Balanced decomposition of colored graphs

Amitava Bhattacharya, Pritam Majumder, Uri N Peled, Murali K. Srinivasan

March 2021

Abstract: Let $G(V, E, u, \ell, c)$ be a graph on a finite set of vertices V, E is a finite set of edges (multiple edges allowed but no self loops), u, ℓ are weight functions of the edges $u, \ell : E \to \mathbb{R}_{\geq 0}$ with $\ell(e) \leq u(e)$ for all edges $e \in E$ and c a coloring map that colors the edges with colors red or blue. In this paper we give a polynomial time algorithm to find a collection of closed alternating trails $\mathcal{T}$ and a weight function $w : \mathcal{T} \to \mathbb{R}_{\geq 0}$ such that

$$\ell(e) \leq \sum_{T \in \mathcal{T}} w(T)\chi_T(e) \leq u(e).$$

This is analogous to Hoffman circulation Theorem and Seymour's Sum of circuits.

1 Introduction

Let $D(V, E)$ be a directed graph with finite set of vertices $V(D)$, finite set of edge labels $E(D) = \{e_1, e_2, \dots e_m\}$ where each edge label corresponds to an element of $V \times V - \Delta$. We refer to the first coordinate of an edge as *start* and second coordinate as *end* vertex. Note that "multiple" edges are allowed in this definition.

A *network* is a quadruple $N(D, c, s, t)$, where $D = D(V, E)$ is a directed graph, *capacity* $c : E(D) \to \mathbb{Q}_{\geq 0}$, s a special vertex called *source* and t a special vertex called *sink*. By a *circuit* we mean $C = (v_0, e_1, v_1, e_2, v_2, \dots, e_m, v_0), \qquad m \geq 0$, where $v_i \in V$ is distinct for all i, $e_j = (v_{j-1}, v_j)$ for all $j < m$, $e_m = (v_{m-1}, v_0)$. For undirected graphs the $e_j = \{v_{j-1}, v_j\}$, for all $j < m$, $e_m = \{v_{m-1}, v_0\}$.

A flow is a mapping $f : E \to \mathbb{Q}^+$, denoted by f_{uv}, subject to the following two constraints:

1. *Capacity Constraint*: For every edge (u, v) in E, $f_{uv} \leq c_{uv}$,

2. *Conservation of Flows*: For each vertex v apart from s and t, the equality $\sum_{\{u:(u,v)\in E\}} f_{uv} = \sum_{\{w:(v,w)\in E\}} f_{vw}$ holds.

Let $\partial^+(U)$ be the set of directed edges leaving the set U. That is, the edges which have the start vertex in U and the end vertex outside U.

The *value* of a flow f is defined by

$$|f| = \sum_{e\in\partial^+(s)} f_e - \sum_{e\in\partial^+(V-s)} f_e.$$

The maximum flow problem asks to maximize $|f|$ on a given network, that is, to route as much flow as possible from s to t.

An $s-t$ cut $C=(S,T)$ is a partition of V such that $s\in S$ and $t\in T$. That is, $s-t$ cut is a division of the vertices of the network into two parts, with the source in one part and the sink in the other. The cut-set X_C of a cut C is the set of edges that connect the source part of the cut to the sink part:

$$X_C := \{(u,v)\in E \ : \ u\in S, v\in T\} = (S\times T)\cap E.$$

The capacity of an $s-t$ cut is the total weight of its edges,

$$c(S,T) = \sum_{e\in\partial^+(S)} c_e$$

It can be seen that the value of a flow is equal to the net flow out of an $s-t$ cut. This implies that the value of any flow f is always less than capacity of any $s-t$ cut.

Then the *max-flow min-cut theorem* assets that for the maximal flow the value of the flow is equal to the capacity of a $s-t$ cut. cut that is equal to the flow.

Theorem 1.1. *The maximum value of an $s-t$ flow is equal to the minimum capacity over all $s-t$ cuts.*

An equivalent form of the above theorem is the Hoffman Circulation Theorem. It may be stated as below.

Theorem 1.2. *(Hoffman) Let $D=(V,E)$ be a directed graph and let $\mathscr{C}$ be its collection of directed circuits. Let $u,\ell : E\to\mathbb{Q}_{\geq 0}$ satisfy $u\geq\ell\geq 0$; then the following are equivalent:*

1. *there exists $\alpha:\mathscr{C}\to\mathbb{Q}_{\geq 0}$ such that $u\geq\sum_{c\in\mathscr{C}}\alpha(C)f_C\geq\ell$;*

2. *for each $X\subset V, u(\partial^+(X)\geq\ell(\partial^+(V-X))$.*

where $f_C: E\to\mathbb{Q}_{\geq 0}$ is the characteristic function on C, i.e. $f_C(e)=1$ if $e\in C$, 0 otherwise and $\partial^+(S) := \{(x,y)\in E : x\in S$ and $y\notin S\}$, for every $S\subseteq V$.

Seymour in [20] considered undirected version of this Theorem. Given an undirected graph $G(V,E)$, and u,ℓ maps from $E\to\mathbb{Q}_{\geq 0}$ when can we write it as a "sum of circuits" that is: There exists $\alpha : \mathscr{C} \to \mathbb{Q}_{\geq 0}$ such that $u \geq \sum_{c \in \mathscr{C}} \alpha(C) f_C \geq \ell$, where $\mathscr{C}$ is the set of all circuits in $G(V, E)$.

Let $V = V_1 \cup V_2$ be a partition of the vertex set. Let $B \subset E(G)$ be the set of edges that have one end point in V_1 and another in V_2. This set B is call a *cut*. For any graph that can be written as a sum of circuits it is clearly necessary that for every cut B and every edge $e \in B$ the following inequality holds:

$$\ell(e) \leq \sum_{f \in B - \{e\}} u(f).$$

Seymour proved that it is also sufficient.

Theorem 1.3. *(Seymour [20]) Let $V = (V, E)$ be a simple graph and let $\mathscr{C}$ be its collection of circuits. Let $u, \ell : E \to \mathbb{Q}_{\geq 0}$ satisfy $u \geq \ell \geq 0$ and f_C denote the characteristic function on C; then the following are equivalent:*

1. *there exists $\alpha : \mathscr{C} \to \mathbb{Q}_{\geq 0}$ such that $u \geq \sum_{c \in \mathscr{C}} \alpha(C) f_C \geq \ell$;*
2. *for each cut B and each $e \in B$; $u(B - \{e\}) \geq \ell(e)$.*

We consider this question in 2-colored weighted graphs. Let $G(V, E)$ be a graph (the graph may have multiple edges) with $c : E \to \{R, B\}$ and two functions $u, \ell : E \to \mathbb{Q}_{\geq 0}$. Note that for colored graphs we are using c to denote the color map on the edges. We consider the following natural question.

Question 1.1. *Given $G(V, E)$ a graph (may have multiple edges) with $c : E \to \{R, B\}$, two functions $\ell, u : E \to \mathbb{Q}_{\geq 0}$ with $\ell(e) \leq u(e)$ does there exist $w : E \to \mathbb{Q}_{\geq 0}$ with*

1. *$\ell(e) \leq w(e) \leq u(e)$ and*
2. *for all vertices $v \in V$, $\sum_{e \in E_R(v)} w(e) = \sum_{e \in E_B(v)} w(e)$, where were $E_R(v)$, $E_B(v)$ denotes the set of R, B colored edges incident on v respectively.*

The key step in proving Hoffman circulation theorem is characterization of reachability from a vertex u to a vertex v by a directed path. In particular if a vertex v is reachable from vertex u then there is a simple algorithm that can produce such a path. If v is not reachable from u then there exists a partition of $V = V_1 \cup V_2$ such that $u \in V_1$, $v \in V_2$ and there are no edges are there starting from a vertex in V_1 and ending in a vertex in V_2. A similar theory is also needed to solve the above Question.

2 Alternating reachability

Let $G = (V, E)$ be a graph with vertex set V and edge set $\{e_1, e_2, \ldots, e_m\}$. The *coloring* of the edges is given by the map $c : E \to C := \{R, B\}$ and a set $S \subseteq V$

of *terminals*. Though we will use only two colors the theory works for finitely many colors.

An *alternating trail* connecting $s, t \in V$ is defined as the sequence

$$W = (v_0 = s, e_1, v_1, e_2, v_2, \ldots, e_m, v_m = t), \qquad m \geq 0,$$

where $v_i \in V$ for all i, $e_j \in E$ for all j, e_j's are distinct, and $c(e_j) \neq c(e_{j+1})$ for each $j = 1, \ldots, m-1$. The alternating trail W is called *closed* if $v_0 = v_m$ and $c(e_m) \neq c(e_1)$. See Figure 1b for an example.

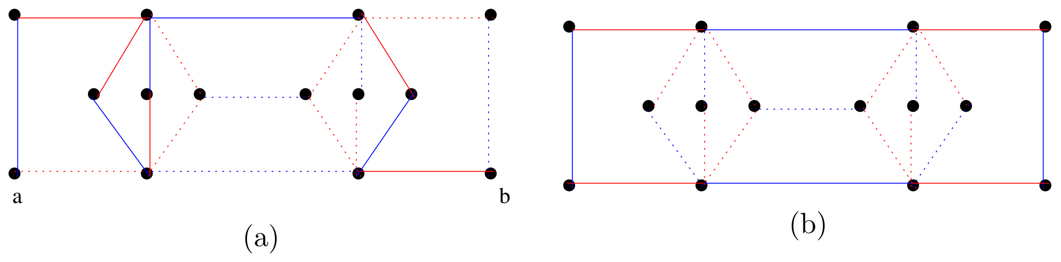

Figure 1

We are interested in the question: when can we find alternating trail connecting two distinct terminals? Next we define Tutte Sets which are the obstacles to such trails.

Definition 2.1. *A subset $A \subseteq (V - S)$ is a* Tutte set *when*

(i) each component of $G - A$ has at most one terminal.

(ii) A can be written as a disjoint union (denoted $\dot\cup$, empty blocks allowed)

$$A = \dot{\bigcup}_{c \in C} A(c)$$

such that conditions (a), (b), and (c) below hold.

A vertex $u \in A$ is said to have color c *if $u \in A(c)$ (there is a unique such c). An edge $e \in E$ is said to be* mismatched *if e connects a vertex $u \in A$ with a vertex $v \in V - A$ and $c(e)$ is different from the color of u, or e connects two vertices $u, v \in A$ and $c(e)$ is different from both the colors of u and of v.*

Conditions (a), (b), and (c) are as follows:

(a) if H is a component of $G - A$ containing a terminal, then there is no mismatched edge with an endpoint in H.

(b) if H is a component of $G - A$ containing no terminals, then there is at most one mismatched edge with an endpoint in H.

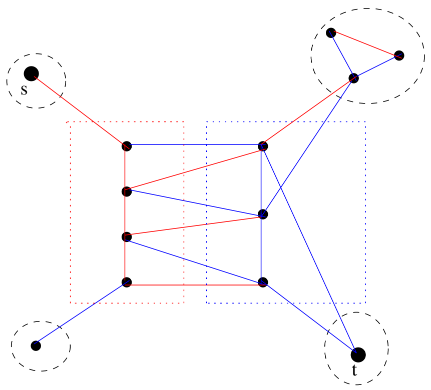

Figure 2

(c) there are no mismatched edges with both endpoints in A.

Figure 2 gives an example of Tutte set.

The next theorem shows that a Tutte set is an obstruction to the existence of an alternating trail connecting distinct terminals.

Theorem 2.1. *Suppose there is no alternating trail starting at s and ending in t. Then a Tutte set A exists such that s and t lie in distinct components of $G - A$ with no 'mismatched' edge.*

Proof. **Sufficiency:**
Let A be a Tutte set and S_1 and S_2 be two components of $G - A$, without mismatched edges. Then there cannot be any alternating trail between a vertex of S_1 and a vertex of S_2.

Suppose an alternating trail T starts at $s \in S_1$. If this alternating trail leaves the component S_1 then it must first reach a vertex in A. Since the component S_1 does not have any mismatched edge it must reach the vertex $v \in A$ with an edge of same color as the vertex v. Since the trail is alternating the next vertex cannot be from a matched component of $G - A$. So the next vertex must be a vertex from A with color different from $c(v)$ or from a component with a mismatched edge. If it is from a component with a mismatched edge then the trail may enter with the mismatched edge and continue inside that component or it will eventually re-enter the set A. In particular, whenever a vertex u in the alternating trail is in A, the edge e used to reach it, is colored same as the vertex u in A i.e. $c(e) = c(u)$. So the alternating trail can never enter a component of $G - A$ without mismatched edges. Hence there can never be an alternating trail connecting a vertex of S_1 and a vertex of S_2.

This completes the sufficiency part. □

To prove the converse we reduce this problem to the special case of finding maximum matching in a graph. For this we define a new graph $G' = (V', E')$

together with a matching M such that G has an alternating trail if and only if G' has an M-augmenting path. This construction is due to Jácint Szabó. See Figure 3 and Figure 4 for an example. Then if G has no alternating trail, the matching M in G' is maximum. Using Edmond's algorithm (see [9]), we get a "maximal blossom forest" w.r.t. M in G'. From this maximal blossom forest in G' we extract our desired Tutte set of G.

First we recall some basic notions connected with Edmond's algorithm. Our purpose here is only to set down terminology and we omit all details. Our main reference for Edmonds' algorithm is Chapter 9 of Lovász and Plummer [12] to which we refer for all unexplained terminology. Consider a graph together with a matching. Edmond's algorithm maintains a *(rooted) blossom forest* with respect to the matching. The blossom forest contains vertex disjoint factor critical subgraphs called *blossoms* such that the given matching contains a near-perfect matching of every blossom and, upon shrinking every blossom to a single vertex, we obtain an alternating forest with respect to the image of the matching. Moreover, the images of the blossoms are outer points of this alternating forest. It follows that every blossom has a unique vertex which is either unmatched by the matching or matched to a vertex outside the blossom. This vertex is called the *base* of the blossom. In case the base is matched to a vertex outside the blossom we call this matched edge the *leading edge* into the base. We classify the vertices of the graph as follows: vertices not in the blossom forest are called *out of forest* and the other vertices are *in forest*. The in forest vertices are further classified as *inner vertices* and *blossom vertices* (in particular, the exposed vertices (w.r.t the matching) are blossom vertices). A blossom vertex is *trivial* if it is the only vertex in its blossom, otherwise it is *nontrivial*. Likewise, a blossom is *trivial* if it contains only one vertex and is *nontrivial* if it contains more than one vertex. Thus, for a matched edge exactly one of the following three possibilities hold: both endpoints are out of forest or both endpoints belong to the same nontrivial blossom or one of the endpoints is the base of a blossom and the edge is the leading edge into this base. The *height* of a blossom is the distance of the blossom from the corresponding root blossom in the blossom forest (so the root blossoms, i.e., the blossoms containing the exposed vertices have height zero). At any stage of the algorithm we can perform three operations: *growing, shrinking, and augmenting.* If the matching were maximum then we will never perform an augmenting operation and only apply the other two operations. When even those two operations cannot be performed we obtain a *maximal blossom forest* with respect to the maximum matching.

Construction of Szabó

Given a graph $G(V, E)$, a coloring $c : E \to \{R, B\}$ of the edges and a set S of terminals we define a graph $G' = (V', E')$ as follows:

$$V' = S \,\dot{\cup}\, \{e_u, e_v : e \in E \text{ and } e \text{ has endpoints } u \text{ and } v\}.$$

The edges in E' and their endpoints are as follows:

- *Every $e \in E$ is also an edge in E'. If its endpoints in G are u and v its endpoints in G' are e_u and e_v.*
- *For every $v \in V$, $e, f \in \nabla_G(v)$ with $c(e) \neq c(f)$, there is an edge $(ef)_v \in E'$ with endpoints e_v and f_v.*
- *For every $s \in S$ and every $e \in \nabla_G(s)$, there is an edge $se \in E'$ with endpoints s and e_s.*

Note that the subset $M = \{e \; : \; e \in E\}$ of E' is a matching in G'. Also note that, for $u, v \in V$, there is a 1-1 correspondence between alternating $u - v$ trails in G with first edge e and last edge f and M-alternating $e_u - f_v$ paths in G' with first edge e and last edge f. It follows that, for $s, t \in S, s \neq t$, there is a $s - t$ alternating trail in G iff there is an $s - t$ M-augmenting path in G'. Thus, by our assumption (of no alternating trails connecting distinct terminals) M is a maximum matching in G'.

We shall now extract a Tutte set in G from information provided by a maximal blossom forest F in G' with respect to the matching M. The terminology out of forest vertex, blossom vertex etc. will be with respect to F.

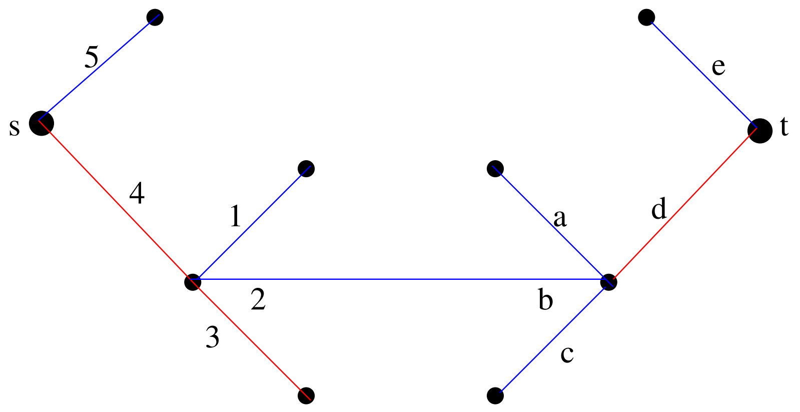

Figure 3

Lemma 2.1. *(i) Let $v \in V - S$. If there exists $e \in \nabla_G(v)$ such that e_v is an in forest vertex then there exists $f \in \nabla_G(v)$ such that f_v is a blossom vertex.*
(ii) Let $v \in V - S$. Define $\Gamma_G(v) = \{e \in \nabla_G(v) \; : \; e_v$ is a blossom vertex$\}$. Then exactly one of the following three possibilities hold:

(a) All vertices e_v, $e \in \nabla_G(v)$ are out of forest (note that, from part (i), this is equivalent to $\Gamma_G(v)$ being empty).

(b) $\Gamma_G(v)$ is not empty , each e_v, $e \in \Gamma_G(v)$ is a trivial blossom vertex and all edges in $\Gamma_G(v)$ have the same color.

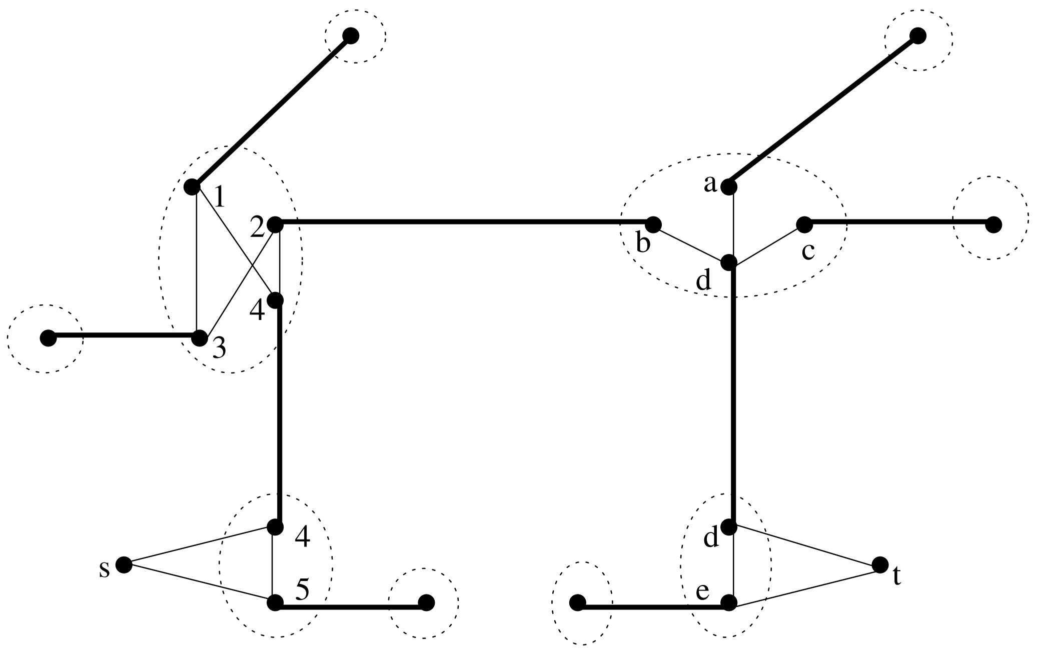

Figure 4

(c) *$\#\Gamma_G(v) \geq 2$, each $e_v,\ e \in \Gamma_G(v)$ is a nontrivial blossom vertex and all of them belong to the same nontrivial blossom of F.*

Proof. (i) Suppose e_v is an inner vertex. Consider the (unique) path P from e_v to a root in F. Since $v \in V - S$, the other endpoint (in V') of the edge of P incident to e_v must be a vertex of the form f_v, which is a blossom vertex.
(ii) Assume that (a) does not hold, i.e., some $e_v,\ e \in \nabla_G(v)$ is in forest. From part (i) it follows that $\Gamma_G(v) \neq \emptyset$. Now assume that each $e_v,\ e \in \Gamma_G(v)$ is a trivial blossom vertex. If $e, f \in \Gamma_G(v)$ have different colors then there will be an edge (in G') between the blossom vertices e_v and f_v, contradicting the fact that F is a maximal blossom forest. Thus all edges in $\Gamma_G(v)$ have the same color.

Now assume that both (a) and (b) do not hold, i.e., $\Gamma_G(v) \neq \emptyset$ and some $e_v,\ e \in \Gamma_G(v)$ is a nontrivial blossom vertex. Since $v \in V - S$, it follows that there is a $f \in \Gamma_G(v)$ with $c(e) \neq c(f)$ and such that f_v belongs to the same nontrivial blossom as e_v. Thus $\#\Gamma_G(v) \geq 2$. If $h_v,\ h \in \Gamma_G(v)$ were to belong to some other blossom of F then, since h would have a different color from one of e or f, there would be an edge (in G') between vertices in two different blossoms, a contradiction. Thus (c) holds. $\square$

We now classify the vertices of $V - S$ as follows:

- $N(S) = \{v \in V - S\ :\ v$ satisfies condition (a) in Lemma 2.1(ii)$\}$,
- $I(S) = \{v \in V - S\ :\ v$ satisfies condition (b) in Lemma 2.1(ii)$\}$,

- $T(S) = \{v \in V - S \ : \ v$ satisfies condition (c) in Lemma 2.1(ii)$\}$.

For $c \in C$, define $I(S,c) = \{v \in I(S) \ : \ c(e) = c$ for some (equivalently, all) $e \in \Gamma_G(v)\}$. If $v \in I(S,c)$, we say that v has *color* c. Elements of $N(S), I(S),$ and $T(S)$ are called *achromatic, monochromatic*, and *bichromatic* vertices respectively. This terminology is justified by the next result which gives a Gallai-Edmonds type decomposition of $V - S$. For $v \in T(S)$, we denote by B_v the nontrivial blossom of F containing all the vertices $e_v, \ e \in \Gamma_G(v)$. Similarly, for $s \in S$, we let B_s denote the blossom of F containing the vertex s. Note that B_s either consists of the single vertex s or is equal to B_v for some $v \in T(S)$.

Theorem 2.2. *([8]) (i) $N(S)$ equals the set of all vertices $t \in V - S$ such that, for all $s \in S$, there is no alternating s-t trail in G.*

(ii) $T(S)$ equals the set of all vertices $t \in V - S$ such that, for some $s \in S$, there are two alternating s-t trails in G whose last edges have different colors.

(iii) $I(S,c)$ equals the set of all vertices $t \in V - S$ satisfying the following property: there are alternating trails starting from S and ending in t, and the last edges of all such alternating trails have the color c.

Proof. The sets $N(S), T(S)$, and $I(S,c), \ c \in C$ partition $V - S$ and the sets on the right hand side of the three statements above are clearly pairwise disjoint. Thus the result will follow if we show that each left hand side is contained in the corresponding right hand side.

Let $v \in N(S)$. Suppose that, for some $s \in S$, there is a $s - v$ alternating trail in G with last edge $e \in \nabla_G(v)$. Then there is a $s - e_v$ M-alternating path in G'. Since e_v is out of forest, this contradicts the maximality of F. Thus there is no $s - v$ alternating trail in G, for any $s \in S$.

Let $v \in T(S)$. Choose $e \in \Gamma_G(v)$. Like in the proof of part (c) of Lemma 2.1(ii) above there exists $f \in \Gamma_G(v)$ with $c(e) \neq c(f)$. Both f_v and e_v belong to B_v. Let $s \in S$ be the base of the root blossom of the component of F containing B_v. It follows that there are M-alternating $s - e_v$ and $s - f_v$ paths in G' with last edges e and f respectively. Thus there are $s - v$ alternating trails in G with last edges of different colors.

Let $v \in I(S,c)$ and let $f \in \Gamma_G(v)$. Then $c(f) = c$. Let $s \in S$ be the base of the root blossom of the component of F containing the trivial blossom vertex f_v. Clearly there is a $s - f_v$ M-alternating path in G' with last edge f and thus there is a $s - v$ alternating trail in G with last edge f of color c.

Now suppose that, for some $s' \in S$, there is a $s' - v$ alternating trail in G with last edge e. Then there is a $s' - e_v$ M-alternating path in G' with last edge e. Since F is maximal it is easy to see that e_v is a blossom vertex. Since $v \in I(S,c)$, it follows that $c(e) = c$. □

We shall now show that the monochromatic vertices form a Tutte set (with coloring as given above). We begin with the following result, which places restrictions on the edges connecting the four different types of vertices in G: the terminals and the achromatic, monochromatic, and bichromatic vertices.

Lemma 2.2. *(i) No edge in G connects a vertex of $S \cup T(S)$ to a vertex of $N(S)$.*

(ii) If an edge e in G connects a vertex u in $I(S,c)$ to a vertex in $N(S)$ then $c(e) = c$.

(iii) If an edge e in G has endpoints $u, v \in S \cup T(S)$ then either $B_u = B_v$ or one of e_u, e_v is a base in F and e is the leading edge into this base.

(iv) If an edge e in G has endpoints $u \in S \cup T(S)$ and $v \in I(S,c)$ and $c(e) \neq c$ then e_u is a base in F and e is the leading edge into e_u.

(v) If an edge e in G has endpoints $u \in I(S,c)$ and $v \in I(S,d)$ then $c(e) = c$ or d.

Proof. (i) Let $e \in E$ with endpoints $u \in S \cup T(S)$ and $v \in N(S)$. Consider the edge $e \in M$ with endpoints e_u, e_v. By Lemma 2.1(ii)(a) e_v is out of forest and thus so is e_u. If $u \in S$ then there is an edge $ue \in E'$ connecting u to e_u, contradicting the maximality of F. If $u \in T(S)$ then it follows (like in Lemma 2.1(ii)(c)) that there exist $f, g \in \Gamma_G(u)$ with $c(f) \neq c(g)$ and such that f_u, g_u both belong to B_u. One of f or g has color different from that of e and thus there is an edge in E' connecting a blossom vertex (f_u or g_u) to an out of forest vertex e_u, a contradiction to the maximality of F.

(ii) Let $v \in N(S)$ be the other endpoint of e and consider the edge $e \in M$ with endpoints e_u and e_v. As in part (i) e_u, e_v are out of forest. Let $f \in \Gamma_G(u)$. Then $c(f) = c$. If $c(e) \neq c$ then there is an edge in E' connecting the (trivial) blossom vertex f_u to the out of forest vertex e_u, a contradiction to the maximality of F. Thus $c(e) = c$.

(iii) Consider $e \in M$ with endpoints e_u, e_v. If e_u, e_v are out of forest then we can get a contradiction like in part (i). So e_u, e_v are in forest. Thus, either both endpoints of e are in the same blossom (in which case this blossom must be $B_u = B_v$) or one of e_u, e_v is a base in F and e is the leading edge into this base.

(iv) Consider $e \in M$ with endpoints e_u, e_v. Like in part (i) we can show that e_u, e_v are in forest. If both endpoints of e were in the same blossom it would follow that $v \notin I(S)$. Thus one of e_u, e_v is a base in F and e is the leading edge into this base. If e_v were the base then, since $v \in I(S,c)$, it would follow that $c(e) = c$, a contradiction. Thus e_u is the base.

(v) Consider $e \in M$ with endpoints e_u, e_v. First suppose that e_u, e_v are out of forest. Let $f \in \Gamma_G(u)$. Since F is maximal there cannot be an edge in E' with endpoints f_u and e_u. Thus $c = c(f) = c(e)$. Now suppose that e_u, e_v are in forest. If both endpoints of e were in the same blossom it would follow that $u, v \notin I(S)$. Thus e is a leading edge in F connecting the base of a blossom (e_u or e_v) to an inner vertex (e_v or e_u). Thus $c(e) = c$ or d. □

We now identify the connected components of $G[S \cup T(S)]$ (the induced subgraph of G on the vertex set $S \cup T(S)$). Define a map $\phi : V' \to V$ by $\phi(s) = s$

($s \in S$) and $\phi(e_v) = v$ ($e \in E$ with v as an endpoint). Consider the blossoms $B_u,\ u \in S \cup T(S)$ (note that any two of these blossoms are either identical or vertex disjoint). Enumerate the distinct blossoms among these as $B_{u_1}, B_{u_2}, \ldots, B_{u_k}$. For $i = 1, \ldots, k$ set $V_i = \phi($ vertex set of $B_{u_i})$. It follows by Lemma 2.1(ii)(c) that the V_i are pairwise disjoint and $S \cup T(S) = V_1 \cup \cdots \cup V_k$. Let $G_i = G[V_i],\ 1 \le i \le k$. It is easy to see that $G_1, \ldots, G_k$ are connected but they need not be the connected components of $G[S \cup T(S)]$. There could be edges in G with endpoints in two distinct G_i's. Let $X = \{e \in E\ :\ e$ has endpoints in G_i and G_j, for some $i \ne j\}$. Thus we obtain $G[S \cup T(S)]$ by taking the disjoint union of $G_1, G_2, \ldots, G_k$ and adding the edges in X. According to Lemma 2.2(iii) each edge e in X is a leading edge in F (into the base of one of the blossoms $B_{u_1}, \ldots, B_{u_k}$). Among the blossoms $B_{u_1}, \ldots, B_{u_k}$ choose one, say B_{u_l}, of maximum height and consider G_l. It follows that there can be at most one edge in X incident with G_l. Applying this argument inductively we see that adding the edges in X to the disjoint union of the G_i's yields, informally speaking, a disjoint union of rooted trees on the G_i's. More precisely, we make the following observations:

(x) Take a connected component C of $G[S \cup T(S)]$. Since each G_i is connected, C will consist of certain G_i's together with a certain subset $Y \subseteq X$ of edges in X. Shrinking each G_i contained in C to a single vertex we get a rooted tree with edge set Y. Consider the G_l corresponding to the root of this tree. The image under ϕ of the base of B_{u_l} will be a vertex in G_l and is called the *base* of C. Note that here we are defining a base in V and not all images of the bases of the blossoms B_{u_i} are bases in V. Also note the following.

(y) If e is a leading edge in F one of whose endpoints in G is in C and the other not in C then the endpoint in C must be the base of C.

(z) Each $s \in S$ is a base in V. In particular, no two vertices in S are in the same component of $G[S \cup T(S)]$.

Theorem 2.3. *([8])Suppose that G has no alternating trails connecting distinct terminals in S. Then $I(S)$ is a Tutte set with coloring given by $I(S) = \cup_{c \in C} I(S, c)$.*

Proof. From Lemma 2.2(i) we see that $G - I(S)$ is the disjoint union of $G[S \cup T(S)]$ and $G[N(S)]$. We now check the conditions in Definition 2.1. Condition (i) follows from observation (z) above and condition (ii)(c) follows from Lemma 2.2(v). We are left to verify conditions (ii)(a) and (ii)(b).

It follows from Lemma 2.2(ii) that there is no mismatched edge with an endpoint in one of the components of $G[N(S)]$. Now consider a mismatched edge e with an endpoint u in one of the components C of $G[S \cup T(S)]$. From Lemma 2.2(iv) and observations (y),(z) above we see that u is the base of C, e is the leading edge (in F) into e_u, and C contains no terminal (as there are no leading edges in F into vertices in S). This shows that there can be at most one mismatched edge with an endpoint in C, verifying condition (ii)(b) and that a mismatched edge cannot have an endpoint in a component of $G[S \cup T(S)]$ containing a terminal, verifying condition (ii)(a). $\square$

Alternating reachability problem was first considered (in a non algorithmic form) by Tutte (in a slightly different version) who calls the obstructions to the existence of alternating trails r-barriers [23, 22, page 331]. There is a very minor error in Tutte's formulation. If we were to apply his definition of r-barrier to the version of alternating reachability considered in this paper, then condition (c) in Definition 2.1(ii) would read:

(c) If an edge e connects two vertices of A, then these vertices have different colors and one of them has the color $c(e)$,

Our condition (c) paraphrased reads: if an edge e connects two vertices of A, then one of them has the color $c(e)$. Thus every r-barrier is a Tutte set but not conversely. It is easy to find instances of the alternating reachability problem where there are no alternating trails connecting distinct terminals, but all obstructions are Tutte sets and not r-barriers.

Since, since finding alternating trails reduces to finding matching in graphs, it can be found in $O(|V'||E'|) = O(|E|^3)$. In particular it can be found in polynomial time.

3 Balanced decomposition

In this section we state an application of alternating reachability. Let $G(V, E)$ be a graph and $c : E \to \{R, B\}$ be a map that colors the edges red or blue. Consider the set of all weights $w : E \to \mathbb{R}_{\geq 0}$ on the edges such that at each vertex the sum of the edges colored blue is equal to the sum of the weights of the edges colored red. Set of all such weights w form a cone and we call it *a*lternating cone. This is analogous to the cone of circulations for directed graphs. In [6] basic properties of this cone has been studied. In [6] it was also showed how the search for an integral vector in the alternating cone of a 2-colored graph satisfying given lower and upper bounds on the edges can be reduced to the alternating reachability problem in a 2-colored graph.

A *cut* in G is the set of edges between X and $V - X$, for some nonempty proper subset X of V. Consider a weight $w : E \to \mathbb{Q}_{\geq 0}$ on the edges. It is said to satisfy (a) the *balance condition* if $\sum_{e \in E_R(v)} w(e) = \sum_{e \in E_B(v)} w(e)$, at each vertex $v \in V$, and (b) the *cut condition* if for each cut B and for each edge e in B, $w(e) \leq w(B - \{e\})$. Here $E_R(v)$ (respectively $E_B(v)$) denotes the set of R colored (respectively B colored) edges incident on v.

In case of circulations in weighted directed graphs we have inflow equal to out flow at each vertex. In 2-colored graphs it is captured by the notion balanced. That is, at every vertex the weight of the blue colored edges is equal to the weight of the red colored edges.

In existence of circulations is given by Hoffman circulation theorem (1.2) and that is proved using the network flow techniques. Seymour (1.3) proved the analogous theorem for undirected graphs using a tricky lemma due to Seymour and Giles.

Circulation in a network can be found in polynomial time using Edmond's hueristics. Decomposing a weighted undirected graph as a sum of circuits can also be found in polynomial time. This was first shown by Arkin and Papadimitriou in [2].

Let $\mathcal{T}$ be the collection of closed alternating trails in G. In this section we give a polynomial time algorithm to find an assignment $\alpha : \mathcal{T} \to \mathbb{Q}_{\geq 0}$ such that $\sum_{T\in\mathcal{T}} \alpha(T) f_T = w$, where f_T denotes characteristic function on T.

In [7] it was shown that

Theorem 3.1. *([7]) Let $V = (V, E)$ be a two colored simple graph with coloring $c : E(G) \to \{R, B\}$, $w : E(G) \to \mathbb{Q}_{geq0}$ and let $\mathcal{T}$ be its collection of closed alternating trails. Let f_T, $T \in \mathcal{T}$ denote the characteristic function on T; then the following are equivalent:*

1. *there exists $\alpha : \mathcal{T} \to \mathbb{Q}_{\geq 0}$ such that $u \geq \sum_{T\in\mathcal{T}} \alpha(T) f_C \geq \ell$;*
2. *(a) each cut B and each $e \in B$; $u(B - \{e\}) \geq \ell(e)$.*
   (b) for all vertices v, $\sum_{e\in E_R(v)} w(e) = \sum_{e\in E_B(v)} w(e)$.

Here we show the following.

Theorem 3.2. *Let $G(V, E)$ be a 2-colored simple graph with coloring c and let $\mathcal{T}$ be the collection of closed alternating trails in G. Let $w : E \to \mathbb{Q}_{\geq 0}$ satisfies the balance and the cut condition. Then we can find, in polynomial time, an assignment $\alpha : \mathcal{T} \to \mathbb{Q}_{\geq 0}$ such that $\sum_{T\in\mathcal{T}} \alpha(T) f_T = w$.*

To prove this theorem we also use the network flow algorithm and a ratio optimization problem. In general ratio optimization problems are hard. In this case we can use an idea by Megiddo ([14]) to solve it in polynomial time.

For $X, Y \subseteq V$, we denote by $\nabla_G(X, Y)$ the set of all edges of G with one endpoint in X and the other endpoint in Y . Let D be a tight cut in G with sides X and $V - X$ of sizes at least 3, and let $e \in D$, an edge between $u_1 \in X$ and $u_2 \in V - X$ be the tight edge. That is, $w(D - e) = w(e)$. Now, we define two edge-weighted 2-colored graphs G_X (respectively, G_{V-X}) by doing the following:

- Delete all edges in $\nabla_G(X, X)$ (respectively, $\nabla_G(V - X, V - X)$).
- Replace X (respectively, $V - X$) with $\{u_1, u'_1\}$ (respectively, $\{u_2, u'_2\}$), where $u'_1, u'_2 \notin V$ are two new vertices.
- The endpoints of each edge in $\nabla_G(V - X, V - X) \cup \{e\}$ (respectively, $\nabla_G(X, X) \cup \{e\}$) remain the same.

- The endpoint of each edge $f \in D - e$ in $V - X$ (respectively, X) is the same as before, and the endpoint in X (respectively, $V - X$) is u_1 (respectively, u_2) if $c(f) \neq c(e)$ and is u'_1 (respectively, u'_2) if $c(f) = c(e)$.

- Add a new edge f_1 (respectively, f_2) between u_1 and u'_1 (respectively, u_2 and u'_2). The color of f_1 (respectively, f_2) is opposite $c(e)$. All other edges retain their original color.

- Define a weight function w_1 on the edges of G_X by setting $w_1(f_1) = \sum_h w(h)$, where the sum is over all $h \in D - e$ with $c(h) = c(e)$, and $w_1(h) = w(h)$ for all other edges h of G_X. Similarly, define a weight function w_2 on the edges of G_{V-X} by setting $w_2(f_2) = \sum_h w(h)$, where the sum is over all $h \in D - e$ with $c(h) = c(e)$, and $w_2(h) = w(h)$ for all other edges h of G_{V-X}.

It can be verified that w_1 (respectively, w_2) satisfies the balance condition at every vertex of G_X (respectively, G_{V-X}) and the cut condition for G_X (respectively, G_{V-X}).

Now we describe the algorithm for decomposing $G(V, E, w, c)$ as a rational sum of closed alternating trails.

Algorithm 1

1. Find a tight cut D in G with both the sides having size at least 3. This is done by finding minimum cut for the graph $(V, E \setminus \{e\})$ for every edge e by using the max-flow algorithm. Let $e \in D$ be the tight edge, i.e. $w(e) = w(D - \{e\})$. Now we divide our problem into two subproblems having fewer vertices. This can be done by solving the problem for (G_X, c_1, w_1) and (G_{V-X}, c_2, w_2) and combining their solutions for a solution of an overall problem ([7]). Similar idea was also used in [19] (The matching polytope, section 8.11, page 109).

2. If no such tight cut is found, we construct one. We choose a closed alternating trail T and subtract the maximum we can from the weights of each edge of T.

3. Stop when no longer step 1 or step 2 is possible.

In the algorithm we did not separately state the case when we have a tight cut with one side having 2 or 1 vertex. In this case we can continue to execute step 2. This follows from the following two observations.

1. Suppose there is a tight cut B with $e \in B$, $w(e) = w(B - \{e\})$ and one of the sides has exactly 1 vertex. Then from balance condition it follows that all edges in $B - \{e\}$ have color different from $c(e)$.

2. Suppose there is a tight cut B with $e \in B$, $w(e) = w(B - \{e\})$ and one of the sides has exactly 2 vertices. Let the two vertices be u_1 and u_2 and u_1 be the endpoint of e. Then from balance condition it follows that Then color of all edges $\partial(u_1) - e$ is different from $c(e)$.It also follows that all edges incident on u_2 other than $\{u_1, u_2\}$ must have color equal to $c(e)$.

These color restrictions imply that if any trail uses any edge in a tight cut B that has a side with at most 2 vertices, then it must use the tight edge e for which $w(e) = w(B - e)$. Hence, step 2 takes care of this case.

The algorithm is an iteration of (1) and (2) till it is no longer possible. To guarantee that the algorithm will eventually terminate we need to show that progress is always made in step 2. We will show this later. In [2] they had to use a tricky Seymour-Giles Lemma to ensure that progress is always made in a similar step of their result.

To perform step 2 of Algorithm 1, we need a polynomial time algorithm to find a closed alternating trail.

Theorem 3.3. *Let $G(V, E, c)$ a 2-colored bridgeless graph, where every vertex has at least one incident red colored edge and at least one incident blue colored edge. Then we can find a closed alternating trail T in polynomial time.*

Proof. We find a CAT as follows. Fix an edge $e = \{u, v\}$, and consider the graph G_e with $V(G_e) = V(G) \cup \{s, t\}$ and $E(V(G_e)) = (E \setminus e) \cup \{\{u, s\}, \{v, t\}\}$. We also set $c(\{s, u\}) = c(\{t, v\}) = c(e)$.

If there is an alternating $s - t$ trail for the graph G_e, then we can construct a CAT in G by deleting the edges $\{u, s\}, \{v, t\}$ and adding back the edge e.

Otherwise we have a maximal (by set inclusion) Tutte set T with s and t in distinct components of $G_e - T$. let O_x be the component of $G_e - T$ that contain the vertex s. Now we construct a closed alternating trail that will not include any vertex from the component O_s. Since G_e is connected there must be a vertex x in the component O_s containing s connected to a vertex in T. Also T is maximal, so every vertex in $O_s - s$ is reachable from s by two alternating trails with the last edge of different colors (Theorem 2.2, (ii)). One of these two alternating trails ending at x can be extended to include a vertex x_1 in T. Now we consider a maximal alternating trail starting at s that includes the vertex x_1 and ends with a vertex in T. The trail cannot stop at a vertex x_1 because it has at least two incident edges of different colors. The the next vertex is either a vertex in T or a vertex in a component with exactly one mismatched edge. If the next vertex x_2 is in T then it reached x_2 with an edge e_2 with $c(e) = c(x_2)$. If the next vertex is in a component O_1 with exactly one mismatched edge, then the alternating trail must enter O_1 with the mismatched edge (since the edge colors need to alternate). The component O_1 must have at least another edge connected to T as G is bridgeless and v or u cannot be in it. Hence, the alternating trail can be extended to another vertex in T (Theorem 2.2, (ii) and maximality of T), say x_2. This means we have an alternating trail of the type

$$s = x_0, T_1, x_1, T_2, x_2 \dots x_i \dots,$$

where T_j are alternating trails starting at a vertex in $x_{j-1} \in T$ and ending at a vertex $x_j \in T$. The internal vertices of T_j may be empty or from a component of $G_e - T$ that have exactly one mismatched edge. This trail cannot stop when a vertex $x_j \in T$ is visited for the first time. Since there are finitely many vertices in T the alternating trail will eventually repeat a vertex in T and that will give the desired closed alternating trail. □

It may be noted that this also implies the Seymour-Giles Lemma as a corollary. In [7] it was shown that they are equivalent. For completeness we state Seymour-Giles Lemma.

Suppose that we are given a graph $G(V, E)$, and a function $F : V \to E$ mapping each vertex to one of its incident edges. We say a circuit respects F, if for every vertex u in the circuit, the edge $F(u)$ is also in the circuit.

Lemma 3.1. *(Seymour-Giles [20]) Given a bridgeless graph $G(V, E)$ and a function F as described above, there exists a circuit respecting the function F.*

Proof. We construct a new two colored graph $G(V', E', c)$ as follows. We start with the graph G. If an edge $e = \{u, v\} \in E(G)$ appears as the image of F only for the vertex u, then add a new vertex u_e, remove the edge e, and add two new edges $\{u, u_e\}$ and $\{u_e, v\}$. Color the first edge red and the latter edge blue. If $e = \{u, v\} \in E(G)$ appears as the image of F for the vertex u and the vertex v as well, then color it red. All other remaining uncolored edges are colored blue. In this graph by we can apply Theorem 3.3, and the closed alternating trail gives the desired F respecting circuit. □

Seymour-Giles Lemma can be proved using shrinking or matching theory techniques. In [2] Seymour-Giles original "shrinking" proof converted into an polynomial time algorithm.

In Step (2) of the algorithm, once we have a positive length closed alternating trail T, we next need to compute the maximum amount t that can be subtracted from the weights of the edges of T. Let $w_t : E \to \mathbb{Q}_{\geq 0}$ be the weight function after subtracting t from the weights of the edges in T. The maximum value of t should satisfy the following conditions:

1. $w_t(e) \geq 0$ for each $e \in E$
2. w_t satisfy the balance condition
3. w_t satisfy the cut condition

Since T is a closed alternating trail, the balance condition will be automatically satisfied. Let t_1 be the minimum weight of the edges in T. In order to satisfy condition (a), the value of t should be at most t_1. If w_{t_1} satisfies the cut condition, then t_1 is the required maximum value.

Suppose w_{t_1} does not satisfy the cut condition. Then we compute the maximum t which does not violate the cut condition. For each edge e we perform the following step: If $e \in T$, then for any cut D containing e, $w_t(e) = w(e) - t$ and $w_t(D-e) = w(D) - w(e) - (|D \cap T| - 1)t$. Thus w_t satisfies the cut condition if for all cuts D containing e, $w(e) - t \leq w(D) - w(e) - (|D \cap T| - 1)t$. For maximum value of t, we need to minimize the following ratio for all cuts D containing e:

$$\frac{w(D) - 2w(e)}{|D \cap T| - 2}. \tag{3.0.1}$$

(Similarly, if $e \notin T$, then we need to minimize the ratio $\frac{w(D)-2w(e)}{|D \cap T|}$ for all cuts D containing e.)

To minimize (3.0.1), we use a technique by Megiddo (see [14]).

Now we consider two minimization problems, Problem A and Problem B.

Problem A:

Minimize $c_1x_1 + \cdots + c_nx_n$

subject to $x := (x_1, x_2, \ldots, x_n) \in F$, where F is a set of feasible solutions.

Problem B:

$$\text{Minimize} \quad \frac{a_0 + a_1x_1 + \cdots + a_nx_n}{b_0 + b_1x_1 + \cdots + b_nx_n}$$

subject to $x \in F$ as in Problem A and denominator is always positive.

Main result of Megiddo is the following.

Theorem 3.4. *(Megiddo [14]) If Problem A is solvable within $O(p(n))$ comparisons and $O(q(n))$ additions then Problem B is solvable in time $O(p(n)(p(n) + q(n)))$.*

A natural idea is to consider Problem B and solve for a given parameter t, Problem A with $c_i = a_i - tb_i$. If v is the optimal value of Problem A and $v = tb_0 - a_0$ then t is the optimum value of Problem B and the x^* is optimum for both problems A and B. If $v < tb_0 - a_0$ then smaller t should be tested. If $v > tb_0 - a_0$ then smaller t should be tested.

This leads to a natural algorithm to find optimum t. Start with a t then increase or decrease t according to the value obtained by solving Problem A. This can lead to solving Problem A arbitrary large number of times. The key idea is to limit the number of times Problem A needs to be solved to to find the optimum t. Megiddo showed that the number of calls to Problem A is not more than number of comparisons made by the algorithm for solving Problem A.

Megiddo's algorithm simulates A-algorithm where the goal is a linear parametric function in t. Initially $t \in [-\infty, \infty]$. Then the feasible interval reduces throughout the execution and finally produces the optimum value. At each "comparison" or branching point reached in the execution of the A-algorithm, the corresponding critical value of t is tested by running the A-algorithm with t fixed at the critical value. Then the appropriate interval is selected and the next comparison point is considered. At the end, the optimal value of problem A will be given in the form of a linear function $v(t)$ defined over an interval $[e, f]$ which contains the optimum value. The optimum value is then calculated by solving $v(t) = tb_0 - a_0$. In particular only when optimum values are to be compared the A-algorithm needs to run with a numerical value. So the number of times the A-algorithm needs to run is equal to the number of comparisons that are used in the A-algorithm.

In our case for a given cut B, $T \in \mathcal{T}$ and edge $e \in T$, we have Problem B with:

1. $a_0 = w(e)$, and $a_f = w(f)$ for all edges $f \in E(G)$
2. $b_0 = -2$, $b_f = 1$ if $f \in T$, otherwise $b_f = 0$.
3. The set F is the set of cuts ($|E|$-dimensional 0-1 vectors such that edges corresponding to 1 form a cut).

This means, Problem A is the min-cut problem with capacities $c_j = a_j - tb_j$, which may include some negative c_j's. If the capacities are negative, then we have to solve the max-cut problem, which is NP-complete. However, we already have an upper bound for the value of the minimum ratio, namely t_{min} weight of the bottleneck edge in T. Hence, we may start by restricting t in the interval $I := [0, t_{\text{min}}]$. In this way, all the c_j's will be nonnegative and we will be able to solve the minimum cut problem required in Megiddo's algorithm. Min-cut can be solved using network flow algorithms in time order $O(|V|^3)$ and it uses at most $O(|V|^3)$ comparisons. Using binary search on the interval we can get a polynomial time algorithm in the size of the input. Using Megiddo's idea we get a strongly polynomial time algorithm in time $O(|V|^6)$.

Now we can solve Problem B for all edges in T using the above procedure. This can be done in time $O(|V|^6|E|)$. Let the minimum value of t over all these edges be t_1.

We also have to ensure that edges outside the trail T do not violate any cut. This is also a ratio minimization problem as in the previous case. The only change in Problem B will be $b_0 = 0$. This can be done in time $O(|V|^6)$ for each edge. Hence, for all edges not in T it will take time $O(|V|^6|E|)$. Let t_2 be the upper bound for t obtained by minimizing over all the edges not in T.

Let $t = \min(t_1, t_2)$. This is the t that we need to use in step 2 of Algorithm 1.

Now we need a few lemmas to find the time taken by Algorithm 1.

Lemma 3.2. *The time taken to verify the balance condition is $O(|V||E|)$.*

Proof. To check balance condition at a vertex it takes time $O(|E|)$. Repeating this for all vertices takes time $O(|V||E|)$. □

Lemma 3.3. *The time taken to verify the cut condition for the graph $G(E, V, w, c)$ is $O(|E||V|^3)$.*

Proof. We need to check for each edge $e = \{u, v\} \in E$ and every cut B with $e \in B$, that it does not violate cut condition. There are exponentially many such inequalities to verify. We accomplish this in polynomial time by finding the min weight $u - v$ cut B in $G - e$ using network flow algorithm. If $2w(e) \leq w(B)$ then the edge e satisfies the the cut condition for every u-v cut. This we need to do for every edge in G. Max-flow algorithm can find min-cut in time $O(|V|^3)$ ([13]). So this verification can be done in $O(|E||V|^3)$. □

Next we bound the time taken by step 2 of Algorithm 1.

Lemma 3.4. *The time taken to find a closed alternating trail in $G(V, E, w, c)$ is $O(|E|^3)$.*

Proof. Using the construction of Szabó we have a graph $G'(V', E')$ with number of vertices of $O(|E|)$ and edges $O(|E|^2)$. In this graph G' we need to find augmenting paths or a Tutte set. This takes time at most $|V'||E'|$ ([16]). Thus, we can find a closed alternating trail using Theorem 3.3 in time $O(|E|^3)$. □

Now, we can find how long it takes to execute step 2 of the algorithm once. To find a closed alternating trail it takes time $O(|E|^3)$. To find the maximum possible weight that can be subtracted we need time $O(|V|^6|E|)$. So the total time taken for executing step 2 once is $O(|V|^6|E|)$.

Each time step 2 is executed either the weight of an edge is reduced to zero or a tight cut with at least 3 vertices on each side is formed. If such a tight cut is formed we go to Step 1. Else we continue with step 2. So step 2 can be executed at most $O(|E|)$ times before step 1 is executed or the algorithm stops.

In step 1 to find a tight cut we use max-flow algorithm on $G - e$. This may be repeated $O(E)$ times. So this part in step 1 may take time $|V|^3|E|$. If the tight cut is found then two smaller problems are solved.

Let $T(n)$ be the time taken for the algorithm to run, where $n = |V|$. We may bound $|E|$ by n^2. Then the following recurrence relation is satisfied.

$$\begin{aligned} T(n) &= T(n-k+1) + T(k+1) + O(|V|^6|E|) \\ &= T(n-k+1) + T(k+1) + O(n^8). \end{aligned}$$

This shows that $T(n) = O(n^9)$.

The overall time can be improved by a factor of n by using appropriate data structures and speedup of Megiddo's algorithm.

This completes the proof of Theorem 3.2.

An illustration of the algorithm is given in the Figure 5. In this example, we first remove the outer CAT. Then the resulting graph has a tight-cut. Then we reduce our problem to two smaller graphs and solve them recursively. Finally we join the solutions for smaller graphs to solve the problem for the original graph.

In [7] it was conjectured that if G is a integer sum of circuits and balanced then it is also a sum of closed alternating trails. In [18] it was pointed out that it is indeed the case.

It is natural to consider integer sum of closed alternating trails. Seymour considered the question integer sum of circuits. It seems that if all weights are integers and weight of every cut is even then the graph can be written as an integer sum of circuits. This is not so. For example consider the Peterson graph with weights and colors as follows. Let the unique perfect matching have weight 2 and all other edges have weight 1. Color the edges in the matching red and all other edges blue. See Figure 6.

Seymour used four color theorem to show that if a graph is planar with integer weights, and satisfies the cut condition and all cuts have even weight then it can be written as a sum of circuits.

It seems that Petersen graph is the main obstacle to integer sum of circuits. In [1] it was shown that if a graph has integer weights, satisfies the cut conditions, every cut has even weight and does not contain a blistered Peterson graph (essentially the example given) as a minor then it can be written as an integer sum of circuits.

Using observations in [18] it follows that the corresponding result will hold for colored graphs.

We end with the following conjecture.

Conjecture 1. *If a graph is balanced, all edge weights are even and satisfies the cut condition then it is integral sum of closed alternating trails.*

A simple case of this is the cycle double cover conjecture, where the weight of every edge is 2. In this case itself the problem reduces to 'snarks' and it gets very difficult to solve. Possibly the algebraic or integer programming ideas outlined in [18] will be more tractable.

References

[1] B. Alspach, L. Goddyn and C. Zhang, Graphs with the Circuit Cover Property, *Transactions of the American Mathematical Society*, Vol. 344, No. 1 (Jul., 1994), pp. 131-154

[2] E. M. Arkin and C. H. Papadimitriou. On the complexity of circulations. *J. Algorithms*, 7:134-145, 1986.

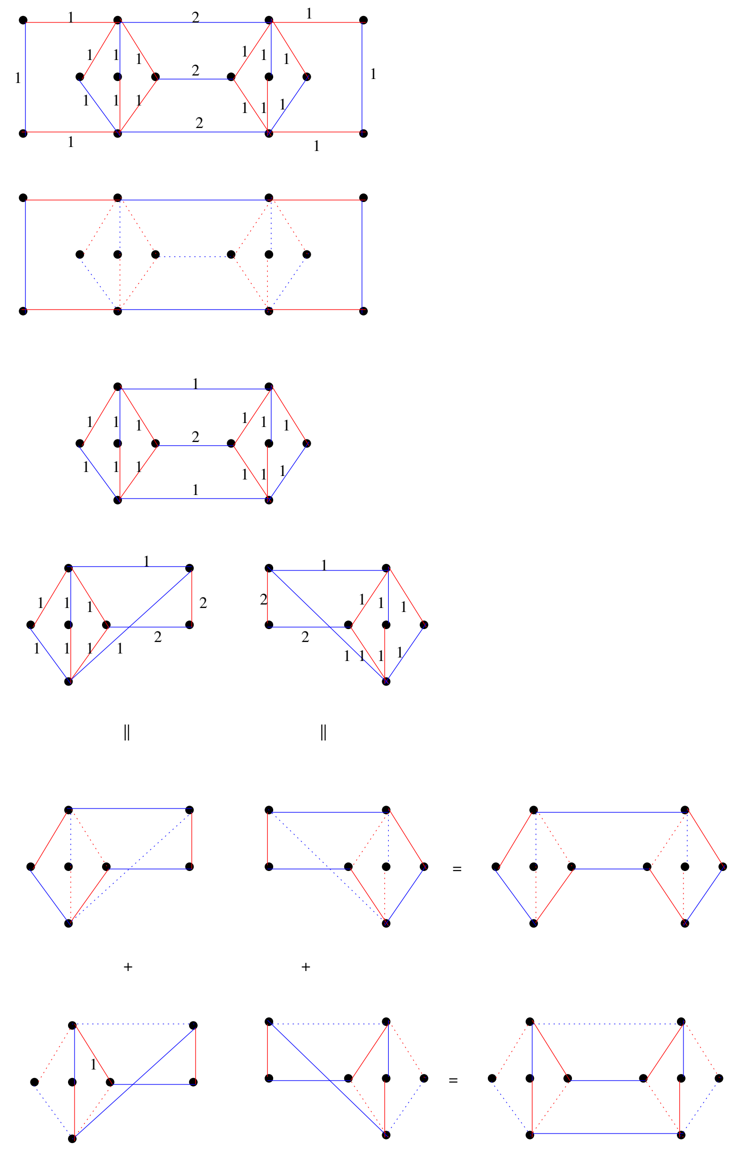

Figure 5

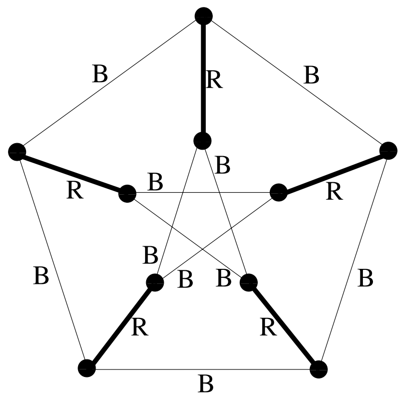

Figure 6: The bold edges have weight 2 and the rest of the edges have weight 1.

[3] L.W. Beineke, Derived graphs and digraphs, *Beitrage zur Graphentheorie, Leipzig*, (1968) 17–33.

[4] C. Berge, *Hypergraphs. Combinatorics of Finite Sets*, North-Holland Mathematical Library, Amsterdam, 1989.

[5] J.C. Bermond and J.C. Meyer, Graphs representatif des arêtes dun multigraphe, *J. Math. Pures Appl.*, **52** (1973) 299–308.

[6] A. Bhattacharya, U. N. Peled, and M. K. Srinivasan, Cones of closed alternating walks and trails, *Linear Algebra and its Applications* **423**: 351–365, (2007).

[7] A. Bhattacharya, U. N. Peled, and M. K. Srinivasan, The cone of balanced subgraphs, bf 431: 266–273,(2009).

[8] A. Bhattacharya, U. N. Peled, and M. K. Srinivasan, Alternating Reachability, *arXiv:math/0511675v2.*

[9] J. Edmonds, Paths, trees, and flowers, *Canadian Journal of Mathematics* **17**: 449–467, (1965).

[10] J. Krausz, Demonstration nouvelle d'un théorème de Whitney sur les réseaux, *Mat. Fiz. Lapok*, 50 (1943), pp. 75–89.

[11] L. Lovász, Problem 9, *Beiträge zur Graphentheorie und deren Anwendungen : Vorgetragen auf dem international kolloquium*, Oberhof (1977) P. 313.

[12] L. Lovász and M. D. Plummer, *Matching theory*, Annals of Discrete Mathematics **29** (1986).

[13] V .M. Malhotra, M.Pramodh Kumar, S. N. Maheshwari, An $O(V^3)$ algorithm for finding maximum flows in networks *Information Processing Letters*, Volume 7, Issue 6, October 1978, Pages 277-278.

[14] N. Megiddo, Combinatorial optimization with rational objective functions, *Mathematics of Operations Reasearch* **4** (4) : 414–424, (1979).

[15] Y. Metelsky and R. Tyshkevich, *On Line Graphs of Linear 3-Uniform Hypergraphs*, On line graphs of linear 3-uniform hypergraphs. J. Graph Theory, 25 (1997) 243–251.

[16] S. Micali, V. Vazirani, An $O(V^{\frac{1}{2}}E)$ algorithm for finding maximum matching in general graphs, *21st Annual Symposium on Foundations of Computer Science*, 1980 IEEE Computer Society Press, New York. pp. 17–27.

[17] R.N. Naik, S.B. Rao, S.S. Shrikhande and N.M. Singhi, Intersection graphs of k-uniform linear hypergraphs, *European J. Combin.*, **3** (1982)159–172.

[18] S. Petrović, A survey of discrete methods in (algebraic) statistics for networks, *arXiv:1510.02838.*

[19] A. Schrijver, *Theory of Linear and Integer Programming,* Wiley, 1998.

[20] P. D. Seymour, Sums of Circuits, *Graph Theory and Related Topics, Eds: J. A. Bondy and U. S. R. Murty*, Academic Press, New York, 341–355, (1979).

[21] P.V. Skums, S.V. Suzdal and R.I. Tyshkevich, Edge intersection graphs of linear 3-uniform hypergraphs, *Discrete Mathematics*, **309** (2009) 3500–3517.

[22] W. T. Tutte, *Graph theory,* Addison-Wesley, (1984).

[23] W. T. Tutte, The method of alternating paths, *Combinatorica* **2** (3) : 325–332, (1982).