs to T and coincides with either the last symbol in $Q[0,i]$ or the first symbol in $Q[i,\ell]$.

Note that, if Q is a shrunk augmenting walk, then so is the reversed sequence of Q. As a shrunk augmenting walk satisfies (A1), (A2'), and (A3), we can observe the following.

Observation 4.1 *If a labeled graph $\mathcal{G}$ contains no pseudo-vertex, then for any forest F, a shrunk augmenting walk in $(\mathcal{G}, F)$ is an augmenting walk in $\mathcal{G}$.*

To find a shrunk augmenting walk, our procedure extends the forest F preserving its admissibility, where a forest F in $\mathcal{G}$ with root set T is called *admissible* if the following conditions hold:

(F1) For every $v \in V(F)$, the string $\gamma(P_F(v))$ has no consecutive appearance of a symbol of T.

(F2) For every pseudo-vertex v in $\mathcal{G}$, we have $v \in V(F) \setminus T$ and $\mathsf{stalk}_F(v)$ is a free edge.

For a labeled graph $\mathcal{G}$ and an admissible forest F, we call an edge $e \in E$ a *frontier edge* if $\partial e = \{u, v\}$ with $u \in V(F)$ and $v \notin V(F)$, and either

- e is free, or
- e is labeled and $\mathsf{mark}_F(u) \neq \sigma_E(e, u)$.

The following statement is easily observed.

Observation 4.2 *For a graph $\mathcal{G}$ and an admissible forest F, suppose that $e \in E$ is a frontier edge such that $\partial e = \{u, v\}$, $u \in V(F)$, and $v \notin V(F)$. Then, the forest F' obtained by adding e and v to F is also an admissible forest.*

We call an edge $e \in E \cup L$ that is not in F an *interior edge* if $\partial e = \{u, v\} \subseteq V(F)$ and either

- $e \in E$, e is labeled, $\mathsf{mark}_F(u) \neq \sigma_E(e, u)$, and $\mathsf{mark}_F(v) \neq \sigma_E(e, v)$,
- $e \in E$, e is free, and either $\mathsf{mark}_F(u) \neq \mathsf{mark}_F(v)$ or $\mathsf{mark}_F(u) = \mathsf{mark}_F(v) = *$,
- $e \in L$, $\mathsf{mark}_F(u) \notin \{s, t\}$, where $st = \sigma_L(e)$.

Let $\mathcal{G}$ be a labeled graph and F be an admissible forest in $\mathcal{G}$. For an interior edge e with $\partial e = \{u, v\}$, we say that *e produces a shrunk augmenting walk* if $P_F(v)$ and $P_F(u)$ have no common free edge, because we have the following proposition. We use a symbol $\cdot$ to represent concatenation.

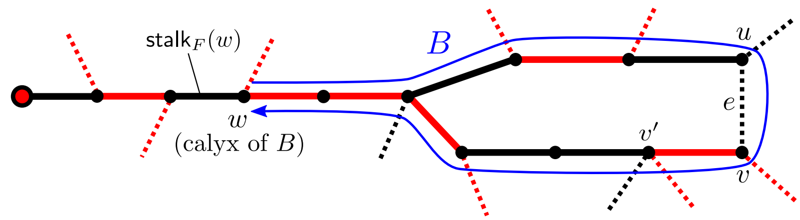

Figure 4: An example of a blossom structure. Thick edges represent a part of the search forest F, where red and black edges are corresponding to labeled and free edges, respectively.

Lemma 4.3 *For a graph $\mathcal{G}$ and an admissible forest F, suppose that $e \in E \cup L$ is an interior edge with $\partial e = \{u, v\}$ and $P_F(u)$ and $P_F(v)$ have no common free edge. Then, $P_F(u) \cdot e \cdot \overline{P_F(v)}$ is a shrunk augmenting walk in $(\mathcal{G}, F)$.*

Proof. Let $Q := P_F(u) \cdot e \cdot \overline{P_F(v)} = (v_0, e_1, v_1, \ldots, e_\ell, v_\ell)$. By this definition, for $i = 1, 2, \ldots, \ell - 1$, the vertex v_i satisfies $\mathsf{stalk}_F(v_i) \in \{e_i, e_{i+1}\}$, and hence (A5) follows. The condition (A1) is clear. The condition (A2') follows from (F1) of F and the definition of an interior edge. The condition (A3) follows from the assumption that $P_F(u)$ and $P_F(v)$ have no common free edge. To see (A4), suppose, to the contrary, that some pseudo-vertex v' appears more than once in Q, which means that v' is a common ancestor of u and v. Then $\mathsf{stalk}_F(v')$ is contained in both $P_F(u)$ and $P_F(v)$, while $\mathsf{stalk}_F(v')$ is a free edge as F satisfies (F2), which contradicts the assumption. ■

If an interior edge e with $\partial e = \{u, v\}$ does not produce a shrunk augmenting walk, it means that $P_F(u)$ and $P_F(v)$ have a common free edge. Take the common ancestor w of u and v such that $\mathsf{stalk}_F(w)$ is the last common free edge in $P_F(u)$ and $P_F(v)$. We then say that e *produces a blossom* B, where $B := P_F(w, u) \cdot e \cdot \overline{P_F(w, v)}$, and call w the *calyx* of B. (See Figure 4.) We denote by $V(B)$ the set of vertices in B. Note that $V(B)$ can be a singleton set if $u = v = w$ and e is a selfloop at u.

We denote by $\mathcal{G}/B$ the graph obtained by shrinking B to one new vertex, called a pseudo-vertex B: The vertex set of $\mathcal{G}/B$ is $V' := (V \setminus V(B)) \cup \{B\}$, and $\sigma_{V'}$ is the same as σ_V on $V \setminus V(B)$ and $\sigma_{V'}(B) = *$. For each $e \in E \cup L$ in $\mathcal{G}$, we replace its end-vertices in $V(B)$ with the new vertex B and denote the new edge again by e. We then remove all selfloops at B. Denote by E' and L' the resultant sets of edges and selfloops. For an edge e whose end-vertices are modified from $\{\hat{u}, \hat{v}\}$ to $\{B, \hat{v}\}$, we set $\sigma_{E'}(e, B) = \sigma_E(e, \hat{u})$. For other edges and selfloops, $\sigma_{E'}$ and $\sigma_{L'}$ are the same as σ_E and σ_L, respectively. An edge or a vertex in $\mathcal{G}/B$ is called a *projection* of the corresponding edge or vertex in $\mathcal{G}$. A pseudo-vertex B can be a projection of multiple vertices, while other vertices and edges have one-to-one correspondence. The forest F/B consists of vertices and edges in $\mathcal{G}/B$ that are the projections of the vertices and edges of F. Note that $\mathsf{stalk}_{F/B}(B)$ in $\mathcal{G}/B$ is the projection of $\mathsf{stalk}_F(w)$ in $\mathcal{G}$.

Suppose that a shrunk augmenting walk $Q' = (v_0, e_1, v_1, \ldots, e_\ell, v_\ell)$ in $(\mathcal{G}/B, F/B)$ contains the pseudo-vertex B as v_i. By (A4), we have $\mathsf{stalk}_{F/B}(v_i) \in \{e_i, e_{i+1}\}$. Without loss of generality, suppose $e_i = \mathsf{stalk}_{F/B}(v_i) = \mathsf{stalk}_{F/B}(B)$. Then, e_i is the projection of $\mathsf{stalk}_F(w)$, and e_{i+1} is the projection of some edge incident to some $v' \in V(B)$. Then v' belongs to $P_F(u)$ or $P_F(v)$. W.l.o.g., let v' belong to $P_F(v)$. The *expanding operation* replaces $v_i (= B)$ in Q' with the subsequence of the blossom B as follows. Note that B appears in Q' only once by (A4).

Expansion: (i) If $\mathsf{mark}_F(v') = *$ or $\mathsf{mark}_F(v')$ is different from the first symbol in $\overline{Q[i, \ell]}$, then we replace $v_i (= B)$ with $P_F(w, v')$. (ii) Otherwise, we replace v_i with $P_F(w, u) \cdot e \cdot \overline{P_F(v', v)}$.

Using these shrinking and expanding operations, we can describe the following procedure to find an augmenting walk in the labeled graph $\mathcal{G}$. When we first call the procedure, the input is set as $\mathcal{G} := \mathcal{G}(\mathcal{P})$ and $F := (T, \emptyset)$.

Procedure $\mathsf{Search}(\mathcal{G}, F)$

Input: A labeled graph $\mathcal{G} = ((V, E \cup L), \sigma_V, \sigma_E, \sigma_L)$ and an admissible forest F in $\mathcal{G}$.
Output: A shrunk augmenting walk in $(\mathcal{G}, F)$.

1. If there exists neither a frontier edge nor an interior edge, then terminate.
2. If there is a frontier edge e with end-vertices $u \in V(F)$ and $v \notin V(F)$, add e and v to F and go back to 1.
3. If there is an interior edge e, then do the following:
   (a) If e produces a shrunk augmenting walk Q, then return Q.
   (b) If e produces a blossom B, then call $\mathsf{Search}(\mathcal{G}/B, F/B)$ to obtain a shrunk augmenting walk Q. If Q contains B, then expand B in Q. Return the resultant walk Q.

---

5 Correctness of the Search Procedure

As we have Observations 4.1, 4.2 and Lemma 4.3, to prove the correctness of Procedure Search, it suffices to show the following two: (1) the correctness of the shrinking and expanding operations, (2) the maximality of the number of T-paths when the procedure terminates without returning a walk. We show these two as Propositions 5.1 and 5.2, respectively.

Proposition 5.1 *For a graph $\mathcal{G}$ and an admissible forest F, suppose that an interior edge $e \in E \cup L$ with $\partial e = \{u, v\}$ produces a blossom B with calyx w. Then,*

- *F/B is an admissible forest in $\mathcal{G}/B$, and*
- *If $(\mathcal{G}/B, F/B)$ admits a shrunk augmenting walk, then so does $(\mathcal{G}, F)$.*

Proof. First, note that we can observe the following conditions for $\mathsf{stalk}_{F/B}$ and $\mathsf{mark}_{F/B}$:

($\star$) For any $\hat{v} \in V \setminus V(B)$, the edge $\mathsf{stalk}_{F/B}(\hat{v})$ is a projection of $\mathsf{stalk}_F(\hat{v})$. Also, $\mathsf{mark}_{F/B}(\hat{v})$ and $\mathsf{mark}_F(\hat{v})$ differ only if $\mathsf{mark}_{F/B}(\hat{v})$ is $*$ that is assigned to B.

Because F satisfies (F1) and the pseudo-vertex B is assigned $*$, for each $\hat{v} \in V(F/B)$, the path $P_{F/B}(\hat{v})$ has no consecutive appearance of a symbol in T. Also, as $\mathsf{stalk}_{F/B}(B)$ is the projection of $\mathsf{stalk}_F(w)$, the choice of w implies that $\mathsf{stalk}_{F/B}(B)$ is a free edge. Also, $\mathsf{stalk}_{F/B}(\hat{v})$ of each $\hat{v} \in V \setminus V(B)$ is the projection of $\mathsf{stalk}_F(\hat{v})$ by ($\star$). Thus, F/B satisfies (F1) and (F2).

Let $Q' = (v_0, e_1, v_1, \ldots, e_\ell, v_\ell)$ be a shrunk augmenting walk in $(\mathcal{G}/B, F/B)$. If Q' does not contain B, then Q' is a shrunk augmenting walk also in $(\mathcal{G}, F)$, where (A5) follows from ($\star$).

If Q' contains B as v_i, obtain a walk Q in $(\mathcal{G}, F)$ by expanding $B = v_i$. W.l.o.g., we assume $e_i = \mathsf{stalk}_{F/B}(v_i)$, e_{i+1} is a projection of an edge incident to $v' \in V(B)$, and v' is on the path $P_F(v)$ (as Figure 4). Clearly Q satisfies (A1). To see (A2') and (A3) for Q, note that Q' and B satisfy these two conditions, where (A3) of the blossom B follows from the choice of w. As all edges in Q' are not spanned by $V(B)$, the walks Q' and B are edge-disjoint, and so Q satisfies (A3). As $e_i = \mathsf{stalk}_{F/B}(v_i)$ implies $e_i \neq \mathsf{stalk}_{F/B}(v_{i-1})$, by applying (A5) to v_{i-1}, we have either (a) $e_{i-1} = \mathsf{stalk}_{F/B}(v_{i-1})$ or (b) $\mathsf{mark}_{F/B}(v_{i-1})$ belongs to T and coincides with the last symbol in $Q'[0, i-1]$ (note that the first symbol in $Q'[i-1, \ell]$ is $*$ as $v_i = B$). In case (a), apply (A5) to v_{i-2}, and so on. Then in any case, we have that the last symbol in $Q'[0, i-1]$ coincides with $\mathsf{mark}_{F/B}(v_{i-1})$. This implies that expanding v_i does not cause a consecutive appearance of a symbol in T. Thus Q satisfies (A2').

To see that Q satisfies (A4) and (A5), it suffices to check the conditions for the inserted part, because Q' satisfies these two and $(\star)$ implies that they are maintained for any vertex in $V \setminus V(B)$. By the definition of expanding operation, only vertices in $P_F(w, w')$ may appear in Q twice, where w' is the nearest common ancestor of u and v. By the choice of w, for any vertex w'' on $P_F(w, w')$ except w, $\mathsf{stalk}_F(w'')$ is labeled. Then (F2) implies that w'' is not a pseudo-vertex. Also, w appears twice in Q only if $v' = w$ and case (ii) is applied in the expanding operation, which implies that w is not a pseudo-vertex. Thus, any pseudo-vertex appears in Q at most once. Also, we see that for any inserted vertex $\hat{v}$, except v' in case (ii), the edge just before or after $\hat{v}$ in Q is $\mathsf{stalk}_F(\hat{v})$. Also, in case (ii), v' is not a pseudo-vertex, and $\mathsf{mark}_F(v')$ and the first symbol in $Q'[i, \ell]$ are the same symbol in T. Thus, all vertices in Q in the inserted part satisfy (A4) and (A5). ∎

Proposition 5.2 *If Procedure* Search *terminates without returning a walk for $\mathcal{G}(\mathcal{P})$ and $(T, \emptyset)$, then $\mathcal{P} = \{P_1, \dots, P_k\}$ consists of the maximum number of edge-disjoint T-paths in $G = (V, E)$.*

Proof. The procedure terminates without returning a walk when there is neither a frontier edge nor an interior edge. Let $G' := (V', E' \cup L')$ be the underlying graph of $(\mathcal{G}, F)$ at such a point. For each $s \in T$, let X_s be the set of vertices $v \in V$ whose projections are vertices $v' \in V'$ with $\mathsf{mark}_F(v') = s$. Note that $s \in X_s$ holds for each $s \in T$, and thus $\mathcal{X} = (X_s)_{s \in T}$ forms a T-subpartition. We show the following three:

1. No free edge connects X_s and X_t with distinct $s, t \in T$.
2. If e is a labeled edge with end-vertices $u \in X_s$ and $v \notin X_s$, then $\sigma_E(e, u) = s$.
3. The projection of any connected component K of $G \setminus \mathcal{X}$ is a connected component K' in G' that is a subtree of F or disjoint from F. In the former case, exactly one free edge connects $\bigcup_{s \in T} X_s$ and K. In the latter case, there is no such a free edge.

The first condition is clear by the nonexistence of an interior edge. To show the second condition, suppose conversely that $\sigma_E(e, u) \neq s$. In case $\sigma_E(e, v) = s$, the edge e is an interior edge, a contradiction. In the other case, i.e., when $\sigma_E(e, v) \neq s$, the vertex $u \in X_s$ has a selfloop with two symbols distinct from s, which is an interior edge, a contradiction.

For the third condition, note that the projection of every vertex v in $G \setminus \mathcal{X}$ is some vertex v' in G' that satisfies either $\mathsf{mark}_F(v') = *$ or $v' \notin V'(F)$. Since any edge connecting v' with $\mathsf{mark}_F(v') = *$ and v'' with $v'' \notin V'(F)$ is a frontier edge, all the vertices in K' are the same type.

In the case that all vertices v' in K' satisfy $\mathsf{mark}_F(v') = *$, K' should be a subtree of F, because otherwise we have an interior edge spanned by K'. Hence there exists a vertex v^* that is the root of K'. Since the parent of v^* belongs to the projection of $\bigcup_{s \in T} X_s$, $\mathsf{mark}_F(v^*) = *$ implies that v^* is a pseudo-vertex. Then $\mathsf{stalk}_F(v^*)$ is a free edge by (F2). Also, any other edge connecting K' and the projection of $\bigcup_{s \in T} X_s$ should be a labeled edge since otherwise it is an interior edge.

In the case that all vertices v' in K' satisfy $v' \notin V'(F)$, there is no free edge connecting K' and the projection of $\bigcup_{s \in T} X_s$, since otherwise we have a frontier edge.

Using the above three conditions, we show $k = \frac{1}{2}\left[\sum_{s \in T} d(X_s) - \mathrm{odd}(G \setminus \mathcal{X})\right]$, which guarantees the maximality of the number of T-paths by Lemma 2.1. In $\sum_{s \in T} d(X_s)$, every T-path is counted exactly twice by the second condition. This implies that labeled edges are counted $2k$ times in total. Also, free edges are counted $\mathrm{odd}(G \setminus \mathcal{X})$ times by the first and third conditions. Thus, we have $\sum_{s \in T} d(X_s) = 2k + \mathrm{odd}(G \setminus \mathcal{X})$, which is equivalent to the required equality. ∎

6 Complexity

We analyze the complexity to find maximum edge-disjoint T-paths in a multigraph $G = (V, E)$. To construct maximum edge-disjoint T-paths, we start with $\mathcal{P} = \emptyset$ and add T-paths one by one by repeating Procedures Search and Augment. A formal description is given as Algorithm EDTP below. We have to call these procedures $O(|E|)$ times, because the number of edge-disjoint T-paths is $O(|E|)$. To guarantee that the entire time complexity of Algorithm EDTP is $O(|V| \cdot |E|^2)$, we show that each procedure can be computed in $O(|V| \cdot |E|)$ time.

6.1 Complexity for Search

We first give the following easy observation.

Lemma 6.1 *The number of selfloops in $\mathcal{G}(\mathcal{P})$ is at most $|E|$.*

Proof. The number of selfloops that come from a T-path P_j is one less than the number of edges in P_j. Because T-paths in $\mathcal{P}$ are edge-disjoint, their total length is at most $|E|$, and so is the number of selfloops in $\mathcal{G}(\mathcal{P})$. ■

We now show the complexity of the search procedure. We also show that the number of $\mathcal{P}$-segments in the returned walk is $O(|V|)$, which will be used to show the complexity of the augmentation procedure.

Proposition 6.2 *Procedure* Search *applied to $\mathcal{G}(\mathcal{P})$ and $(T, \emptyset)$ runs in $O(|V| \cdot |E|)$ time. The output walk Q contains at most $2|V|$ vertices, and hence $\mu_{\mathcal{P}}(Q) = O(|V|)$.*

Proof. The vertex sets of the blossoms produced in the procedure has a laminar structure. Therefore, the number of shrinking operations applied in the procedure is at most $2|V|$. Each shrinking needs a contraction of the graph, which can be done in $O(|E|)$. Thus the complexity for the shrinking operations in the search procedure is $O(|V| \cdot |E|)$ in total.

Aside from shrinking, we have to consider a complexity for expanding the search forest. Let us call an edge $e \in E \cup L$ *available* if it is a frontier or interior edge. For each edge $e \in E \cup L$ incident to F, we have to check whether it is available or not. Note that $e \in E$ with $\partial e = \{u, v\}$ can turn from unavailable to available only if its end-vertex u or v is replaced by a pseudo-vertex. Hence, we have to check the availability of each $e \in E$ at most three times. Also note that a selfloop is deleted if its end-vertex turns to a pseudo-vertex. Then the availability of each selfloop is checked at most once. Thus, the total complexity to construct the search forest is $O(|E|)$.

Algorithm EDTP

Input: A multigraph $G = (V, E)$ and a set $T \subseteq V$ of terminals.

Output: A collection $\mathcal{P}$ of maximum edge-disjoint T-paths.

1. Set $\mathcal{P} := \emptyset$.
2. Construct the auxiliary labeled graph $\mathcal{G}(\mathcal{P})$.
3. Apply Procedure Search to $\mathcal{G} := \mathcal{G}(\mathcal{P})$ and $F := (T, \emptyset)$.
   (a) If an augmenting walk Q is returned, then apply Procedure Augment to $\mathcal{P}$ and Q. Update $\mathcal{P}$ by the T-paths returned by Procedure Augment and go back to 2.
   (b) Otherwise, return the current $\mathcal{P}$.

Finally, we show that any vertex appears in Q at most twice. When an augmenting walk Q is found in some recursive call, Q has the form $P_F(u) \cdot e \cdot \overline{P_F(v)}$. Hence, any vertex appears at most twice in Q, and also any pseudo-vertex appears at most once by (A4). This property is preserved when we expand any pseudo-vertex, because the set of inserted vertices are disjoint from other vertices in Q, and in the inserted part any vertex appears at most twice. ■

6.2 Complexity for Augmentation

In the augmentation procedure, an augmenting walk is repeatedly updated and its length is not monotone decreasing. Then, we use the following trivial bound for its length.

Lemma 6.3 *The length of any augmenting walk Q in $\mathcal{G}(\mathcal{P})$ is $O(|E|)$.*

Proof. By Lemma 6.1 and (A3), selfloops appear at most $|E|$ times in total. Also by (A3), the edges in E appear at most $2|E|$ times in total. Thus, the length of Q is $O(|E|)$. ■

Using Lemma 6.3 and the last claim of Proposition 6.2, we show the complexity of the augmentation procedure.

Proposition 6.4 *Given an augmenting walk Q returned by Procedure* Search*, Procedure* Augment *runs in $O(|V| \cdot |E|)$ time.*

Proof. Note that both Steps 2 (a) and (b) of Procedure Augment decreases $\mu_{\mathcal{P}}(Q)$ by at least one. Also, $\mu_{\mathcal{P}}(Q) = O(|V|)$ at the beginning of the algorithm by Proposition 6.2. Then, Steps 2 (a) and (b) are applied $O(|V|)$ times in total. Therefore, it suffices to show that each of the following operations can be done in $O(|E|)$ time: (1) checking whether $\mu_{\mathcal{P}}(Q) = 0$, (2) finding a T-path included in Q when $\mu_{\mathcal{P}}(Q) = 0$, (3) checking whether there is a pair of P_j-segments to which a shortcut is applicable, where P_j is one specified T-path, (4) applying the shortcut operation for a specified segment pair, and (5) updating P_j and Q in the manner in Step 2 (b). We can easily check that (1),(2),(4), and (5) can be done in $O(|E|)$ time because the length of Q is $O(|E|)$ by Lemma 6.3.

The task (3) can be done by the following steps (Steps 1–4). Recall that Lemma 3.4 characterizes the nonexistence of a pair of P_j-segments to which a shortcut is applicable.

1. By tracking $Q = (v_0, e_1, v_1, \ldots, e_\ell, v_\ell)$ from first to last, do the following: if e_i belongs to the x-th P_j-segment, set $p_{st}(e_i) = x$ or $p_{ts}(e_i) = x$ depending on whether it is st- or ts-directed. Note that $p_{st}(e)$, $p_{ts}(e)$ are defined only for edges or selfloops on P_j.
2. For every vertex on P_j, check whether it is incident to some $e, e' \in E \cup L$ with $p_{st}(e) < p_{st}(e')$. If so, return the $p_{st}(e)$-th and $p_{st}(e')$-th P_j-segments. Check the same condition with s and t interchanged.

If no pair of segments is returned in this step, then all st-directed P_j-segments are mutually vertex-disjoint and all ts-directed P_j-segments are mutually vertex-disjoint.

3. By tracking the edges and selfloops on P_j from s-side to t-side, check whether the value $p_{st}(e)$ increases at some point on P_j. If such a increase happens, let x and y be the values just before and after the increase and return the x-th and y-th P_j-segments. Check the same condition with s and t interchanged.

If no pair of segments is returned in Steps 2 and 3, then the value $p_{st}(e)$ (resp., the value $p_{ts}(e)$) is monotone decreasing on P_j from s-side to t-side (resp., from t-side to s-side). This means that the condition (i) of Lemma 3.4 is satisfied.

4. See the values $p_{st}(e)$ and $p_{ts}(e)$ of every $e \in E \cup L$ on P_j from s-side to t-side while updating p^*_{st}, the last $p_{st}(e)$ value found so far. Whenever we find $e' \in E \cup L$ such that $p_{ts}(e') > p^*_{st}$, check whether the first symbol in Q after $p_{ts}(e')$-th P_j-segment is t. If it is not t, then return the p^*_{st}-th and $p_{ts}(e')$-th P_j-segments. Check the same condition with s and t interchanged.

If no pair of segments is returned in Steps 2,3, and 4, then the condition (ii) of Lemma 3.4 is also satisfied, and hence there is no pair of P_j-segments to which a shortcut is applicable.

We can also check that, if a pair of P_j-segments is returned in some step, a shortcut operation is applicable to the pair. We now check the complexity of Steps 1–4. Step 1 can be done in $O(|E|)$ time because the length of Q is $O(|E|)$. Steps 2 and 3 can be done in $O(|V|)$ as the length of P_j is $O(|V|)$. In Step 4, we have to track subsequences of Q to find the first symbol after the $p_{ts}(e)$-th P_j-segment. Because such a symbol is found until we reach the next P_j-segment, each part of Q is tracked at most once in Step 4. As the length of Q is $O(|E|)$, the total length of tracked sequences is $O(|E|)$. Thus, Step 4 can be done in $O(|V| + |E|) = O(|E|)$ time. ■

Acknowledgements

The authors thank Etsuko Ishii for her help in reading the German literature.

References

[1] B. V. Čerkasskiĭ: Reshenie odnoĭ zadachi o mnogoproduktovykh potokakh v seti, *Ékonomika i Matematicheskie Metody*, 13 (1977), pp. 143–151.

[2] H. Y. Cheung, L. C. Lau, and K. M. Leung: Algebraic algorithms for linear matroid parity problems, *ACM Trans. Algorithms*, 10 (2014), 10: 1–26.

[3] M. Chudnovsky, W. H. Cunningham, and J. Geelen: An algorithm for packing non-zero A-paths in group-labelled graphs, *Combinatorica*, 28 (2008), pp. 145–161.

[4] J. Edmonds: Paths, trees, and flowers, *Canadian J. Math.*, 17 (1965), pp. 449–467.

[5] H. N. Gabow and M. Stallmann: An augmenting path algorithm for linear matroid parity, *Combinatorica*, 6 (1986), pp. 123–150.

[6] H. Hirai and G. Pap: Tree metrics and edge-disjoint S-paths, *Math. Program.*, 147 (2014), pp. 81–123.

[7] A. V. Karzanov: Edge-disjoint T-paths of minimum total cost, Report No. STAN-CS-92-1465, Department of Computer Science, Stanford University, 1993.

[8] A. V. Karzanov: Multiflows and disjoint paths of minimum total cost, *Math. Program.*, 78 (1997), pp. 219–242.

[9] J. C. M. Keijsper, R. A. Pendavingh, and L. Stougie: A linear programming formulation of Mader's edge-disjoint paths problem, *J. Combin. Theory*, B96 (2006), pp. 159–163.

[10] L. Lovász: On some connectivity properties of Eulerian graphs, *Acta Math. Acad. Sci. Hungar.*, 28 (1976), pp. 129–138.

[11] L. Lovász: Matroid matching and some applications, *J. Combin. Theory*, B28 (1980), pp. 208–236.

[12] W. Mader: Über die Maximalzahl kantendisjunkter A-Wege, *Archiv. Math.*, 30 (1978), pp. 325–336.

[13] W. Mader: Über die Maximalzahl kreuzungsfreier H-Wege, *Archiv. Math.*, 31 (1978), pp. 387–402.

[14] J. B. Orlin: A fast, simpler algorithm for the matroid parity problem, *Proceedings of the 13th International Conference on Integer Programming and Combinatorial Optimization*, LNCS 5035, Springer-Verlag, 2008, pp. 240–258.

[15] J. B. Orlin and J. H. Vande Vate: Solving the linear matroid parity problem as a sequence of matroid intersection problems, *Math. Programming*, 47 (1990), pp. 81–106.

[16] A. Schrijver: A short proof of Mader's $\mathcal{S}$-paths theorem, *J. Combin. Theory*, B82 (2001), pp. 319–321.

[17] A. Schrijver: *Combinatorial Optimization — Polyhedra and Efficiency*, Springer-Verlag, 2003.

[18] S. Tanigawa and Y. Yamaguchi: Packing non-zero A-paths via matroid matching, *Discrete Appl. Math.*, 214 (2016), pp. 169–178.