Maximum Flows on Disjoint Paths

Guyslain Naves*, Nicolas Sonnerat† and Adrian Vetta‡

September 8, 2010

Abstract. We consider the question: What is the maximum flow achievable in a network if the flow must be decomposable into a collection of edge-disjoint paths? Equivalently, we wish to find a maximum weighted packing of disjoint paths, where the weight of a path is the minimum capacity of an edge on the path. Our main result is an $\Omega(\log n)$ lower bound on the approximability of the problem. We also show this bound is tight to within a constant factor. Surprisingly, the lower bound applies even for the simple case of undirected, planar graphs.

Our results extend to the case in which the flow must decompose into at most k disjoint paths. There we obtain $\Theta(\log k)$ upper and lower approximability bounds.

1. Introduction

Network flows have played a fundamental role in the advancement of combinatorial optimization [14] and are ubiquitous in applications [2]. In the standard single-commodity flow problem we have a capacitated graph $G = (V, E)$ and terminal vertices s and t.[4] The goal is to find a maximum valued flow from s to t that satisfies the capacity constraints on each edge. Equivalently, we are searching for a maximum packing of weighted $s - t$ paths; the packing constraints simply state that the total weight of all paths passing through an edge must not exceed the *capacity/weight*, w_e, of that edge. Viewed in this light, a special case is the classical problem of finding a maximum collection of disjoint paths.

Thus, there has been a long-standing and close relationship between network flows and the disjoint packing of *unweighted* paths. An immediate question arises: what about the weighted case, namely, what if we desire that our network flow decomposes into a disjoint collection of *weighted* paths? Surprisingly given the apparent simplicity of the question, as far as we are aware, this question has not previously been considered in the literature.

Consequently, this paper investigates how to find a maximum flow whose path decomposition consists only of disjoint paths. Specifically, take a collection of pairwise edge-disjoint $s - t$ paths $\mathcal{P}$. Then the maximum flow, $w(P)$, we can send down a path $P \in \mathcal{P}$ is simply the minimum capacity of an edge in P. The value of the flow is then the sum of the flows along each path, $\sum_{P \in \mathcal{P}} w(P)$. Our goal is to obtain a flow $\mathcal{P}$ consisting of edge-disjoint paths that has maximum value. We call this the *Disjoint Weighted Flow Problem*.

*Department of Mathematics and Statistics, McGill University. Email: naves@math.mcgill.ca

†Department of Mathematics and Statistics, McGill University. Email: sonnerat@math.mcgill.ca

‡Department of Mathematics and Statistics, and School of Computer Science, McGill University. Email: vetta@math.mcgill.ca

[4]Of course, by incorporating a supersource and supersink, this framework also models the case of multiple sources and sinks, provided any source can route to any sink.

Observe this problem does indeed correspond to a simple weighted path-packing problem. Specifically, let the weight of an $s-t$ path P be $w(P) := \min\{w(e) \mid e \in E(P)\}$. Then we are looking for a collection $\mathcal{P}$ of disjoint paths of maximum total weight, that is $\sum_{P\in\mathcal{P}} w(P)$ as before. As well as being an elegant combinatorial question, we remark that this requirement for disjoint paths is also a natural one in applications where flow paths can interfere with one another or where technological constraints at links and nodes compel disjointness.

In this paper, we examine approximation algorithms for the disjoint weighted flow problem. We present $\Theta(\log n)$ lower and upper approximation bounds for the weighted disjoint paths problem, where n is the number of vertices. Standard reductions show that these bounds also apply in directed graphs and/or if we insist the paths be vertex-disjoint rather than edge-disjoint. Furthermore, our lower bound applies even for the special case of planar graphs.

1.1. Related Work.

Given the applicability of network flows there is a vast literature optimizing flows given additional constraints. These side-constraints may arise from the application itself, but they can also arise due to restrictions induced by available technology or by the choice of routing protocol; see [13] for a survey illustrating some of these issues. The work most closely related to our own, though, concerns k-splittable flows introduced by Baier, Köhler and Skutella [8]. A k-splittable flow is a flow that can be routed along k paths - note that these paths are not required to be disjoint. Thus, Kleinberg's unsplittable flows [10] can be viewed as 1-splittable flows. Baier et al. present a 2-approximation algorithm for the k-splittable single-commodity flow problem.

Our results also extend to the case in which feasible solutions must be decomposable into at most k disjoint paths, for some k. Disjoint weighted flows, however, are harder to deal with and approximate than k-splittable flows. In particular, for single-commodity flows, we obtain $\Theta(\log k)$ lower and upper approximations bounds when we are constrained to use at most k disjoint paths.

2. The Lower Bound

In this section we present our main result:

Theorem 1. *For undirected planar networks, the hardness of approximation for the maximum disjoint weighted flow problem is $\Omega(\log n)$, unless $P = NP$.*

Before proving Theorem 1 we outline the structure of the proof. First, we introduce a graph G_N that has a maximum disjoint weighted flow of value equal to the harmonic number $H_N \approx \log N$. But if we use a slightly modified weight function for the paths then G_N has a maximum disjoint weighted flow of value one.

We then build a new network $\mathcal{G}$ by replacing each node of G_N by an instance of an NP-hard routing problem. The routing problem will be chosen to have the following properties. If it is a YES-instance then path weightings for the disjoint weighted flow problem on $\mathcal{G}$ will correspond to the original weighting scheme on G_N. In contrast, if it is a NO-instance, then path weightings for the disjoint weighted flow problem on $\mathcal{G}$ will correspond to the modified weighting scheme on G_N.

It follows that an approximation algorithm with guarantee better than logarithmic would allow us to distinguish between YES- and NO-instances of our routing problem, giving a lower bound of $\Omega(\log N)$. We will see that this bound is equal to $\Theta(\log n)$.

Furthermore, at all stages we will show this reduction can be applied using only undirected, planar graphs. Theorem 1 will follow.

2.1. A Half-Grid Graph.

Let's begin by defining the graph G_N. There are N rows (numbered from top to bottom) and N columns (numbered from left to right). All the edges in the ith row and all the edges in the ith column have weight $\frac{1}{i}$. The ith row extends as far as the ith column and vice versa; thus, we obtain a "half-grid" that is a weighted version of the network considered by Guruswami et al [9]. Finally we add a source s and a sink t. There are edges of weight $\frac{1}{i}$ from s to the first vertex in row i and from t to the last vertex in column i. The complete construction is shown in Figure 1.

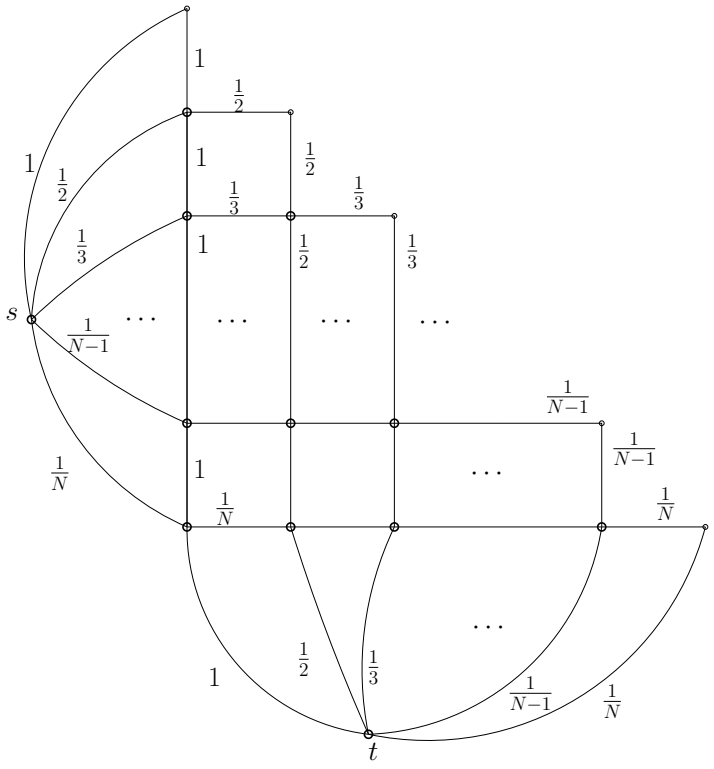

Figure 1. Grid Graph G_N.

Note that there is a unique $s-t$ path P_i consisting only of edges of weight $\frac{1}{i}$, that is, the L-shaped path that goes from s along the ith row and then down the ith column to t. Moreover, for $i \neq j$, the path P_i intersects P_j precisely once. Clearly each path P_i has weight $w(P_i) = \frac{1}{i}$, so the collection of edge-disjoint paths $\mathcal{P}^* = \{P_1, P_2, \ldots, P_N\}$ gives a flow of total value $H_N = 1 + \frac{1}{2} + \ldots \frac{1}{N}$. Since every edge incident to s is used in $\mathcal{P}^*$ with its maximum weight, this solution is optimal. Similarly, if we are constrained to use flows that decompose into at most k disjoint paths then the optimal flow has weight H_k.

Now consider what happens when we modify the weight function for the paths. Given a collection $\mathcal{P}$ of paths, let the *modified weight*, $\hat{w}_{\mathcal{P}}(P)$, of a path $P \in \mathcal{P}$ be the the minimum weight amongst its edges and those edges incident to a vertex at which P crosses another path $Q \in \mathcal{P}$. Formally,

$$\hat{w}_{\mathcal{P}}(P) = \min\{w_{uv} \,|\, v \in P, uv \in Q \text{ for some } Q \in \mathcal{P}\}$$

where we will omit the subscript if $\mathcal{P}$ is clear.

The maximum value of a flow is significantly reduced if we use these modified weights.

Lemma 2. *The maximum modified value of a weighted flow in G_N is 1.*

Proof. Define the *rank* of a path P to be the index j for which this path uses the weight $\frac{1}{j}$ edge incident to t. Suppose $\frac{1}{i}$ is the maximum modified value of any path in a flow $\mathcal{P}$. Let j be the rank of some path $Q \in \mathcal{P}$ of modified weight $\frac{1}{i}$. Then set $\mathcal{P}^+$ to be the collection of paths in $\mathcal{P}$ with ranks greater than j, and $\mathcal{P}^-$ to be the paths with ranks less than j.

Observe that Q must contain as a sub-path all the edges in column j that lie below row i. Otherwise, Q would contain an edge in a row of lower weight than $\frac{1}{i}$, contradicting the fact that Q has modified weight $\frac{1}{i}$. Similarly, no other other path in $\mathcal{P}$ crosses Q on this sub-path, as this would reduce Q's modified weight. This implies that any path in $\mathcal{P}^+$ must use one edge of the columns j to $i+1$ between row i and row $i+1$. Consequenty, $|\mathcal{P}^+| \leq i-j$. Obviously $|\mathcal{P}^-| \leq j-1$ and so $|\mathcal{P}| \leq 1+(i-j)+(j-1) = i$. Since each path has modified weight at most $\frac{1}{i}$, this gives an upper bound of 1 on the modified value of the flow.

For $\mathcal{P}^* = \{P_1, P_2, \ldots, P_N\}$, we see that $\hat{w}(P_i) = \frac{1}{N}$, for all i. Thus, this collection of paths obtains the maximum modified value of one. □

2.2. The 2-Edge-Disjoint Weighted Paths Problem.

Recall the next step is to replace, in G_N, each vertex at the crossing of two paths P_i and P_j with an instance of an NP-hard routing problem. To define this routing problem, let H be an undirected graph whose edges have weight either a or b, where $b > a$. Given two pairs of vertices (s_1, t_1) and (s_2, t_2), we wish to find a path P_1 from s_1 to t_1 and a path P_2 from s_2 to t_2 with the properties that
(i) P_1 and P_2 are edge-disjoint.
(ii) P_2 may only use edges of weight b (P_1 may use either weight a or weight b edges).
We call this the *Two Edge-Disjoint Weighted Paths Problem*, or 2-EDWP.

Evidently, we will be most interested in the case where the graph H is planar. Then we have:

Theorem 3. Planar-2-EDWP *is NP-hard, even if the pairs of terminals lie on the outer face of H in the order s_1, s_2, t_1, t_2.*

We remark that in graphs which are directed and not planar, the hardness of 2-EDWP follows directly from the hardness of the 2-arc-disjoint paths problem ([7]).

Before embarking on the proof, observe that Theorem 3 immediately tells us that the maximum disjoint weighted flow problem is hard in planar graphs. Simply take an instance of Planar-2-EDWP and add a super-source s and a super-sink t. Then

connect s to s_1 and s_2 with edges of weights a and b, respectively. Similarly, connect t to t_1 and t_2 with edges of weights a and b, respectively. Then there is a disjoint weighted $s-t$ flow of value $a+b$ if and only if there are paths P_1 and P_2 satisfying properties (i) and (ii). Of course, we desire a much stronger hardness result than this, but this observation will be useful in motivating the subsequent construction.

In order to prove Theorem 3, we will need a geometric result. An edge $uv \in E(G)$ is a *separating edge* if $G - \{u, v\}$ is disconnected. A triangle $uv, vw, wu \in E(G)$ is a *separating triangle* if $G-\{u, v, w\}$ is disconnected. We will need the following theorem:

Theorem 4 (Whitney [15], 1931)**.** *Every maximal planar graph with no separating triangle is Hamiltonian.*

It was shown in [3] that such a Hamiltonian cycle can be computed in linear-time.

Lemma 5. *Let $G = (V, E)$ be an embedded planar graph, such that G has girth 4 and has no separating edge. Let $\phi_e \subseteq \mathbb{R}^2$ be the open curve corresponding to the embedding of e, for each edge $e \in E$. Then there is a simple closed curve in $\mathcal{R} = \mathbb{R}^2 \setminus \bigcup_{e\in E} \phi_e$ that intersects the image of every vertex.*

Proof. We define the graph G' obtained form G by adding a new vertex in each face. Each of these new vertices is adjacent to every vertex on the boundary of its face. G' is then obviously a maximal planar graph. In order to apply Theorem 4 to G', we must prove that G' does not contain a separating triangle. First assume it. Then we get a Hamiltonian cycle on G'. By slightly shifting this Hamiltonian cycle, we get a curve intersecting G' exactly once on each vertex, and only there. Then, the lemma is proved by simply ignoring the new vertices.

To conclude the proof, we show that G' has no separating triangle. We denote by E' the set of new edges. First, every triangle T has exactly two edges in E', because G has girth 4 (so $|T \cap E'| \neq 0$), the graph induced by E' is bipartite (so $|T \cap E'| \neq 3$), and there is no edge in E' with both extremities on $V(G)$ (so $|T \cap E'| \neq 1$). Suppose that T were a separating triangle, and let e be its edge in G. Then each component of $G' - T$ would contain a vertex of G, since each new vertex is still adjacent to a vertex of G. So e would be a separating edge of G, contradicting the assumption. □

In the following, we identify vertices, edges and graphs with their respective images on the plane. For $\gamma \in \{a, b\}$, we call an edge of weight γ a *γ-edge*.

Proof of Theorem 3. We give a reduction from Planar-3-Sat to Planar-2-EDWP. Let $\mathcal{C}$ be a set of clauses over the variables $\mathcal{X}$, such that the bipartite graph $G = (\mathcal{X} \cup \mathcal{C}, \{xC \ : \ x \in \mathcal{X}, C \in \mathcal{C}, x \in C \vee \overline{x} \in C\})$ is planar. First, we may assume that G has no separating edges. Because if e is a separating edge, let W_1 and W_2 be two components of $G-e$ sharing a common face. Each of W_1 and W_2 has a variable vertex on the boundary of that common face (as every cycle alternates between variables and clauses), denote them x_1 and x_2 respectively. Then, add a new variable z, and two new clauses with value $x_1 \vee x_2 \vee z$. This can clearly be done without violating the planarity, nor modifying the satisfiablity of the formula. By repeating this argument, we will finally get a graph without a separating edge.

Without loss of generality, we can also suppose that each variable appears at most three times. To see this, observe that if x appears in $k \geq 4$ clauses we may introduce k

new variables, $x_1, \ldots, x_k$, and new clauses $\overline{x_1} \vee x_2$, $\overline{x_2} \vee x_3$, $\ldots$, $\overline{x_k} \vee x_1$, and replace each occurrence of x by an occurrence of one of the x_i. We can also assume that each variable appears exactly once negatively. These transformations can clearly be implemented whilst preserving the planarity of G and without creating separating edges. Thus, we obtain a formula whose corresponding bipartite graph G has maximum degree 3 and has no separating edge.

Now take a planar embedding for G. By Lemma 5, we may find a closed curve $\mathcal{D}$ intersecting the embedding of G exactly on its vertices.

We will transform $(G, \mathcal{D})$ into an instance of Planar-2-EDWP in polynomial-time. To do this, we need to build an auxiliary edge-weighted planar graph G' for the routing problem. Towards this goal, we first take G and use $\mathcal{D}$ to induce an additional set of embedded a-edges whose endpoints are in $V(G)$.

Then we replace each edge $e = uv \in E(G)$ by a 4-cycle consisting of b-edges us_e, ut_e, vs_e, vt_e, where s_e and t_e are new vertices.

Next we replace each *variable* vertex $x \in \mathcal{X}$ by a variable gadget and each *clause* vertex by a clause gadget. Each variable vertex x of degree three is replaced by one of four possible *variable gadgets*; the actual choice is dependent upon the relative position of $\mathcal{D}$ with respect to the edges incident to x and upon the sign of x in the adjacent clauses. These four gadgets are illustrated in Figure 2, where the edges corresponding to $\mathcal{D}$ and the other a-edges are dashed, the edges corresponding to $E(G)$ and the other b-edges are bold (recall there must be two edges out of the gadget for each edge in G as we initially replaced such edges by a 4-cycle). The $+$ and $-$ signs indicate whether the variable appears positively or negatively in the adjacent clause.

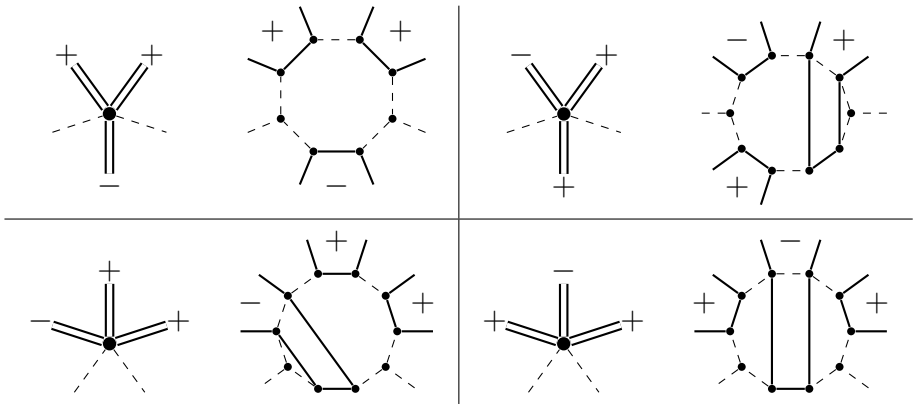

Figure 2. Variable gadgets.

Each clause vertex C of degree three is replaced by one of two possible *clause gadgets*; again, the actual choice is dependent upon the relative position of $\mathcal{D}$ with respect to the edges incident to C. These two gadgets are shown in Figure 3. The gadgets for clauses with two literals and for variables occuring only twice are similar to those presented, but simpler.

To complete the construction we need to specify the sources and the sinks. To do so, we first specify a multicommodity flow formulation with many source-sink pairs. Later we will show how to implement it as a flow with just two source-sink pairs.

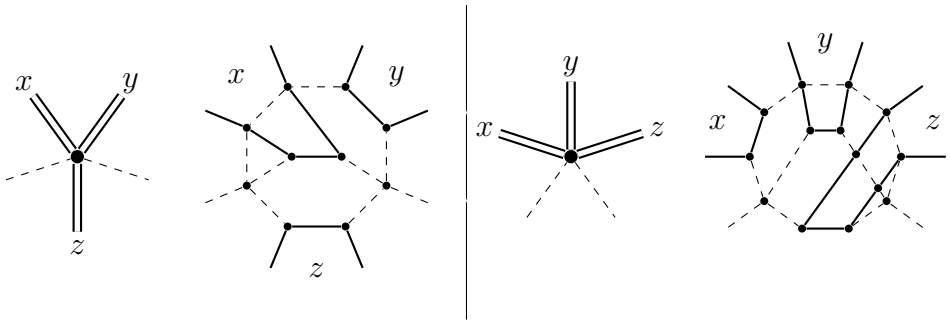

FIGURE 3. Clause gadgets, with the same convention as in Figure 2.

Towards the former goal, we will have a source-sink pair (s_e, t_e) for each edge $e \in G$. Furthermore, we will have one additional source-sink pair (s_a, t_a). To define this pair, arbitrarily choose one of the edges uv of $\mathcal{D}$. Then replace uv by two edges us_a and vt_a each with weight a. Observe that s_a and t_a are on the boundary of a common face of the resultant planar graph G'.

This multicommodity flow problem relates to the planar 3-SAT instance in the following manner.

Claim 6. *The formula is satisfiable if and only if there are edge-disjoint paths $\{P_e\}_{e\in E(G)}$ and Q in G', with the following properties.*

(*i*) *P_e has endpoints s_e and t_e and uses only b-edges.*
(*ii*) *Q has endpoints s_a and t_a.*

Proof. First, note the 4-cycles of b-edges that initially replaced each edge have become larger under the construction but are still b-cycles. Moreover, these b-cycles (call them H_e, for each $e \in G$) are edge disjoint and their union covers all of the b-edges in G'.

Now suppose that all the paths exist. There are only two possible routes in H_e between s_e and t_e that P_e can take; if $e = xC$ then one route passes through the variable gadget x and the other passes through the clause gadget C. Since s_e and t_e have degree two, it follows immediately that Q cannot use any of the edges incident to them. Consequently, Q must *follow* the curve $\mathcal{D}$. We will show how to obtain from Q a satisfying truth-assignment.

For any edge $e = xC$, we say that the cycle H_e is *positive* if x appears positively in C, *negative* otherwise. Then, for a variable gadget x, it is easy to see that if Q does not intersect the unique negative cycle going through the gadget then it must use at least one edge of each of the positive cycles H_e going through that gadget. If it intersects the negative cycle, set variable x to *true*, otherwise set it to *false*.

To see that this does produce a satisfying assignment, take any clause C, say over the variables x, y and z. Since Q follows $\mathcal{D}$ it must pass through each clause gadget. Consequently, Q intersects at least one of H_{xC}, H_{yC}, and H_{zC}. Without loss of generality, let it intersect H_{xC}. This means that P_{xC} cannot go through the clause gadget C and, hence, must go through the variable gadget x. But, again, as Q follows $\mathcal{D}$ it must pass through the variable gadget x too. Therefore, Q cannot intersect H_{xC} in the variable gadget x. This precisely means that x is *true* if x appears positively in C, and x is *false* if it appears negatively. So C is satisfied by x.

On the other hand, given a satisfying assignment, it is easy to find a collection of feasible paths. This is because, for each variable gadget, there is a sub-path that intersects only the positive cycles in that gadget and there is a sub-path that intersects only the negative cycle. Therefore, Q can always follow the appropriate sub-path. □

To complete the proof of Theorem 3 we need to reduce the number of commodities in the flow to two. For this, we will keep the source-sink pair (s_a, t_a) but group into one all of the pairs (s_e, t_e) via the use of a new source sink pair (s_b, t_b). To accomplish this, we first need to position the new vertices s_b and t_b in G'. Let $\mathcal{B}$ be a closed curve that intersects G' on s_a and t_a only. Then add s_b arbitrarily on the "upper" path between s_a and t_a induced by $\mathcal{B}$. Similarly add t_b on the "lower" path between s_a and t_a induced by $\mathcal{B}$.

Our goal now is to force any path of b-edges between s_b and t_b to follow b-paths between s_e and t_e for every $e \in G$. To do this, let $e_1, e_2, \ldots, e_m$ be any ordering of the edges of G. For a cycle H_e, we define its *inside* as the connected component of $\mathbb{R}^2 \setminus H_e$ that does not contain any vertex of G'. Then set $\mathcal{R}$ to be the union of the inside of every cycle H_e plus $V(G')$ and the inside of $\mathcal{B}$. Observe that $\mathcal{R}$ is a union of disjoint balls, so its complement is connected. Let P be a path between s_b and s_{e_1} in this complement. Build a path of b-edges along P and add them to G', inserting new vertices whenever P crosses an a-edge (note that these are the only edges P can cross). Next add P to $\mathcal{R}$; this does not change the connectedness of its complementary set. In this manner, we may iteratively add paths of b-edges between t_i and s_{i+1}, for $1 \leq i \leq m-1$, and finally between between t_m and t_b. By construction, these paths are disjoint and cross only a-edges. We thus obtain a new planar graph G'' with four terminals on the same face, as desired.

Clearly this new instance of Planar-2-EDWP is equivalent to the previous multi-commodity flow problem. To see this, simply note that the new b-edges are isthmi in the subgraph consisting of the b-edges. Consequently, the (s_b, t_b)-path must use each of these new b-edges and then, as before, in each H_e route through either the variable gadget or through the clause gadget. This completes the reduction. □

2.3. The Hardness Result.

We can now complete the proof of the approximation hardness. Observe that any vertex of degree four in G_N is incident to two edges of weight $\frac{1}{i}$ and to two edges of weight $\frac{1}{j}$, for some $i \neq j$. We construct a graph $\mathcal{G}$ by replacing each vertex of degree four with the routing graph H. We do this in such a way that the weight $\frac{1}{i}$ edges of G_N are incident to s_1 and t_1, and the weight $\frac{1}{j}$ edges are incident to s_2 and t_2. Moreover, for that copy of H placed at the intersection of P_i and P_j, we then let $a = \frac{1}{i}$ and $b = \frac{1}{j}$, where we may assume that $j < i$.

The hardness result will follow once we see how this construction relates to the original and modified weight functions.

Lemma 7. *If H is a YES-instance then the optimal disjoint weighted flow in $\mathcal{G}$ has value H_N. If H is a NO-instance then the optimal disjoint weighted flow in $\mathcal{G}$ has value at most* 1.

Proof. It is clear that if H is a YES-instance, then paths in $\mathcal{G}$ induce paths in G_N which are free to cross at any vertex without restrictions on their values. This means we obtain a flow of value H_N by using the canonical paths P_i, $1 \leq i \leq N$.

However, if H is a NO-instance, then it contains only an $s_1 - t_1$ path, or only an $s_2 - t_2$ path, or the $s_2 - t_2$ path is forced to use a lower weight a-edge. This implies that the induced paths $\mathcal{P}$ in G_N either do not cross at all, or if they cross then the weight of the path using the $\frac{1}{j}$-edge is forced down to a weight of $\frac{1}{i}$ (recall $j < i$). But this means that the weight of a path is upper bounded by the modified weight function $\hat{w}$. This allows us to apply Lemma 2, and hence the value of an optimal flow in this case is at most 1. □

Proof of Theorem 1. It follows that if we could approximate the maximum disjoint weighted flow problem in $\mathcal{G}$ to a factor better than H_N, we could determine whether the optimal solution is 1 or H_N. This in turn would allow us to determine whether H is a YES- or a NO-instance.

Note that $\mathcal{G}$ has $n = \Theta(pN^2)$ edges, where $p = |V(H)|$. If we take $N = \Theta(p^{\frac{1}{2}(\frac{1}{\epsilon}-1)})$, where $\epsilon > 0$ is a small constant, then $\log n = \Theta(H_N) = \Theta(\log p)$. This gives our lower bound of $\Omega(\log n)$. □

Similarly, if we are restricted to consider only flows that decompose into k disjoint paths then it is not hard to see that:

Theorem 8. *For undirected, planar networks, there is a $\Omega(\log k)$ hardness of approximation, unless $P = NP$, for the problem of finding a maximum flow that decomposes into at most k edge-disjoint paths.* □

3. An Approximation Algorithm

Our lower bound is tight to within a constant factor - there is a simple approximation algorithm that gives an almost matching upper bound.

Theorem 9. *For any network, there is an $O(\log n)$ approximation algorithm for the maximum disjoint weighted flow problem.*

Proof. To begin, round each edge weight down to the nearest power of 2. This can only cost us a factor 2 in our approximation guarantee. Next, we claim that we may assume that every edge weight lies between $1 = 2^0$ and 2^t where $t = 1 + \lceil \log n \rceil$. To see this, first note that there can be at most n edge-disjoint $s - t$ paths in any flow. Therefore, for any j, the total contribution from all paths that contain an edge of weight 2^j or less is upper bounded by $n2^j$. Now, let 2^{j_0} be the highest edge weight such that there exists a path of weight 2^{j_0}. Deleting the edges of weight 2^j for all indices j where $2^j < \frac{1}{n}2^{j_0-1}$ loses us at most 2^{j_0-1} in weight, that is, half of the optimal flow value. The lowest remaining edge weight, 2^{j_1}, then satisfies $j_1 \geq j_0 - 1 - \lceil \log n \rceil$. Scaling down the edge weights by a factor 2^{j_1} gives the claim.

The approximation algorithm now proceeds as follows. For each i such that $0 \leq i \leq t = 1 + \lceil \log n \rceil$, let E_i be the edges of weight at least 2^i, and let $G_i = (V, E_i)$. Let ϕ_i be the maximum number of edge-disjoint $s - t$ paths in G_i. Clearly, these paths induce a weighted disjoint flow of value at least $2^i\phi_i$ in G. Furthermore the optimal weighted disjoint flow must have value at most $\sum_{i=0}^{t} 2^i\phi_i$. To see this, note that the paths of

weight 2^i in the optimal solution together form a feasible solution for the disjoint paths problem in G_i. Then, since $t = 1 + \lceil \log n \rceil$, one of the G_i produces a weighted disjoint flow whose value is at least a logarithmic fraction of the optimal flow value. As we can easily solve the maximum disjoint paths problem in G_i in polynomial time, this gives the claimed $O(\log n)$ approximation algorithm. □

Corollary 10. *There is an $O(\log k)$ approximation algorithm for the problem of finding a maximum flow that decomposes into at most k edge-disjoint paths.*

Proof. This previous argument applies. The approximation guarantee, however, improves to $O(\log k)$ because now the paths of weight at most 2^j can only contribute a total value of at most $k2^j$. □

4. Conclusion

We have given approximation guarantees for the maximum disjoint weighted flow problem in single-commodity networks. Therefore, a natural question would be to look at the multi-commodity case, where we wish to find weighted flows between s_i and t_i, for $i = 1, \ldots, k$, that are disjoint and maximize total weight. By the techniques of Section 3, we can easily obtain an upper bound of $O(\alpha \log n)$, where α is the approximation achievable in the unweighted case. Unfortunately, the unweighted version is extremely hard to approximate since it is the edge-disjoint paths problem studied by Guruswami et al. [9]. They show this problem is inapproximable to within $\alpha = m^{\frac{1}{2}-\epsilon}$, for any $\epsilon > 0$, in directed graphs and give an approximation algorithm that essentially matches this lower bound.

In addition, given that our lower bound is essentially tight, the search for bi-criteria results is of interest. Here we would relax the condition that the paths in a weighted flow be strictly disjoint; instead, one would allow a limited amount $c \geq 2$ of congestion on each edge. For multi-commodity flows, the unweighted version of the problem has recently been studied extensively; for ground-breaking results in this area, see Chekuri et al. [4] for upper bounds in planar graphs, and Andrews et al. [1] and Chuzhoy et al. [5] for lower bounds in general graphs.

References

[1] M. Andrews, J. Chuzhoy, V. Guruswami, S. Khanna, K. Talwar and L. Zhang, "Inapproximability of edge-disjoint paths and low congestion routing on undirected graphs", *Electronic Colloquium on Computational Complexity*, **14:113**, 2007.

[2] R. Ahuja, T. Magnanti and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, 1993.

[3] T. Asano, S. Kikuchi and N. Saito, "An efficient algorithm to find a Hamiltonian circuit in a 4-connected maximal planar graph", *Graph Theory and Algorithms*, pp182-195, 1981.

[4] C. Chekuri, S. Khanna and B. Shepherd, "Edge-disjoint paths in planar graphs with constant congestion", *SIAM J. Computing*, **39(1)**, pp281-301, 2009.

[5] J. Chuzhoy, V. Guruswami, S. Khanna, and K. Talwar, "Hardness of routing with congestion in directed graphs", *Proceedings of the 39th ACM Symposium on Theory of Computing (STOC)*, pp, 2007.

[6] Y. Dinitz, N. Garg and M. Goemans, "On the single-source unsplittable flow problem", *Combinatorica*, **19**, pp17-41, 1999.

[7] S. Fortune, J. Hopcroft and J. Wyllie, "The directed subgraph homeomorphism problem", *Theoretical Computer Science*, **10**, pp111-121, 1980.

[8] G. Baier, E. Kohler and M. Skutella, "The k-splittable flow problem", *Algorithmica*, **42**, pp231-248, 2005.

[9] V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd and M. Yannakakis, "Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems", *Journal of Computer and System Sciences*, **67(3)**, pp473-496, 2003.

[10] J. Kleinberg, "Single-source unsplittable flow", *Proceedings of the 37th on Foundations of Computer Science (FOCS)*, pp68-77, 1996.

[11] R. Koch and I. Spenke, "Complexity and approximability of k-splittable flows", *Theoretical Computer Science*, **369**, pp338-347, 2006.

[12] F. Salazar and M. Skutella, "Single-source k-splittable min-cost flows", *Operations Research Letters*, **37(2)**, pp71-74, 2009.

[13] B. Shepherd, "Single-sink multicommodity flow with side constraints", *Research Trends in Combinatorial Optimization*, W. Cook, L. Lovasz, J. Vygen (Editors), Springer, pp429-450, 2009.

[14] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, Springer, 2003.

[15] H. Whitney, "A theorem on graphs", *Annals of Mathematics*, **32(2)**, pp378-390, 1931.