Unit Capacity Maxflow in Almost $O(m^{4/3})$ Time

Tarun Kathuria  
EECS  
University of California Berkeley  
Berkeley, USA  
tarunkathuria@berkeley.edu

Yang P. Liu  
Department of Mathematics  
Stanford University  
Stanford, USA  
yangpliu@stanford.edu

Aaron Sidford  
MS&E  
Stanford University  
Stanford, USA  
sidford@stanford.edu

Abstract—We present an algorithm, which given any $m$-edge $n$-vertex directed graph with positive integer capacities at most $U$ computes a maximum $s$-$t$ flow for any vertices $s$ and $t$ in $O(m^{4/3} + O(1)\sqrt[4]{m})$ time. This improves upon the previous best running times of $O(m^{11/8} + O(1)\sqrt[4]{m})$ [1], $O(m\sqrt{n}\log U)$ [2] and $O(mn)$ [3] when the graph is not too dense and doesn’t have large capacities. We build upon advances for sparse maxflow based on interior point methods [1], [4], [5]. Whereas these methods increase the energy of local $\ell_2$-norm minimizing electrical flows, we instead increase the Bregman divergence value of flows which minimize the Bregman divergence with respect to a weighted log barrier. This allows us to trace the central path with progress depending only on $\ell_\infty$ norm bounds on the congestion vector as opposed to the $\ell_2$ norm, which arises in these prior works. Further, we show that smoothed $\ell_2$-$\ell_p$ flows [6], [7] which were used to maximize energy [1] can also be used to efficiently maximize divergence, thereby yielding our desired runtimes. We believe our approach towards Bregman divergences of barriers may be of further interest.

Keywords—Max flow, interior point method, bipartite matching, optimization

This is an extended abstract corresponding to a merge of the full paper arxiv.org/abs/2003.08929 [8] and the full paper arxiv.org/abs/2009.03260 [9].

I. INTRODUCTION

The $s$-$t$ maximum flow problem and its dual, the $s$-$t$ minimum cut on graphs are among the most fundamental problems in combinatorial optimization and have a wide range of applications. They serve as a testbed for new algorithmic concepts which have found uses in other areas of theoretical computer science and optimization. In the well-known $s$-$t$ maximum flow problem we are given a graph $G = (V, E)$ with $m$ edges and $n$ vertices with positive integer edge capacities $1 \leq u_e \leq U$, and aim to route as much flow as possible from $s$ to $t$ while restricting the magnitude of the flow on each edge to its capacity.

Decades of work in combinatorial algorithms for this problem led to a large set of results culminating in the work of Goldberg-Rao [10] which achieves a $O(m \min\{m^{3/2}, n^{2/3}\} \log U)$ time algorithm for directed maxflow. This bound remained unimproved for many years,

1Theorem 1 is a result of works resulting in a $O(mn)$ time algorithm for strongly polynomial time maxflow [3] – we will not address these here.

2There were several earlier combinatorial algorithms giving improvements in the case of bounded flow values and $\epsilon$-approximate maxflow on undirected graphs [14]–[17], though none achieved a better than $n^{3/2}$ runtime on all sparse graphs.

3Several algorithms discussed here, including the original algorithm of [13], are randomized. We will not distinguish randomized versus deterministic algorithms in this introduction.

2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)
insights lead to [4] which was improved by [5] to obtain an \( \tilde{O}(m^{10/7}U^{1/7}) \) time maxflow algorithm.

Even more recently, Liu and Sidford [1] improved the runtime of M톀dry [5] and showed that instead of carefully tuning the weights based on the electrical energy, one can consider the separate problem of finding a new set of weights under a budget constraint to maximize the energy. They showed that a version of this problem reduces to solving \( \ell_2-\ell_p \) norm flow problems and hence can be solved in almost-linear time using the work of [6], [7]. This leads to a \( O(m^{11/5+o(1)}U^{1/5}) \)-time algorithm for maxflow. However, this result still relied on the amount of progress one can take in each iteration being limited to the bounds one can ensure on the \( \ell_4 \) norm of the congestion of an electrical flow vector, as opposed to the ideal \( \ell_\infty \) norm of the congestion vector which governed [13].

We remark here that there are IPMs for linear programming which only measure centrality in \( \ell_\infty \) norm as opposed to the \( \ell_2 \) or \( \ell_4 \) norm. In particular [23], [24] show how to take a step with respect to a softmax function of the duality gap and trace the central path only maintaining \( \ell_\infty \) norm bounds. However, it is unclear how to leverage these for faster algorithms for maxflow and this paper takes a different approach based on measuring progress based on the \( \ell_\infty \) norm of the congestion vector.

A. Our Contributions

Building on the past decade of advances to maxflow, this paper obtains an \( O(n^{4/3+o(1)}) \) maxflow algorithm on sparse, unit-capacity graphs, thereby closing the longstanding gap between the runtime achieved for approximately solving maxflow using Laplacian system solvers in [13] and the best runtime known for solving maxflow on directed graphs. We also shed light on the energy maximization framework that underlie all previous \( O(n^{3/2-o(1)}) \) results for exact directed maxflow and depart from it to achieve our bounds. Energy maximization, though straightforward to analyze, is somewhat mysterious from an optimization perspective - it is unclear what (if any) standard optimization technique it corresponds to. In this paper, we show that energy maximization arises naturally when locally optimizing the second order Taylor approximation of a Bregman divergence with respect to a logarithmic barrier. We then show that by optimizing the entire function, instead of its second order Taylor approximation, we can obtain improved convergence. We believe this divergence maximization technique is of intrinsic interest.

Finally, to achieve our result, we show that divergence maximization can be performed efficiently for graphs. Whereas [1] showed that energy maximization could be performed efficiently by solving smoothed \( \ell_2-\ell_p \) flows of [6], [7], here we need to solve problems not immediately representable this way. Nevertheless, we show previous solvers can be applied to solve a quadratic extension of the divergence maximization problem in Lemma V.8, which suffices for our algorithms. More generally, we open up the algorithms which underlie \( \ell_2-\ell_p \) flow solvers and show in Theorem 2 that a range of undirected flow optimization problems, including divergence maximization, can be solved efficiently. We hope this generalized flow solving routine may find further use as well.

B. Our Results

The main result of this paper is the following theorem.

**Theorem 1 (Maximum Flow).** There is an algorithm which solves maxflow in \( m \)-edge, integer capacitated graphs with maximum capacity \( U \) in \( m^{3/3+o(1)}U^{1/3} \) time.

Theorem 1 yields an exact maxflow runtime matching the \( \tilde{O}(mn^{1/3}e^{-11/3}) \) runtime of [13] for \( (1-e) \)-approximate undirected maxflow on sparse graphs. Further, this improves on the recent \( m^{11/8+o(1)}U^{1/4} \) time algorithm of [1] when \( U \leq m^{1/2-\varepsilon} \) for some \( \varepsilon > 0 \). When \( U \geq m^{1/2} \), the algorithms of Theorem 1 and [1], [4], [5] all have runtime \( O(m^{3/2}) \), which was known by [10]. Hence, we assume \( U \leq \sqrt{m} \) throughout the paper. An immediate corollary of Theorem 1 is the following result on bipartite matching.

**Corollary I.1 (Bipartite Matching).** There is an algorithm which given a bipartite graph with \( m \) edges computes a maximum matching in time \( m^{8/3+o(1)} \).

We note that Theorem 1 and Corollary I.1 are deterministic. This is achieved by leveraging [26], which recently showed that algorithms for many flow algorithms, including Laplacian system solvers [18], smoothed \( \ell_2-\ell_p \) flow computation methods [6], [7] and some maxflow IPMs [4], [5], may be derandomized with a \( n^{o(1)} \) runtime increase.

C. Previous work

**Approximate undirected maxflow:** An extensive line of work exploiting continuous optimization techniques for faster maxflow algorithms was inspired by [13]. Lee et al. [27] also presented a \( O(n^{1/3+\poly(1/\varepsilon)}) \) iteration algorithm for unit-capacity graphs also using electrical flows. Further, Kelner et al. [28] and Sherman [29] presented algorithms achieving \( O(m^{1+o(1)}\poly(1/\varepsilon)) \) runtimes for maxflow and its variants, [30] improved this to nearly linear time for maximum flow, and [19], [31] further improved the \( \varepsilon \) dependence.
IPMs for maxflow: In order to obtain highly accurate solutions and improved runtimes for directed maxflow, recent work has leveraged IPMs for linear programming [20], [32]. As discussed, classic IPMs [21] and nearly linear time Laplacian systems solvers [18] directly yield an $O(m^{3/2} \log(U))$-time maxflow algorithm. In the case of bounded $U$, this was improved by the sequence of works [1], [4], [5]. Beyond these results, Lee and Sidford [2] also devised a faster IPM using weighted barriers to achieve an $O(m \sqrt{m} \log(1/\epsilon))$-time maxflow. Further, [33] achieved a runtime for minimum cost flow matching the runtimes of [4], [5] and this was recently improved by [22], leveraging the techniques of this paper. Very recently, [34] also provided an IPM-based algorithm which yields an $O(m + n^{3/2})$ time algorithm for bipartite matching and similar runtimes for related problems.

$\ell_p$-flows and beyond: Closely related to this paper, and the preceding lines of research, is recent work on obtaining high-precision solvers for $\ell_p$ flow problems on graphs, i.e. sending a specified amount of flow while minimizing a weighted $\ell_p$-norm. These problems interpolate between electrical flow problems $p = 2$, undirected maximum flow problems $p = \infty$, and transshipment problems $p = 1$. Bubeck et al. [35] note that although standard self-concordance theory for IPMs cannot improve upon $\sqrt{m}$ iterations for $p \neq 2$, an alternative homotopy-based method can solve the problem in $O_p(m^{3/2 - \frac{5}{64}} \log(1/\epsilon))$ iterations, where $O_p$ hides dependencies on $p$ in the runtime. This lead to improvements on the runtime for constant values of $p$. Next, Adil et al. [36] provided an $\ell_p$ flow algorithm which reduced the problem to approximately solving a sequence of smoothed $\ell_2$-$\ell_p$ flows, i.e. objectives with both $\ell_2$ resistances and $\ell_p$ weights, and applied a variant of [13] to solve it. This lead to a $O_p(m^{1 + \frac{1}{p - 2}} \log^2(1/\epsilon))$-time algorithm, which is $O_p(m^{3/3} \log^2(1/\epsilon))$ even for large $p$. Further, Kyng et al. [6] leveraged the tools from Spielman and Teng [18] for $\ell_2$-norm flow problems to show that smoothed $\ell_2$-$\ell_p$ flows could be solved for unit $\ell_p$-norm weights and $p \in [\omega(1), o(\log n)]$ in $m^{1 + o(1)}$ time. Although, these results do not immediately lead to faster directed maxflow algorithm, as naively this would require $p = poly(m)$, both the maxflow algorithm of this paper and [1] leverage the almost linear time $\ell_2$-$\ell_p$-solver, for $p = O(\sqrt{\log m})$ of [6], [7].

II. Preliminaries

General notation: We let $\mathbb{R}^{m,t}_{2\alpha}$ denote the set of $m$-dimensional real vectors which are entrywise at least $\alpha$. For $v \in \mathbb{R}^m$ and real $p \geq 1$ we define $\|v\|_p$, the $\ell_p$-norm of $v$, as $\|v\|_p \overset{def}{=} \left(\sum_{i=1}^{m} |v_i|^p \right)^{1/p}$, and $\|v\|_\infty \overset{def}{=} \max_{i=1}^{m} |v_i|$. For $n \times n$ positive semidefinite matrices $M_1, M_2$ we write $M_1 \succeq M_2$ for $r \geq 1$ if $r^{-1} x^T M_1 x \leq x^T M_2 x \leq r x^T M_1 x$ for all $x \in \mathbb{R}^n$. For a differentiable function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ we define its induced Bregman divergence as $D_f(x\parallel y) \overset{def}{=} f(x) - f(y) - \nabla f(y)^T (y - x)$ for all $x, y \in \mathbb{R}^n$.

Graphs: Throughout this paper, in the graph problems we consider, we suppose that there are both upper and lower capacity bounds on all edges. We let $G$ be a graph with vertex set $V$, edge set $E$, and upper and lower capacities $u^+ \geq 0$ and $u^- \geq 0$ respectively on edge $e$. We use $U$ to denote the maximum capacity of any edge, so that $\max\{u^+_e, u^-_e\} \leq U$ for all edges $e$. We let $n$ denote the number of vertices $|V|$, and let $m$ denote the number of edges $|E|$. Further we view undirected graphs as directed graphs with $u^+_e = u^-_e$ by arbitrarily orienting its edges.

The Maximum Flow Problem: Given a graph $G = (V, E)$ we call any assignment of real values to the edges of $E$, i.e. $f \in \mathbb{R}^E$, a flow. For a flow $f \in \mathbb{R}^E$, we view $f_e$ as the amount of flow on edge $e$. If $f_e > 0$ we interpret this as sending $f_e$ units in the direction of the edge orientation and if $f_e < 0$ we interpret this as sending $|f_e|$ units in the direction opposite the edge orientation.

In this paper we consider $ab$-flows, where $a \in V$ is called the source, and $b \in V$ is called the sink. An $ab$-flow is a flow which routes $t$ units of flow from $a$ to $b$ for some real number $t \geq 0$. Define the unit demand vector $\chi_{ab} = 1_b - 1_a$, a vector with a 1 in position $a$ and −1 in position $b$. When $a$ and $b$ are implicit, we write $\chi = \chi_{ab}$. In this way, we also refer to an $ab$-flow which routes $t$ units from $a$ to $b$ as a $t$-flow. The incidence matrix for a graph $G$ is $m \times n$ matrix $B$, where the row corresponding to edge $e = (u, v)$ has a 1 (respectively −1) in the column corresponding to $v$ (respectively $u$). Note that $f \in \mathbb{R}^E$ is a $t$-flow if and only if $B^T f = t \chi$. More broadly, we call any $d \in \mathbb{R}^V$ a demand vector if $d \perp 1$ and we say $f \in \mathbb{R}^E$ routes $d$ if $B^T f = d$.

We say that a $t \chi$-flow $f$ is feasible in $G$ if $-u^- \leq f_e \leq u^+$ for all $e \in E$, so that $f$ satisfies the capacity constraints. We define the maximum flow problem as the problem of given a graph $G$ with upper and lower capacities $u^+$ and $u^-$, and source and sink vertices $a$ and $b$, to compute a maximum feasible $ab$-flow. We denote the maximum value as $t^*$. For a real number $t \leq t^*$, we let $F_t \overset{def}{=} t^* - t$ denote the remaining amount of flow to be routed.

III. Algorithm Derivation and Motivation

In this section we present a principled approach for deriving our new method and the previous energy-based methods [1], [4], [5] for maxflow from an IPM setup.

A. Interior Point Method Setup

The starting point for our method is the broad IPM framework for maxflow of [1], which in turn was broadly inspired by [5]. We consider the setup described in Section II and design algorithms that maintain a flow $f \in \mathbb{R}^E$, a parameter $t \geq 0$, and weights $w^+, w^- \in \mathbb{R}^n_{>1}$ such that $B^T f = t \chi$ and $f$ is a high accuracy approximation of the solution $F^*_{\epsilon, \omega}$ of the following optimization problem:

$$F_{\epsilon, \omega}^* \overset{def}{=} \arg \min_{f} V(f) \quad (1)$$
where
\[ V(f) \overset{\text{def}}{=} - \sum_{e \in E} \left( w^+_e \log(u^+_e - f_e) + w^-_e \log(u^-_e + f_e) \right) \]

Here, \( V(f) \) is known as the weighted logarithmic barrier and penalizes how close \( f \) is to violating the capacity constraints and \( t \) is the amount of flow sent from \( a \) to \( b \).

Generally, IPMs proceed towards optimal solutions by iteratively improving the quality, i.e. increasing the parameter \( t \), and decreasing the proximity to the constraints, i.e. decreasing \( V(f) \). Previous maxflow IPMs \cite{PPS, SS, GT, JMT} all follow this template. Specifically, \cite{PPS, JMT} alternate between Newton step to improve the optimality of \( f \) for Eq. (1) (called centering steps) and computing a new flow and weights to approximately solve Eq. (1) for a larger value of \( t \) (called progress steps). Applying such an approach, while using Laplacian system solvers to implement the steps in nearly linear time, coupled with a preliminary graph preprocessing step known as preconditioning (Section IV-A) directly yields to an \( \tilde{O}(m^{3/2}) \) time algorithm. Recent advances \cite{PPS, SS, GT, JMT} were achieved by performing further work to modify the weights and flows used.

**B. Progress steps via divergence minimization**

To understand (and improve upon) previous maxflow IPMs, here we explain how to view progress steps in this framework as computing a divergence minimizing \( \delta \chi \)-flow. Note that, without weight changes, the cumulative result of a progress and centering step is essentially moving from \( f^*_{t,w} \) to \( f^*_{t,\delta,w} \) for a step size \( \delta \). The optimality conditions of Eq. (1) give that the gradient of \( V \) at the optimum \( f^*_{t,w} \) of Eq. (1) is perpendicular to the kernel of \( B^T \), so there is a vector \( y \) with \( B y = \nabla V(f^*_{t,w}) \). Define
\[
\hat{f} \overset{\text{def}}{=} \arg \min_{B^T f = \delta x} D_V(f^*_{t,w} + f || f^*_{t,w}) = \arg \min_{B^T f = \delta x} V(f^*_{t,w} + f) - V(f^*_{t,w}) - \nabla V(f^*_{t,w})^T f,
\]
i.e. the \( \delta \chi \)-flow with smallest divergence from \( f^*_{t,w} \) against the barrier \( V \). Again, optimality conditions give that there is a vector \( z \) with \( B z = \nabla D_V(f^*_{t,w} + \hat{f} || f^*_{t,w}) = \nabla V(f^*_{t,w} + \hat{f}) - \nabla V(f^*_{t,w}) \). Therefore, \( B(y + z) = \nabla V(f^*_{t,w} + \hat{f}) \). Since \( f^*_{t,w} + \hat{f} \) is a \((t+\delta)\)-\( \chi \)-flow, we must have \( f^*_{t,w} + \hat{f} = f^*_{t,\delta,w} \), by optimality conditions, so that adding \( \hat{f} \) to an optimal point \( f^*_{t,w} \) lands us at the next point \( f^*_{t,\delta,w} \).

Now, a standard progress step in this framework may be computed by taking a Newton step, i.e. minimizing the second order Taylor approximation of the divergence. The second order Taylor expansion of \( D_V(f^*_{t,w} + f || f^*_{t,w}) \) is
\[
\frac{1}{2} f^T \nabla^2 V(f^*_{t,w}) f
\]
and the resulting step is
\[
\arg \min_{B^T f = \delta x} \frac{1}{2} f^T \nabla^2 V(f^*_{t,w}) f = \delta \nabla^2 V(f^*_{t,w})^{-1} B (B^T \nabla^2 V(f^*_{t,w})^{-1} B)^T \chi.
\]

This can be computed in \( O(m) \) time plus the time to solve a Laplacian system, i.e. \( \tilde{O}(m) \) \cite{V87}. Choosing \( \delta \) that routes \( \Omega(m^{-1/2}) \) fraction of the remaining flow, adding the flow in Eq. (4) to our current point, and taking further Newton steps to re-center yields an \( \tilde{O}(m^{3/2}) \) time maxflow algorithm.

**C. Energy-based improvements**

Improvements to the above \( \tilde{O}(m^{3/2}) \) time algorithm \cite{PPS, GT, JMT} arise by a more careful analysis of the step size \( \delta \) of the Newton step that we may take such that recentering may still be performed in \( \tilde{O}(m) \) time, and by leveraging that the flow in Eq. (4) is an electric flow. Precisely, the size of the step we may take is governed by the congestion of the flow we wish to add, which is defined edge-wise as the ratio of flow on an edge to its residual capacity (see \( c^+_e, c^-_e \) in Section IV-A). In this way, the \( \ell_\infty \) norm of congestion of the \( \chi \)-electric flow governs the amount of flow we may add before violating capacity constraints. On the other hand, because the \( \chi \)-electric flow was a minimizer to a second order approximation of the divergence, the \( \ell_4 \) norm of congestion of the \( \chi \)-electric flow governs the amount of flow we may add so that centering can still be performed in \( \tilde{O}(m) \) time, whereas a bound on the \( \ell_2 \) norm of congestion suffices to achieve the \( \tilde{O}(m^{3/2}) \) time algorithm.

In this way, it is natural to attempt to compute weight changes that reduce the \( \ell_4 \) norm of congestion induced by the \( \chi \)-electric flow. Madry \cite{GT, JMT} achieves this by boosting edges with high congestion in the \( \chi \)-electric flow and trading off against a potential function that is the energy of the \( \chi \)-electric flow with resistances induced by the Hessian of the weighted logarithmic barrier at the current point.

To improve on the algorithm of \cite{JMT, PPS} instead views increasing energy via budgeted weight change as its own optimization problem. Precisely, the optimization problem was to maximize the energy of an electric flow in a graph \( G \) that originally had resistances \( r \) under a resistance increase budget. Written algebraically, for a weight budget \( W \):
\[
\max_{\|r\|_{L^1} \leq W} \min_{B^T f = \delta x} \sum_{e \in E} (r_e + r^*_e) f^2_e.
\]
\cite{PPS} showed a slightly different version of this objective was solvable in \( m^{1+o(1)} \) time using smoothed \( \ell_2-\ell_p \) flows \cite{GT68}, and that the combinatorial edge boosting framework of \cite{GT, JMT} can essentially be viewed as greedily taking gradient steps against the objective in Eq. (5).

**D. Our new method: beyond electric energy**

A disappointing feature of the above discussion is that while the \( \ell_\infty \) norm of congestion governs the amount of flow we may add and still have a feasible flow, we are forced to control the \( \ell_4 \) norm of congestion of the electric flow to allow for efficient centering. This is due to the fact that although the step can be taken without violating capacity constraints, there is sufficient loss in local optimality
that $\tilde{O}(1)$ centering Newton steps cannot recenter it. This leads to the heart of our improvement – we resolve this discrepancy between the $\ell_\infty$ and $\ell_4$ norm of congestion by directly augmenting via the divergence minimizing flow of Eq. (2). As a result, it suffices to compute weight changes to minimize the $\ell_\infty$ norm of congestion of this flow.

The next challenge is to compute this divergence minimizing flow, and compute weight changes to reduce the $\ell_\infty$ norm of its congestion. To approach this, we consider the problem of moving from $f^t_{t,w}$ to $f^t_{t+\delta,w}$ for a step size $\delta$, assuming that the residual capacities induced by $f^t_{t,w}$ and $f^t_{t+\delta,w}$ are within 1.05 multiplicatively. This implies that $\nabla^2 V(f^t_{t,w}) \approx 1.2 \nabla^2 V(f^t_{t+\delta,w})$. To solve this problem, for each piece of the $V(f)$ objective, i.e. $(w^+ \log(u^+ - f_e) + w^- \log(u^- + f_e))$, we replace it with a quadratic extension, a function that agrees with it on some interval, and extends quadratically outside. Our new objective will have a stable Hessian everywhere, hence can be minimized by Newton’s method. By construction, the optimum of the quadratically extended problem and original are the same using convexity (see Observation 1). Further details are provided in Section IV-B.

Finally, we must compute weights that reduce the $\ell_\infty$ norm of congestion of the divergence minimizing flow. As the approach of [1] computes weight changes to maximize the electric energy, we instead compute weight changes to maximize the divergence of the divergence minimizing flow. Doing this requires extending the analysis of [5] and energy maximization framework of [1] to nonlinear objectives, such as the quadratic extensions described above, and then generalizing the iterative refinement framework introduced by [6], [36] to a large family of new objectives. We hope that both this unified view of energy and divergence maximization as well as the methods we give for performing this optimization efficiently may have further utility.

IV. TECHNICAL INGREDIENTS

In this section, we elaborate on several technical aspects discussed in Section III. We give details for setting up the IPM in Section IV-A, discuss preconditioning in Section IV-A, elaborate on quadratic extensions in Section IV-B, and discuss iterative refinement in Section IV-C.

A. IPM Details and Preconditioning

In this section, we give a detailed description of our IPM setup. One can reduce directed maxflow to undirected maxflow with linear time overhead (see [4], [37] or [1] Section B.4) and consequently, we assume our graph is undirected, so that $u^+ = u^-$. Assuming that there is a feasible $t\chi$-flow, optimality conditions of Eq. (1) give that the gradient of $V$ at the optimum $f^t_{t,w}$ of Eq. (1) is perpendicular to the kernel of $B^T$, i.e. there is a dual vector $y \in \mathbb{R}^V$ such that $By = \nabla V(f^t_{t,w})$. Consequently, for parameter $t$ and weight vectors $w^+, w^-$ we say that a flow $f$ is on the weighted central path if and only if there exists a dual vector $y \in \mathbb{R}^V$ such that

$$B^T f = t\chi$$

and

$$[By]_e = [\nabla V(f)_e] = \frac{w^+}{u^+ - f_e} - \frac{w^-}{u^- + f_e} \text{ for all } e \in E$$

For simplicity, we write $w = (w^+, w^-) \in \mathbb{R}^{2|E|}$, where we define $\mathbb{R}^{2|E|} \equiv \mathbb{R}^E \times \mathbb{R}^E$. We define residual capacities $c^+_{e} = u^+ - f_e, c^-_{e} = u^- + f_e$ and $c_e = \min(c^+_{e}, c^-_{e})$. Note $c_e \geq 0$ for all $e \in E$ if and only if $f$ is feasible.

We initialize $w^+ = w^- = 1$, $t = 0$, and $f = 0$, which is central. Previous IPM based algorithms for maxflow [1], [4], [5] alternated between progress steps and centering steps. Progress steps increase the path parameter $t$ at the cost of centrality, which refers to the distance of $f$ from satisfying Eq. (7) in the inverse norm of the Hessian of $V(f)$ – see [1] Definition 4.1. Centering steps improve the centrality of the current point without increasing the path parameter $t$. Our algorithm more directly advances along the central path – given parameter $t$, weights $w$, and central path point $f^t_{t,w}$ we compute new weights $w^{new}$, advance the path parameter to $t + \delta$, and compute $f^{t+\delta,w^{new}}_{t+\delta,w^{new}}$.

The goal of the IPM is to reduce the value of the residual flow $f_t = t^* - t$ below a threshold, at which point we may round and use augmenting paths [27]. We assume that the algorithm knows $f_t$ throughout, as our algorithm succeeds with any underestimate of the optimal flow value $t^*$, and we can binary search for the optimal flow value.

Preconditioning. To precondition our undirected graph $G$, we add $m$ undirected edges of capacity $2U$ between source $a$ and sink $b$. This increases the maximum flow value by $2mU$. Throughout the remainder of the paper, we say that the graph $G$ is preconditioned if it is undirected and we have added these edges. Intuitively, preconditioning guarantees that a constant fraction of the remaining flow in the residual graph may be routed in its undirectification, i.e. $G$ with capacities $c_e$. The following lemma was shown in [1] Section B.5.

**Lemma IV.1.** Consider a preconditioned graph $G$. For parameter $t$ and weights $w$ let $c_e$ be the residual capacities induced by the flow $f^t_{t,w}$. Then for every preconditioning edge $e$ we have that $c_e \geq \frac{F_t}{7\|w\|_1}$. If $\|w\|_1 \leq 3m$ then $c_e \geq \frac{F_t}{21m}$.

At the start of the algorithm, as we initialized $w^+ = w^- = 1$, we have $\|w\|_1 = 2m$. To apply Lemma IV.1 we maintain the stronger invariant that $\|w\|_1 \leq 5m/2$ before each step, but may temporarily increase to $\|w\|_1 \leq 3m$ during the step.
B. Advancing along the central path via quadratic smoothing

Let $t$ be a path parameter, and let $\delta$ be a step size. Let $c_{e}^{+}, c_{e}^{-}$ be the residual capacities induced by $f_{\ell_{+},w}^{*}$, and let $(c_{e}^{+})', (c_{e}^{-})'$ be those induced by $f_{\ell_{+},w}^{*}$. We sketch an algorithm that computes $f_{\ell_{+},w}^{*}$ to high accuracy from $f_{\ell_{+},w}^{*}$ in $O(m)$ time given that for all $e \in E$ that $c_{e}^{+} \approx_{0.05} (c_{e}^{+})'$ and $c_{e}^{-} \approx_{0.05} (c_{e}^{-})'$. Let $f = f_{\ell_{+},w}^{*} - f_{\ell_{+},w}^{*}$ and define the change in the value of the barrier $V$ when we add $f$ as

$$B(f) \overset{\text{def}}{=} V(f + f_{\ell_{+},w}^{*}) - V(f_{\ell_{+},w}^{*})$$

$$= -\sum_{e \in E} w_{e}^{+} \log \left( 1 - \frac{f_{e}}{u_{e} - (f_{\ell_{+},w})_{e}} \right) + w_{e}^{-} \log \left( 1 + \frac{f_{e}}{u_{e} + (f_{\ell_{+},w})_{e}} \right),$$

so that $\hat{f} = \arg \min_{B(f) = \delta \chi} B(f)$. To compute $\hat{f}$, we smooth Eq. (9) by replacing each instance of $\log(\cdot)$ with a function $\log(\cdot)$ defined as

$$\tilde{\log}_{e}(1+x) \overset{\text{def}}{=} \begin{cases} \log(1+x) & \text{for } |x| \leq \varepsilon \\ \log(1+\varepsilon) + \frac{(x-\varepsilon)}{(1+\varepsilon)} & \text{for } \varepsilon \geq x \geq -\varepsilon \\ \log(1-\varepsilon) - \frac{(x+\varepsilon)}{(1-\varepsilon)} & \text{for } x \leq -\varepsilon. \end{cases}$$

Here, we fix $\varepsilon = 1/10$ and write $\tilde{\log}(1+x) \overset{\text{def}}{=} \tilde{\log}_{1/10}(1+x)$. Note that $\log_{e}(1+x)$ is the natural quadratic extension of $\log(1+x)$ outside the interval $|x| \leq \varepsilon$. Specifically, the functions agree for $|x| \leq \varepsilon$, and we $\log_{e}(1+x)$ is the second order Taylor expansion of $\log(1+x)$ at $\varepsilon$, $-\varepsilon$ for $x > \varepsilon, x < -\varepsilon$ respectively. In this way, $\tilde{\log}(1+x)$ is twice differentiable everywhere.

$$\tilde{B}(f) \overset{\text{def}}{=} -\sum_{e \in E} \left( w_{e}^{+} \tilde{\log} \left( 1 - \frac{f_{e}}{u_{e} - (f_{\ell_{+},w})_{e}} \right) + w_{e}^{-} \log \left( 1 + \frac{f_{e}}{u_{e} + (f_{\ell_{+},w})_{e}} \right) \right).$$

We now claim that $\hat{f} = \arg \min_{B(f) = \delta \chi} \tilde{B}(f)$, and that it can be computed in $O(m)$ time. To argue the latter, note that by construction, all Hessians $V^{2}B(f)$ are within a multiplicative factor of 2 of each other, hence we can compute $\arg \min_{B(f) = \delta \chi} \tilde{B}(f)$ in $O(m)$ time using Newton’s method and electric flow computations. Because $c_{e}^{+} \approx_{0.05} (c_{e}^{+})'$ and $c_{e}^{-} \approx_{0.05} (c_{e}^{-})'$, we know that $B$ and $\tilde{B}$ agree in a neighborhood of $f$, so $\hat{f} = \arg \min_{B(f) = \delta \chi} \tilde{B}(f)$ by the following simple observation.

Observation 1. [35] Let $\chi \subseteq \mathbb{R}^{n}$ be a convex set, and let $f, g : \mathbb{R}^{n} \rightarrow \mathbb{R}$ be convex functions. Let $x^{*} = \arg \min_{x \in \chi} f(x)$, and assume that $f, g$ agree on a neighborhood of $x^{*}$ in $\mathbb{R}^{n}$. Then $g(x^{*}) = \min_{x \in \chi} g(x)$.

We emphasize that we do not directly do the procedure described here, and instead apply quadratic smoothing in a different form in Section V. There we smooth the function $D \overset{\text{def}}{=} D_{-\log(1-x)}(x|0)$, the Bregman divergence of $x$ to 0 with respect to the function $-\log(1-x)$, instead of directly smoothing $\log(1+x)$. The smoothed function $D$ is shown in Eqs. (11) and (12).

C. Iterative refinement

The idea of iterative refinement was introduced in [6], [36] to solve $p$-norm regression problems, such as $\min_{B(f) = d} \|f\|_{p}$. Iterative refinement solves such an objective by reducing it to approximately minimizing objectives which are combinations of quadratic and $\ell_{p}$ norm pieces. Specifically, we can show that for a fixed flow $f$ there is a gradient vector $g$ such that for any circulation $\Delta$ we have

$$\|f + \Delta\|_{p}^{\varepsilon} - \|f\|_{p}^{\varepsilon} = g^{T}\Delta + \Theta_{p} \left( \sum_{e \in E} |f_{e}|^{p-2} \Delta_{e}^{2} + \|\Delta\|_{p}^{\varepsilon} \right),$$

so that approximately minimizing Eq. (10) over circulations $\Delta$ suffices to make multiplicative progress towards the optimum of $\min_{B(f) = d} \|f\|_{p}$. We sketch in Section V-A how to reduce general class of objectives to approximately minimizing objectives which are combinations of quadratic and $\ell_{p}$ norm pieces. Specifically, any convex function $h$ with stable second derivative admits an expansion for $h(x + \Delta)^{p} - h(x)^{p}$ similar to Eq. (10). We then combine this expansion with the almost linear time smoothed $\ell_{2}\ell_{p}$ solver of [6] to solve the necessary objectives.

V. EFFICIENT DIVERGENCE MAXIMIZATION

In this section we present our interior point framework and show Theorem 1. Throughout the section we set $\eta = \log_{m} (m^{1/6-\varepsilon}U_{1/3})$, in pursuit of a $m^{3/2-\eta+o(1)}$ time algorithm. We assume $G$ is preconditioned Section IV-A and we maintain the invariant $\|w\|_{1} \leq 5m/2$ before each step, and $\|w\|_{1} \leq 3m$ at all times. As our algorithm runs in time at least $m^{3/2}$ for $U > \sqrt{m}$ we assume $U \leq \sqrt{m}$ throughout this section. We assume that our algorithm knows $F_{t}$, as discussed in Section IV-A, and that $F_{t} \geq m^{1/2-\eta}$ or else we can round to an integral flow and run augmenting paths.

Notational conventions: We largely adopt the same notation as used in Section IV-B. We use $D$ to refer to functions of the logarithmic barrier, and for a function $h$, we use $\tilde{h}$ to denote a quadratic extension of $h$. For flows we use $f$ and $\tilde{f}$, the latter which refers to flows we wish to augment by. We use $\Delta$ and $\tilde{\Delta}$ for circulations. The letters $w, \mu, v$ refer to weights and weight changes, and $W$ refers to a weight budget.
For $\varepsilon < 1$ define the functions
\begin{equation}
D(x) \overset{\text{def}}{=} -\log(1 - x) - x \quad \text{and} \quad \tilde{D}_\varepsilon(x) \overset{\text{def}}{=} \begin{cases} D(x), & \text{for } |x| \leq \varepsilon \\ D(\varepsilon) + D'(\varepsilon)(x - \varepsilon) + \frac{D''(\varepsilon)}{2}(x - \varepsilon)^2, & \text{for } x \geq \varepsilon \\ D(-\varepsilon) + D'(-\varepsilon)(x + \varepsilon) + \frac{D''(-\varepsilon)}{2}(x + \varepsilon)^2, & \text{for } x \leq -\varepsilon. \end{cases}
\end{equation}

Throughout, we will omit $\varepsilon$ and write $\tilde{D}(x) \overset{\text{def}}{=} \tilde{D}_{1/10}(x)$. As discussed in Section IV-B, $\tilde{D}(x)$ is such that it behaves as a quadratic around $x = 0$, and has continuous second derivatives. We have defined $D$ as the Bregman divergence of $-\log(1-x)$ from 0, and $\tilde{D}$ is the quadratic extension of $D$ as described in Section IV-B. Several useful properties of the derivatives and stability of $D$ and $\tilde{D}$ are collected in Lemma V.1, which we prove in the full version.

**Lemma V.1 (Properties of $\tilde{D}$).** We have that $1/2 \leq D''(x) \leq 2$. Also, for $x \geq 0$ we have that $x/2 \leq \tilde{D}'(x) \leq 2x$ and $-x/2 \geq \tilde{D}'(-x) \geq -2x$. We have that $x^2/4 \leq \tilde{D}(x) \leq x^2$ for all $x$.

Now, we define the higher order analog of electric energy which we maximize under a weight budget. Below, we assume without loss of generality that $c^+ e \leq c^- e$ for all edges $e$, as the orientation of each edge is arbitrary in the algorithm. In this way, $c_e = c^+_e$ for all $e$.

\begin{align*}
D_V^w(f) & \overset{\text{def}}{=} \sum_{e \in E} \left( w^+_e D\left( \frac{f_e}{c^+_e} \right) + w^-_e D\left( \frac{f_e}{c^-_e} \right) \right) \\
\tilde{D}_V^w(f) & \overset{\text{def}}{=} \sum_{e \in E} \left( w^+_e \tilde{D}\left( \frac{f_e}{c^+_e} \right) + w^-_e \tilde{D}\left( \frac{f_e}{c^-_e} \right) \right)
\end{align*}

Define $\text{val}$ and $\text{val}$ as follows, where $p = 2[\sqrt{\log m}]$, and $W = m^{100}$ is a constant. For clarity, we express the vector inside the $\| \cdot \|_p$ piece of Eqs. (13) and (14) coordinate-wise, where the coordinate corresponding to edge $e$ is written.

\begin{align*}
\text{val}(f) & \overset{\text{def}}{=} D_V^w(f) + W\left( (c^+_e)^2 \left( D\left( \frac{f_e}{c^+_e} \right) + \left( \frac{c^-_e}{c^+_e} \right) D\left( \frac{f_e}{c^-_e} \right) \right) \right) \\
\text{val}(f) & \overset{\text{def}}{=} \tilde{D}_V^w(f) + W\left( (c^+_e)^2 \left( \tilde{D}\left( \frac{f_e}{c^+_e} \right) + \left( \frac{c^-_e}{c^+_e} \right) \tilde{D}\left( \frac{f_e}{c^-_e} \right) \right) \right)
\end{align*}

These are defined so that for $q$ as the dual norm of $p$, i.e. $1/q + 1/p = 1$, we have that $\text{val}$ and $\text{val}$ correspond to maximizing the minimum values of $D_V^w(f)$ and $\tilde{D}_V^w(f)$ under a weighted $\ell_q$ weight budget. Specifically, we can compute using Sion’s minimax theorem that
\begin{equation}
\max_{\| (c^+_e)^2 \|_q \leq W} \min_{\| (c^-_e)^2 \|_q \leq W} D_V^{w+\nu}(f)
\end{equation}

and similarly for $\tilde{D}_V^w(f)$ and $\tilde{\text{val}}(f)$. The objective requires the constraint $\nu^+_e c^+_e = \nu^-_e c^-_e$ for all $e \in E$ to ensure that the weight increase $\nu$ maintains centrality, and the coefficient of $(c^+_e)^{-2}$ in the weight budget $\| (c^+_e)^{-2} \nu^+_e \|_q \leq W$ is chosen to ensure that the $f_p$-piece in $\text{val}(f)$ is ensured to have approximately unit weights on the $f_e$. Precisely, by Lemma V.1 and $c^+_e \leq c^-_e$ we have

\begin{align*}
(c^+_e)^2 \left( \tilde{D}\left( \frac{f_e}{c^+_e} \right) + \left( \frac{c^-_e}{c^+_e} \right) \tilde{D}\left( \frac{f_e}{c^-_e} \right) \right) = \Theta(f_p^2).
\end{align*}

We require this property in order to apply the smoothed $\ell_2-\ell_p$ flow solvers in Theorem 3.

As $\min_{\| f \|_q \leq d} \text{val}(f)$ is the result of applying Sion’s minimax theorem to a saddle point problem, there will be an optimal solution pair $(f^*, \mu^*)$. Ultimately, $f^*$ will be the flow which we add to our current flow to arrive at the next central path point, and the weight change will be derived from applying a weight reduction procedure to $\mu^*$.

The remaining arguments in this section heavily use local optimality of convex functions. For this reason, we note that $\text{val}(f)$ and $\tilde{\text{val}}(f)$ are convex, proof given in the full version.

**Lemma V.2.** $\text{val}(f)$ and $\tilde{\text{val}}(f)$ are convex.

From now on, we fix a step size $\delta = \frac{f_p}{m^{100}\log m - 1}$. This simplifies our analysis, as our objectives are no longer linear in $\delta$, as is the case with electric flows. We now bound the minimum value of $D_V^w(f)$ over all $\delta_\chi$-flows, and show a congestion bound for the minimizer, where we recall that congestion of a flow is the ratio of flow on an edge to its residual capacity $c_e$. Lemmas V.3 and V.4 generalize
corresponding bounds for electric flows shown in [5] and [1] Lemma 4.5 and 5.2.

**Lemma V.3.** Let $\delta = \frac{F_i}{10^{5m^{1/2-\eta}}}$. Then

$$
\min_{B^T f = \delta \check{f}, \check{f} \in \mathbb{R}^E} \mathcal{D}^V(f) \leq 5 \cdot 10^{-7} m^{2\eta}.
$$

**Proof:** Let $f'$ be the flow which routes $\delta/m$ units of flow on each of the $m$ preconditioning edges. For a preconditioning edge $e_p$, Lemma IV.1 and the invariant $\|w\|_1 \leq 3m$ tells us that

$$
\frac{f'_{e_p}}{c_{e_p}} \leq \frac{\delta/m}{F_i/7\|w\|_1} \leq \frac{21}{10^5 m^{1/2-\eta}}.
$$

Therefore, applying Lemma V.1 to $P$, the set of $m$ preconditioning edges, and again applying that $\|w\|_1 \leq 3m$ gives the desired bound, as

$$
\mathcal{D}^V(f') = \sum_{e \in P} \left( w_e^+ \bar{D} \left( \frac{f_e}{c_e} \right) + w_e^- \bar{D} \left( -\frac{f_e}{c_e} \right) \right)
\leq \left( \frac{f'_{e_p}}{c_{e_p}} \right)^2 \sum_{e \in P} \left( w_e^+ + w_e^- \right)
\leq \frac{21}{10^5 m^{1/2-\eta}} \|w\|_1 \leq 5 \cdot 10^{-7} m^{2\eta}.
$$

**Lemma V.4.** Let $\delta = \frac{F_i}{10^{5m^{1/2-\eta}}}$, and $\hat{f} = \arg \min_{B^T f = \delta \check{f}} \mathcal{D}^V(f)$. Then $|\hat{f}_e| c_e^{-2} \leq m^{2\eta}$.

**Proof:** Local optimality tells us that there is a vector $z \in \mathbb{R}^V$ satisfying $Bz = \nabla \mathcal{D}^V_w(\hat{f})$. This, Lemma V.3, and Lemma V.1, specifically that $x\mathcal{D}^T(x) \leq 2x^2 \leq 8\bar{D}(x)$ gives

$$
\delta \chi^T z = \hat{f}^T B z = \hat{f}^T \nabla \mathcal{D}^V_w(\hat{f})
= \sum_{e \in E} \hat{f}_e \left( \frac{w_e^+}{c_e} \bar{D} \left( \frac{\hat{f}_e}{c_e} \right) - \frac{w_e^-}{c_e} \bar{D} \left( -\frac{\hat{f}_e}{c_e} \right) \right)
\leq 8\mathcal{D}^V_w(\hat{f}).
$$

Note that the flow $\hat{f}$ is acyclic, i.e. there is no cycle where the flow is in the positive direction for every cycle edge, as decreasing the flow along a cycle reduces the objective value. Also, for all edges $e = (u,v)$, we have $z_e - z_u = [Bz]_e = [\nabla \mathcal{D}^V_w(\hat{f})]_e$, which has the same sign as $\hat{f}_e$. As $\hat{f}$ is acyclic, it can be decomposed into alpha-beta paths. Since, some path contains the edge $e$, we get that $|[Bz]_e| = |z_e - z_u| \leq z_b - z_a = \chi^T z$. Using that $x\mathcal{D}^T(x) \geq x^2/2$ from Lemma V.1 we get that

$$
|[Bz]_e| = |[\nabla \mathcal{D}^V_w(\hat{f})]_e| \geq \frac{w_e^+ |\hat{f}_e|}{2(c_e^2)^2} + \frac{w_e^- |\hat{f}_e|}{2(c_e^2)^2} \geq \frac{1}{2} |\hat{f}_e| c_e^{-2}.
$$

Combining these gives

$$
|\hat{f}_e| c_e^{-2} \leq 2 |[Bz]_e| \leq 2 \chi^T z \leq 16\delta^{-1} \mathcal{D}^V_w(\hat{f}) \leq m^{2\eta}
$$

after using Lemma V.3 and $\delta = \frac{F_i}{10^{5m^{1/2-\eta}}} \geq 10^{-5}$.

Now, we show that computing $\hat{f} = \arg \min_{B^T f = \delta \check{f}} \text{val}(f)$ gives us weight changes to control the congestion of the flow we wish to augment by. For clarity, the process is shown in Algorithm 1.

**Algorithm 1** AUGMENT($G, w, F_i, f^*, w^*_{\mu}$). Takes a preconditioned undirected graph $G$ with maximum capacity $U$, weights $w \in \mathbb{R}^{2E}$ with $\|w\|_1 \leq 5m/2$, residual flow $F_i = t^* - t$, central path point $f^*_t,w$. Returns step size $\delta$, weights $\nu$, and $\delta\chi$-flow $\hat{f}$ with $f_{t,w}^{\nu} = f_{t,w}^{\ast} + \hat{f}$.

1: $\eta \leftarrow \log_m \left( \frac{m^{1/6-o(1)}U^{-1/3}}{\delta} \right)$
2: $\delta \leftarrow \frac{10^5 m^{1/2-\eta}}{m^{1/2}}$
3: $c_e^\nu \leftarrow w_e^+ - [f_{t,w}^\ast]_e, c_e^- \leftarrow w_e^- + [f_{t,w}^\ast]_e$.
4: $W \leftarrow m^6\eta$
5: $f \leftarrow \arg \min_{B^T f = \delta \check{f}} \text{val}(f)$
6: $v \leftarrow \mathbb{R}^E$
7: $\nu \leftarrow \mathbb{R}^{2E}$
8: Initialize $\nu \in \mathbb{R}^{2E}_0$
9: for $e \in E$ do
10: if $\frac{\mu^\nu}{\mu^\nu - \mu^\nu_{\ast}} \geq 0$ then
11: $\nu_e^+ \leftarrow \nu_e^+ - \nu_e^\nu (\nu_e^\nu - \nu_e^-) / (\nu_e^+ + \mu_e^- + \mu_e^\nu_{\ast})$, $\nu_e^- \leftarrow 0$
12: else
13: $\nu_e^+ \leftarrow 0, \nu_e^- \leftarrow -\nu_e^- (\nu_e^- + \nu_e^\nu + \mu_e^- + \mu_e^\nu_{\ast}) / (\nu_e^- + \mu_e^- + \mu_e^\nu_{\ast})$
14: end if
15: end for
16: return $(\delta, \hat{f}, \nu)$.

Now, we analyze Algorithm 1. We start by showing that $\hat{f} = \arg \min_{B^T f = \delta \check{f}} \mathcal{D}^V_{w_{\nu}}(f)$ for the weight change $\mu$ in line 7 of Algorithm 1, and that $\nu$ has bounded $\ell_1$ norm. This essentially follows from duality in our setup in Eq. (15).

**Lemma V.5.** Let parameters $\eta, W, c_e^\nu, c_e^-, \delta, f, \hat{f}$, and weight change $\mu$ be defined as in Algorithm 1, and assume that $F_i \geq m^{1/2-\eta}$. Then we have that $\|\mu\|_1 \leq m/2$, $f_{t,w}^{\nu} = f_{t,w}^{\ast} + \mu$, and $\hat{f} = \arg \min_{B^T f = \delta \check{f}} \mathcal{D}^V_{w_{\nu}}(f)$.

**Proof:** Let $v \in \mathbb{R}^{2E}$ be the vector as defined in line 6 in Algorithm 1. By local optimality of $\hat{f}$, we have that there is a vector $z$ satisfying for all $e \in E$ that

$$
[Bz]_e = \left[ \nabla \mathcal{D}^V_{w_{\nu}}(f) \right]_e
= \left( \frac{w_e^+}{c_e} + \frac{v_{p_{\nu}}}{c_e} W(c_e^\nu)^2 \right) D \left( \frac{\hat{f}_e}{c_e} \right)
- \left( \frac{w_e^-}{c_e} + \frac{v_{p_{\nu}}}{c_e} W(c_e^\nu)^2 \right) D \left( -\frac{\hat{f}_e}{c_e} \right).
$$

(16)
For clarity, we rewrite line 7 of Algorithm 1 here as
\[
\mu_\epsilon^+ = W(c_\epsilon^+)^2 \cdot \frac{v_p^{-1}}{\|v_p\|^p}.
\]
and
\[
\mu_\epsilon^c = \frac{c_\epsilon^c}{c_\epsilon^+} \mu_\epsilon^+ = Wc_\epsilon^{-c} \cdot \frac{v_p^{-1}}{\|v_p\|^p}.\tag{17}
\]
Note that \(\mu_\epsilon^+ / c_\epsilon^+ = \mu_\epsilon^c / c_\epsilon^c\), hence \(f_{t,w}^* = f_{t,w+\mu}^*\) by Eq. (7). Combining Eqs. (16) and (17) and optimality conditions of the objective \(\min_{B^T f = \delta_x} D_{W_{w+\mu}}^V(f)\) shows that \(\hat{f} = \arg\min_{B^T f = \delta_x} D_{W_{w+\mu}}^V(f)\). Additionally, if \(\hat{q}\) is the dual of \(p\), i.e. \(1/q + 1/p = 1\), then
\[
\|\mu\|_1 \leq m_0(\eta) \|\mu\|_q \leq 2m_0(\eta)WU^2 \left\| \frac{v_p^{-1}}{\|v_p\|^p} \right\|_q
\]
\[
= 2m_0(\eta)WU^2 = m_0^\epsilon + o(1)U^2 \lesssim m/2
\]
by our choice of \(\eta = \log_{m_0}(m^{1/6 - o(1)}U^{1/3})\).

We now show congestion bounds on \(\hat{f}\) by imitating the proof of Lemma V.3 and applying Lemma V.4. Recall that \(c_\epsilon = \min(c_\epsilon^+, c_\epsilon^-)\).

**Lemma V.6.** Let parameters \(\eta, W, c_\epsilon^+, c_\epsilon^-, \delta, \delta, \hat{f}\), and weight change \(\mu\) be defined as in Algorithm 1, and assume that \(F_1 \geq m^{1/2 - \eta}\). Then we have \(|f_\epsilon| \leq \frac{1}{\sqrt{m}} c_\epsilon\) for all edges \(e\). It follows that \(\hat{f} = \min_{B^T f = \delta_x} \text{val}(f)\).

**Proof:** We first show \(\text{val}(\hat{f}) \leq 10^{-6}m_\epsilon^2\). Let \(f'\) be the flow which routes \(\frac{1}{\sqrt{m}}\) units of flow on each of the \(m\) preconditions edges. As in Lemma V.3 we have that \(D_{W^V(f')} \leq 5 \cdot 10^{-7}m_\epsilon^2\). For a preconditioning edge \(e\), using Lemma V.1 and Lemma V.4.1 gives that
\[
(c_\epsilon^+)^2 \left( D - \frac{f'}{c_\epsilon^+} \right) + \left( c_\epsilon^- \right)^2 \left( D - \frac{f'}{c_\epsilon^-} \right) \leq 2(f_\epsilon)^2
\]
\[
\leq 2(\delta/m)^2 \leq 2m_\epsilon^2 \left( \frac{F_1}{10^{3m^{1/2 - \eta}}} \right)^2 \leq 10^{-8}m_\epsilon^{2-1}U^2.
\]
As \(F_1 \leq 3mU\). For the choice \(W = m_\epsilon^6\) we get that
\[
\text{val}(f') \leq \text{val}(f')
\]
\[
+ W \left\| \left( c_\epsilon^+ \right)^2 \left( D - \frac{f'}{c_\epsilon^+} \right) + \left( c_\epsilon^- \right)^2 \left( D - \frac{f'}{c_\epsilon^-} \right) \right\|_p
\]
\[
\leq 5 \cdot 10^{-7}m_\epsilon^2 + 10^{-8}m_\epsilon^{2-1-o(1)}U^2 \leq 10^{-6}m_\epsilon^2
\]
where we have used \(\|x\|_p \leq m_0^{o(1)}\|x\|_\infty\) for the choice \(p = 2\sqrt{\log m}\), and the choice of \(\eta = \log_{m_0}(m^{1/6 - o(1)}U^{1/3})\) to get \(m_\epsilon^{2-1-o(1)}U^2 \leq m_\epsilon^{2n}\).

We now show \(\text{val}(f_\epsilon) \leq \frac{1}{500}m_\epsilon^2\) for all \(e\). Indeed, applying \(\hat{D}(x) \geq x^2/4\) from Lemma V.1 yields
\[
\frac{1}{4}Wf_\epsilon^2 \leq \text{val}(f) \leq 10^{-6}m_\epsilon^2.
\]
Using the choice \(W = m_\epsilon^6\) and rearranging gives \(\text{val}(f_\epsilon) \leq \frac{1}{500}m_\epsilon^2\).

Let \(\mu\) be the weight increases given by line 7 of Algorithm 1, and Eq. (17). As \(\hat{f} = \min_{B^T f = \delta_x} D_{W_{w+\mu}}^V(f)\) and \(|w + \mu\|_1 \leq 5m/2 + m/2 \leq 3m\) by our invariant and Lemma V.5, using Lemma V.4 gives
\[
|\text{val}(f_\epsilon)| \leq \left( \frac{1}{500}m_\epsilon^2 \cdot m_\epsilon^2 \right)^{1/2} \leq \frac{1}{20}.
\]
Using that the functions \(D(x)\) and \(\tilde{D}(x)\) agree for \(|x| \leq \frac{1}{\sqrt{m}}\), and \(|f_\epsilon| \leq \frac{1}{\sqrt{m}}c_\epsilon\) for all \(e\), Observation 1 gives that \(\hat{f}\) is also a minimizer of \(\min_{B^T f = \delta_x} \text{val}(f)\) as desired.

We now show that applying weight change \(\mu\) and adding \(\hat{f}\) to our current central path point \(f_{t,w}^*\) stays on the central path, for path parameter \(t + \delta\). This follows from optimality conditions on the objective, which we designed to satisfy exactly the desired property.

As the weight change \(\mu\) may be too large, we reduce the weight change \(\mu\) to a weight change \(\nu\) after advancing the path parameter, and bound \(\|\nu\|_1\). Intuitively, this weight reduction procedure can never hurt the algorithm. It happens to help because we carefully designed our objective to induce smoothed \(l_\nu\)-flow instances with unit weights on the \(\ell_p\) part, the only instances which are known to admit almost linear runtimes [6], which we use in Section V-A.

**Lemma V.7.** Let parameters \(\eta, W, c_\epsilon^+, c_\epsilon^-, \delta, \delta, \hat{f}\), and weight changes \(\mu, \nu\) be defined as in Algorithm 1, and assume that \(F_1 \geq m^{1/2 - \eta}\). Then we have that \(\|\nu\|_1 \leq m^{2\eta} + o(1)U \) and \(f_{t+\delta,w+\nu}^* = f_{t,w}^* + \hat{f}\).

**Proof:** We first show \(f_{t+\delta,w+\nu}^* = f_{t,w+\mu}^* + \hat{f} = f_{t,w}^* + \hat{f}\).

By Lemma V.6, we have \(f_{t,w}^* = \min_{B^T f = \delta_x} \text{val}(f)\). Let the vector \(v\) be defined as in line 6 of Algorithm 1. There exist vectors \(y, z \in \mathbb{R}^p\) such that
\[
[Bz]_e = \left[ \text{val}(\hat{f}) \right]_e
\]
\[
= \left( \frac{w_\epsilon^+ + \mu_\epsilon^+}{c_\epsilon^+} + \frac{w_\epsilon^+ - \mu_\epsilon^+}{c_\epsilon^+} \cdot \frac{W(c_\epsilon^+)^2}{c_\epsilon^+} \right) D' \left( \frac{\hat{f}_\epsilon}{c_\epsilon^+} \right)
\]
\[
= \left( \frac{w_\epsilon^- + \mu_\epsilon^-}{c_\epsilon^-} + \frac{w_\epsilon^- - \mu_\epsilon^-}{c_\epsilon^-} \cdot \frac{W(c_\epsilon^-)^2}{c_\epsilon^-} \right) D' \left( \frac{\hat{f}_\epsilon}{c_\epsilon^-} \right)
\]
\[
= \left( \frac{w_\epsilon^+ + \mu_\epsilon^+}{c_\epsilon^+} + \mu_\epsilon^+ \frac{w_\epsilon^- + \mu_\epsilon^-}{c_\epsilon^-} - \frac{w_\epsilon^- + \mu_\epsilon^-}{c_\epsilon^-} \right) - \left( \frac{w_\epsilon^+ - \mu_\epsilon^+}{c_\epsilon^+} + \mu_\epsilon^+ \frac{w_\epsilon^- - \mu_\epsilon^-}{c_\epsilon^-} - \frac{w_\epsilon^- - \mu_\epsilon^-}{c_\epsilon^-} \right)\right] - [By]_e.
\]
Here, the first line follows from local optimality of \(\hat{f} = \arg\min_{B^T f = \delta_x} \text{val}(f)\), the third is explicit computation of \(D'\), and the fourth follows from centrality of \(f_{t,w+\mu}^*\).
Therefore, the $(t + \delta)\chi$-flow which is $f_{t, w + \mu} + \hat{f}$ satisfies

$$B(y + z) = \begin{bmatrix} w^+ + \mu^+ & -w^+ + \mu^- \\ u^e - f_{t, w + \mu} - \hat{f} & u^e + [f_{t, w + \mu} + \hat{f}] \end{bmatrix}$$

hence is central for weights $w + \mu$. So $f_{t, w + \mu} = f_{t, w + \mu} + \hat{f} = f_{t, w} + \hat{f}$.

Now, note that $\nu$ as defined in lines 8 to 13 of Algorithm 1 satisfies

$$\nu = \frac{\mu^+}{c^+ - \hat{f}} = \mu^- \frac{c^- + \hat{f}}{c^- - \hat{f}}$$

and centrality conditions Eq. (7) tell us that $f_{t, w + \mu} = \hat{f}$. We now bound $||\nu||_1$. Line 13 of Algorithm 1 and $\mu^+ / c^+ = \mu^- / c^-$ gives that

$$\nu^+ + \nu^- = -(e^+ - \hat{f}) \left( \mu^+ \frac{c^+ - \hat{f}}{c^+ - \hat{f}} - \mu^- \frac{c^- + \hat{f}}{c^- - \hat{f}} \right)$$

$$= -\mu^- \left( \frac{c^+ - \hat{f}}{c^+ - \hat{f}} - 1 \right) \leq 3c^+\hat{f}||\mu^+||$$

where we have used that $c^+\hat{f} \leq 1/20$. A similar analysis of line 11 gives that

$$\nu^+ + \nu^- = (\mu^+ \frac{c^+ - \hat{f}}{c^+ - \hat{f}} - \mu^- \frac{c^- + \hat{f}}{c^- - \hat{f}})$$

$$= \mu^- \left( 1 - \frac{c^+ - \hat{f}}{c^+ - \hat{f}} - \frac{c^- - \hat{f}}{c^- - \hat{f}} \right) \leq 3c^-\hat{f}||\mu^-||.$$

In both cases, we have that $\nu^+ + \nu^- \leq 3c\hat{f}||\mu^+ + \mu^-||$.

Using the choice $W = m^{6m}$, Eq. (17) and Lemma V.6 yield

$$||\nu||_1 \leq \sum_{e \in E} 3c\hat{f}||\mu^+ + \mu^-||$$

$$\leq 6W \sum_{e \in E} ||\hat{f}||c^+ \cdot \frac{v_{p-1}^p}{||v||_{p-1}}$$

$$\leq 12W \cdot \frac{1}{500} m^{-2q} U \cdot \frac{v_{p-1}^p}{||v||_{p-1}}$$

$$\leq m^{4\eta + o(1)} U \cdot \frac{v_{p-1}^p}{||v||_{p-1}} = m^{4\eta + o(1)} U.$$  

To prove Lemma V.8, we first show a generalization Theorem 2.

**Theorem 2.** For graph $G = (V, E)$ and all $e \in E$, let $0 \leq a_e \leq 2^{\text{poly}(\log m)}$ be constants, $q_e : \mathbb{R} \rightarrow \mathbb{R}$ be functions with $|q_e(0)|, |q'_e(0)| \leq 2^{\text{poly}(\log m)}$ and $a_e/4 \leq q''_e(x) \leq 4a_e$ for all $x \in \mathbb{R}$, and $h_e : \mathbb{R} \rightarrow \mathbb{R}$ be functions with $h_e(x) = h'_e(x) = 0$ and $1/4 \leq h''_e(x) \leq 4$ for all $x \in \mathbb{R}$. For demand $d$ with entries bounded by $2^{\text{poly}(\log m)}$, even integer $p \in (\omega(1), (\log n)^{2/3-o(1)})$, and all flows $f$ define

$$\text{val}(f) \overset{\text{def}}{=} \sum_{e \in E} q_e(f) + \left( \sum_{e \in E} h_e(f_e) \right)^{1/p}$$

and $\text{OPT} \overset{\text{def}}{=} \min_{B^T f = d} \text{val}(f)$.

We can compute in time $m^{1+o(1)}$ a flow $f'$ with $B^T f' = d$ and $\text{val}(f') \leq \text{OPT} + \frac{1}{2^{\text{poly}(\log m)}}$.

We do this by carefully applying and extending the analysis of the following result of [6] on smoothed $\ell_2$-lp flows.

**Theorem 3** (Theorem 1.1 in [6], arXiv version). Consider $p \in (\omega(1), (\log n)^{2/3-o(1)})$, $g \in \mathbb{R}^E$, $r \in \mathbb{R}^E$, demand vector $d \in \mathbb{R}^V$, real number $s \geq 0$, and initial solution $f_0 \in \mathbb{R}^E$ such that all parameters are bounded by $2^{\text{poly}(\log m)}$ and $B^T f_0 = d$. For a flow $f$, define

$$\text{val}_{g, r, s}(f) \overset{\text{def}}{=} \sum_{e \in E} q_e f_e + \left( \sum_{e \in E} r_e f_e^2 \right)^{1/p} + s||f||_p$$

and $\text{OPT} \overset{\text{def}}{=} \min_{B^T f = d} \text{val}_{g, r, s}(f)$.

There is an algorithm that in $m^{1+o(1)}$ time computes a flow $f$ such that $B^T f = d$ and

$$\text{val}_{g, r, s}(f) \leq \text{OPT} \leq \frac{1}{2^{\text{poly}(\log m)}}(\text{val}_{g, r, s}(f_0) - \text{OPT}) + \frac{1}{2^{\text{poly}(\log m)}}.$$

The remaining details of the proof of Lemma V.8 and Theorem 2 are deferred to the full version.

**B. Algorithm**

Here we state Algorithm 2 to show Theorem 1. It repeatedly takes steps computed with Algorithm 1, and when the remaining flow is $m^{1/2-n}$, we stop the algorithm, round to an integral flow and run augmenting paths.

**Proof of Theorem 1:** We show that $\text{MAXFLOW}(G)$ computes a maximum flow on $G$ in time $m^{3/2-n+O(1)} = m^{4/3+O(1)} U^{1/3}$ by the choice of $\eta$. Correctness follows from Lemmas V.7 and V.8.

It suffices to control the weights. Our choice of $\delta$ guarantees that we route $\Omega(m^{-1/2+\eta})$ fraction of the remaining flow per iteration, hence line 3 executes $O(m^{1/2-n})$ times.
Algorithm 2 MAXFLOW(G). Takes a preconditioned undirected graph G with maximum capacity U. Returns the maximum ab flow in G.

1: \( \eta \leftarrow \log_{m}(m^{1/6-\alpha(1)}U^{-1/3}) \).
2: \( f \leftarrow 0, t \leftarrow 0, w \leftarrow 1. \)
3: while \( F_{t} \geq m^{1/2-\eta} \) do
4: \((\delta, f, \nu) \leftarrow \text{AUGMENT}(G, w, t+1, f)\).
5: \( f \leftarrow f + f, w \leftarrow w + \nu, \text{ and } t \leftarrow \nu + \delta \).
6: end while
7: Round to an integral flow and use augmenting paths until done.

\[ \|\nu\| \leq m^{4\eta+o(1)}U \]
always by Lemma V.7, hence at the end of the algorithm by the choice of \( \eta \) we have
\[ \|w\| \leq 2m + \tilde{O} \left( m^{1/2-\eta} \cdot m^{4\eta+o(1)}U \right) \leq 5m/2. \]

To analyze the runtime, first note that by Lemma V.8 line 4 takes \( m^{1+o(1)} \) time, so the total runtime of these throughout the algorithm is \( m^{3/2-\eta+o(1)} \). Rounding to an integral flow takes \( \tilde{O}(m) \) time [4], [27]. Augmenting paths takes \( O(m^{3/2-\eta}) \) time also, as desired.

VI. CONCLUSION

We conclude by first stating the difficulties in going beyond an \( m^{4/3} \) runtime for maxflow, and then discussing potential directions for future research on the topic.

A Possible Barrier at \( m^{4/3} \): Here we briefly discuss why we believe that \( m^{4/3} \) is a natural runtime barrier for IPM based algorithms for maxflow on sparse unweighted graphs. All currently known weighted IPM advances satisfy that the weights only increase and do not become super linear and that the methods step from one central path point to the next one in \( O(m) \) time and the congestion of this step is multiplicatively bounded. Our algorithm precisely computes the weight changes under a budget to ensure that the congestion to the next central path point is reduced significantly. In this sense, to break the \( m^{4/3} \) barrier, one would have to depart from this template and e.g., backtrack on weight changes on the central path so that they are not additive throughout the algorithm, show better amortized bounds on weight change than shown here, or provide a novel algorithm to step to faraway points on the central path, outside a ball where the congestions are bounded.

Potential future directions: An interesting question is potentially achieving a \( mn^{1/3+o(1)} \) time algorithm for maxflow through the approach of [2] and robust central paths [23], [24]. It would also be interesting to understand whether IPMs can achieve \( m^{3/2-\Omega(1)} \) \( \log^{3}(1)U \) runtimes for maxflow or understand whether \( m^{4/3} \) barrier described above can be broken.


