2013 IEEE 54th Annual Symposium on Foundations of Computer Science

# Navigating Central Path with Electrical Flows: from Flows to Matchings, and Back[†] (Extended Abstract)

Aleksander Mądry[*]
*EPFL*
*Lausanne, Switzerland*
*aleksander.madry@epfl.ch*

***Abstract*—We present an $\widetilde{O}(m^{\frac{10}{7}}) = \widetilde{O}(m^{1.43})$-time[1] algorithm for the maximum $s$-$t$ flow and the minimum $s$-$t$ cut problems in directed graphs with unit capacities. This is the first improvement over the sparse-graph case of the long-standing $O(m \min\{\sqrt{m}, n^{2/3}\})$ running time bound due to Even and Tarjan [16]. By well-known reductions, this also establishes an $\widetilde{O}(m^{\frac{10}{7}})$-time algorithm for the maximum-cardinality bipartite matching problem. That, in turn, gives an improvement over the celebrated $O(m\sqrt{n})$ running time bound of Hopcroft and Karp [25] whenever the input graph is sufficiently sparse.**

**At a very high level, our results stem from acquiring a deeper understanding of interior-point methods – a powerful tool in convex optimization – in the context of flow problems, as well as, utilizing certain interplay between maximum flows and bipartite matchings.**

***Keywords*-maximum flow problem; minimum s-t cut problem; bipartite matchings; electrical flows; interior-point methods; central path; Laplacian linear systems;**

## I. Introduction

The maximum $s$-$t$ flow problem and its dual, the minimum $s$-$t$ cut problem, are two of the most fundamental and extensively studied graph problems in combinatorial optimization [45], [2]. They have a wide range of applications (see [3]), are often used as subroutines in other algorithms (see, e.g., [5], [46]), and a number of other important problems – e.g., bipartite matching problem [11] – can be reduced to them. Furthermore, these two problems were often a testbed for development of fundamental algorithmic tools and concepts. Most prominently, the Max-Flow Min-Cut theorem [15], [18] constitutes the prototypical primal-dual relation.

Several decades of extensive work resulted in a number of developments on these problems (see Goldberg and Rao [22] for an overview) and many of their generalizations and special cases. Still, despite all this effort, the basic problem of computing maximum $s$-$t$ flow and minimum $s$-$t$ cut in general graphs has resisted progress for a long time. In particular, the current best running time bound of $O(m \min\{m^{\frac{1}{2}}, n^{\frac{2}{3}}\} \log(n^2/m) \log U)$ (with $U$ denoting the largest arc capacity) was established over 15 years ago in a breakthrough paper by Goldberg and Rao [22] and this bound, in turn, matches the $O(m \min\{m^{\frac{1}{2}}, n^{\frac{2}{3}}\})$ bound for unit-capacity graphs that Even and Tarjan [16] put forth over 35 years ago.

Recently, however, important progress was made in the context of undirected graphs. Christiano et al. [9] developed an algorithm that allows one to compute a $(1+\varepsilon)$-approximation to the undirected maximum $s$-$t$ flow (and the minimum $s$-$t$ cut) problem in $\widetilde{O}(mn^{\frac{1}{3}}\varepsilon^{-11/3})$ time. Their result relies on devising a new approach to the problem that combines electrical flow computations with multiplicative weights update method (see [5]). Later, Lee et al. [34] presented a quite different – but still electrical-flow-based – algorithm that employs purely gradient-descent-type view to obtain an $\widetilde{O}(mn^{1/3}\varepsilon^{-2/3})$-time $(1+\varepsilon)$-approximation for the case of unit capacities. Finally, very recently, this line of work was culminated by Sherman [46] and Kelner et al. [26] who independently showed how to integrate non-Euclidean gradient-descent methods with fast poly-logarithmic-approximation algorithms for cut problems of Mądry [37] to get an $O(m^{1+o(1)}\varepsilon^{-2})$-time $(1+\varepsilon)$-approximation to the undirected maximum flow problem.

Finally, we note that, in parallel to the above work that is focused on designing weakly-polynomial algorithms for the maximum $s$-$t$ flow and minimum $s$-$t$ cut problems, there is also a considerable interest in obtaining running time bounds that are strongly-polynomial, i.e., that do not depend on the values of arc capacities. The current best such bound is $O(mn)$ and it follows by combining the algorithms of King et al. [29] and Orlin [43].

*Bipartite Matching Problem:* Another problem that we will be interested in is the (maximum-cardinality) bipartite matching problem – a fundamental assignment problem with numerous applications (see, e.g., [2], [36]) and long history. Already in 1931, König [30] and Egerváry [14] provided first constructive characterization of maximum matchings in bipartite graphs. This characterization can be turned into a polynomial-time algorithm. Then, in 1973, Hopcroft and Karp [25] devised the celebrated $O(m\sqrt{n})$-time algorithm. Till date, this bound is the best one known in the regime of

[†]A full version of this paper is available as [39].

[*]Part of this work was done when the author was with Microsoft Research New England.

[1]We recall that $\widetilde{O}(f)$ denotes $O(fpoly(\log f))$.

0272-5428/13 $26.00 © 2013 IEEE
DOI 10.1109/FOCS.2013.35

relatively sparse graphs. It can be improved, however, when the input graph is dense, i.e., when $m$ is close to $n^2$. In this case, one can combine the algebraic approach of Rabin and Vazirani [44] – that itself builds on the work of Tutte [50] and Lovász [35] – with matrix-inversion techniques of Bunch and Hopcroft [8] to get an algorithm that runs in $O(n^\omega)$ time (see [41]), where $\omega \leq 2.3727$ is the exponent of matrix multiplication [10], [51]. Also, later on, Alt et al. [4], as well as, Feder and Motwani [17] developed combinatorial algorithms that offer a slight improvement – by a factor of, roughly, $\log_n \frac{n^2}{m}$ – over the $O(m\sqrt{n})$ bound of Hopcroft and Karp whenever the graph is sufficiently dense.

Finally, it is worth mentioning that there was also a lot of developments on the (maximum-cardinality) matching problem in general, i.e., not necessarily bipartite, graphs. Starting with the pioneering work of Edmonds [13], these developments led to bounds that essentially match the running time guarantees that were previously known only for bipartite case. More specifically, the running time bound of $O(m\sqrt{n})$ for the general-graph case was obtained by Micali and Vazirani [40], [52] (see also [19] and [21]). While, building on the algebraic characterization of the problem due to Rabin and Vazirani [44], Mucha and Sankowski [42] and then Harvey [23] gave $O(n^\omega)$-time algorithms for general graphs.

### A. Our Contribution

In this paper, we develop a new algorithm for solving maximum $s$-$t$ flow and minimum $s$-$t$ cut problems in directed graphs. More precisely, we prove the following theorem.

**Theorem I.1.** *Let $G = (V, E)$ be a directed graph with $m$ arcs and unit capacities. For any two vertices $s$ and $t$, one can compute an integral maximum $s$-$t$ flow and minimum $s$-$t$ cut of $G$ in $\widetilde{O}(m^{\frac{10}{7}})$ time.*

This improves over the long-standing $O(m \min\{\sqrt{m}, n^{2/3}\})$ running time bound due to Even and Tarjan [16] and, in particular, finally breaks the $\Omega(n^{\frac{3}{2}})$ running time barrier for sparse directed graphs.

Furthermore, by applying a well-known reduction (see [11]), our new algorithm gives the first improvement on the sparse-graph case of the seminal Hopcroft-Karp algorithm [25] for the maximum-cardinality bipartite matching problem.

**Theorem I.2.** *Let $G = (V, E)$ be an undirected bipartite graph with $m$ edges, one can solve the maximum-cardinality bipartite matching problem in $G$ in $\widetilde{O}(m^{\frac{10}{7}})$ time.*

This, again, breaks the 40-years-old running time barrier of $\Omega(n^{\frac{3}{2}})$ for this problem in sparse graphs.

Additionally, we design a simple reduction of the maximum $s$-$t$ flow problem to perfect bipartite $\boldsymbol{b}$-matching problem (see Theorem III.1). (This reduction can be seen as an adaptation of the reduction of the maximum vertex-disjoint $s$-$t$-path problem to the bipartite matching problem due to Hoffman [24] – cf. Section 16.7c in [45].[2]) As the reduction in the other direction is well-known already, this establishes an algorithmic equivalence of these two problems. We also show (see Theorem III.3 and Corollary III.4) how this reduction, together with the sub-linear-time algorithm for perfect matching problem in regular bipartite graphs of Goel et al. [20], leads to an efficient, nearly-linear time, rounding procedure for $s$-$t$ flows.[3]

Finally, our main technical contribution is a primal-dual algorithm for (near-)perfect bipartite $\boldsymbol{b}$-matching problem (see Theorem III.2). This iterative algorithm draws on ideas underlying interior-point methods and the electrical flow framework of Christiano et al. [9]. It employs electrical flow computations to gradually improve the quality of maintained solution by advancing it toward (near-)optimality along so-called central path.

We develop a way of analyzing this algorithm's rate of convergence by relating it to the structure of the corresponding electrical flows. This understanding enables us to devise a way of perturbing and preconditioning our intermediate solutions to ensure a convergence in only $\widetilde{O}(m^{\frac{3}{7}})$ iterations and thus improve over the well-known barrier of $\Omega(m^{\frac{1}{2}})$ iterations that all the previous interior-point-methods-based algorithms suffer from. (To the best of our knowledge, this is the first time that this barrier was broken for a natural optimization problem.)

We also note that most of this understanding of convergence behavior of interior-point methods can be carried over to general LP setting. Therefore, we are hopeful that our techniques can be extended and will eventually lead to breaking the $\Omega(m^{\frac{1}{2}})$ iterations barrier for general interior-point methods.

### B. Our Approach

The core of our approach comprises two components. One of them is combinatorial in nature and exploits an intimate connection between the maximum $s$-$t$ flow problem and bipartite matching problem. The other one is more linear-algebraic and relies on interplay of interior-point methods and electrical flows.

*Maximum flows and bipartite matchings:* The combinatorial component shows that not only one can reduce bipartite matching problem to the maximum $s$-$t$ flow problem, but also that a reduction in the other direction exists. Namely, one can reduce, in a simple and purely combinatorial way, the maximum $s$-$t$ flow problem to a certain variant of bipartite matching problem (see Theorem III.1). Once this reduction is established, it allows us to shift our attention to the matching problem.

[2]We thank Lap Chi Lau [33] for pointing out this similarity.

[3]Recently, it came to our attention that a very similar rounding result was independently obtained by Khanna et al. [28].

Also, as a byproduct, this reduction – together with the algorithm of Goel et al. [20] – yields a fast procedure for rounding fractional maximum flows (see Corollary III.4). This enables us to focus on obtaining solutions that are only nearly-optimal, instead of being optimal.

*Bipartite Matchings and Electrical Flows:* The other component is based on using the interior-point method framework in conjunction with nearly-linear time electrical flow computations, to develop a faster algorithm for the bipartite matching problem.

The point of start here is a realization that the recent approaches to approximating undirected maximum flow [9], [34], [47], [26], despite achieving impressive progress, have fundamental limitations that make them unlikely to yield improvements for the exact undirected or (approximate) directed setting.[4] Very roughly speaking, these limitations stem from the fact that, at their core, all these algorithms employ some version of gradient-descent method that relies on purely primal arguments, while almost completely neglecting the dual aspect of the problem. It is well-understood, however, that getting a running time guarantee that depends logarithmically, instead of polynomially, on $\varepsilon^{-1}$ – and such dependence is a prerequisite to making progress in directed setting – one needs to also embrace the dual side of the problem and take full advantage of it.

*Interior-point methods and fast algorithms:* The above realization motivates us to consider a more sophisticated approach, one that is inherently primal-dual and achieves logarithmic dependence on $\varepsilon^{-1}$: interior-point methods. These methods constitute a powerful optimization paradigm that is a cornerstone of convex optimization (see, e.g., [7], [53], [54]) and already led to development of polynomial-time exact algorithms for a variety of problems. Unfortunately, despite all its advantages and successes in tackling hard optimization tasks, this paradigm has certain shortcomings in the context of designing fast algorithms. The main reason for that is the fact that each iteration of interior-point method requires solving of a linear system, a task for which the current fastest general-purpose algorithm runs in $O(n^{\omega}) = O(n^{2.3727})$ time [1], [10], [51]. So, this bound becomes a bottleneck if one was aiming for, say, even sub-quadratic-time algorithm.

Fortunately, it turns out that there is a way to circumvent this issue. Namely, even though the above bound is the best one known in general, one can get a better running time when dealing with some specific problem. This is achieved by exploiting the special structure of the corresponding linear systems. A prominent (and most important from our point of view) example here is the family of flow problems. Daitch and Spielman [12] showed that in the context of flow problems one can use the power of fast (approximate) Laplacian system solvers [49], [31], [32], [27] to solve the corresponding linear systems in nearly-linear time. This enabled [12] to develop a host of $\widetilde{O}(m^{\frac{3}{2}})$-time algorithms for a number of important generalizations of the maximum flow problem for which there was no such algorithms before.

Unfortunately, this bound of $\widetilde{O}(m^{\frac{3}{2}})$ time turns out to also be a barrier if one wants to obtain even faster algorithms. The new difficulty here is that the best worst-case bound on the number of iterations needed for an interior-point method to converge to near-optimal solution is $\Omega(m^{1/2})$. Although it is widely believed that this bound is far from optimal, it seems that our theoretical understanding of interior-point method convergence is still insufficient to make any progress on this front. In fact, improving this state of affairs is a major and long-standing challenge in mathematical programing.

*Beyond the $\Omega(m^{\frac{1}{2}})$ barrier:* Our approach to circumventing this $\Omega(m^{\frac{1}{2}})$ barrier and obtaining the desired $\widetilde{O}(m^{\frac{10}{7}})$-time algorithm for the bipartite $\boldsymbol{b}$-matching problem consists of two stages.

First one corresponds to setting up a primal-dual framework for solving the near-perfect $\boldsymbol{b}$-matching problem. This framework is directly inspired by the principles underlying path-following interior-point methods and, in some sense, is equivalent to them. In it, we start with some initial sub-optimal solution (that is encoded as a minimum-cost flow problem instance) and gradually improve its quality up to near-optimality. These gradual improvements are guided by certain electrical flow computations – the flows are used to update the primal solution and the corresponding voltages update the dual one – and our solution ends up following a special trajectory in the feasible space: so-called central path.

We analyze the performance of this optimization process by establishing a formal connection that ties the size of each improvement step to a certain characteristic of the corresponding electrical flow. Very roughly speaking, this size (and thus the resulting rate of convergence) is directly related to how much the electrical flow we compute resembles the current primal solution (which is also a flow). Once this connection is established, a simple energy-based argument immediately recovers the generic $O(m^{\frac{1}{2}})$ iterations bound known for interior-point methods. So, as each electrical flow computation can be performed in $\widetilde{O}(m)$ time, this gives an overall $\widetilde{O}(m^{\frac{3}{2}})$-time algorithm.

Finally, to improve upon the above $O(m^{\frac{1}{2}})$ iterations bound and deliver the desired $O(m^{\frac{10}{7}})$-time procedure, we devise two techniques: perturbation of arcs – that can be seen as a refinement of the edge removal technique of Christiano et al. [9]; and solution preconditioning – a way of adding auxiliary arcs to the solution to improve its conductance properties. We show that by a careful composition of these techniques, one is able to ensure that the guiding electrical flows align better with the primal solution – thus allowing taking larger progress steps and guaranteeing faster con-

[4]Note that it is known – see, e.g., [38] – that computing exact maximum $s$-$t$ flow in undirected graphs is algorithmically equivalent to computing the exact or approximate maximum $s$-$t$ flow in directed graph.

vergence – while keeping the unwanted impact of these modifications on the quality of final solution minimal. The analysis of this process constitutes the technical core of our result and is based on understanding of the interplay between the interior-point method and both the primal and dual structure of electrical flows.

We believe that this approach of understanding interior-point methods through the lens of electrical flows is a promising direction and our result is just a first step towards realizing its full potential.

## II. Preliminaries

In this section, we introduce some basic notation and definitions we will need later.

### A. $\boldsymbol{\sigma}$-Flows and the Maximum $s$-$t$ Flow Problem

Throughout this paper, we denote by $G = (V, E, \boldsymbol{u})$ a directed graph with vertex set $V$, arc set $E$ (we allow parallel arcs), and (non-negative) integer capacities $u_e$, for each arc $e \in E$. We usually define $m = |E|$ to be the number of arcs of the graph in question and $n = |V|$ to be the number of its vertices. Each arc $e$ of $G$ is an ordered pair $(u, v)$, where $u$ is its *tail* and $v$ is its *head*.

The basic notion of this paper is the notion of a $\boldsymbol{\sigma}$*-flow* in $G$, where $\boldsymbol{\sigma} \in \mathbb{R}^n$, with $\sum_v \sigma_v = 0$, is the *demand vector*. By a $\boldsymbol{\sigma}$-flow in $G$ we understand any vector $\boldsymbol{f} \in \mathbb{R}^m$ that assigns values to arcs $G$ and satisfies the *flow conservation constraints*:

$$\sum_{e \in E^+(v)} f_e - \sum_{e \in E^-(v)} f_e = \sigma_v, \quad \text{for each vertex } v \in V. \tag{1}$$

Here, $E^+(v)$ (resp. $E^-(v)$) is the set of arcs of $G$ that are leaving (resp. entering) vertex $v$. Intuitively, these constraints enforce that the net balance of the total in-flow into vertex $v$ and the total out-flow out of that vertex is equal to $\sigma_v$, for every $v \in V$.

Furthermore, we say that a $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ is *feasible* in $G$ iff $\boldsymbol{f}$ obeys the *non-negativity and capacity constraints*:

$$0 \leq f_e \leq u_e, \quad \text{for each arc } e \in E. \tag{2}$$

One type of $\boldsymbol{\sigma}$-flows that will be of special interest to us are $s$-$t$ flows, where $s$ (the *source*) and $t$ (the *sink*) are two distinguish vertices of $G$. Formally, a $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ is an $s$-$t$ *flow* iff its demand vector $\boldsymbol{\sigma}$ is equal to $F \cdot \chi_{s,t}$ for some $F \geq 0$ – we call $F$ the *value* of $\boldsymbol{f}$ – and the demand vector $\chi_{s,t}$ that has $-1$ (resp. 1) at the coordinate corresponding to $s$ (resp. $t$) and zeros everywhere else.

Now, the *maximum $s$-$t$ flow problem* corresponds to a task of finding for a given graph $G = (V, E, \boldsymbol{u})$, a source $s$, and a sink $t$, a feasible $s$-$t$ flow $\boldsymbol{f}^*$ in $G$ of maximum value $F$. We call such a flow $\boldsymbol{f}^*$ that maximizes $F$ *the maximum $s$-$t$ flow of $G$* and denote its value by $F^*$.

Sometimes, we will be also interested in (uncapacitated) *minimum-cost $\boldsymbol{\sigma}$-flow* problem (with non-negative costs). In this problem, we have a directed graph $G$ with infinite capacities on arcs (i.e., $u_e = +\infty$, for all $e$) and certain *(non-negative) length* (or *cost*) $l_e$ assigned to each arc $e$. Our goal is to find a feasible $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ in $G$ whose *cost* $l(\boldsymbol{f}) := \sum_e l_e f_e$ is minimal. (Note that as we have infinite capacities here, the feasibility constraint (2) just requires that $f_e \geq 0$ for all arcs $e$.)

Finally, one more problem that will be relevant in this context is the *minimum $s$-$t$ cut* problem. In this problem, we are given a directed graph $G = (V, E, \boldsymbol{u})$ with integer capacities, as well as, a source $s$ and sink $t$, and our task is to find an $s$-$t$ cut $C \subseteq V$ in $G$ minimizes the *capacity* $\boldsymbol{u}(C) := \sum_{E^-(C)} u_e$ among all $s$-$t$ cuts. Here, a cut $C \subseteq V$ is an $s$-$t$ *cut* iff $s \in C$ and $t \notin C$, and $E^-(C)$ is the set of all arcs $(u, v)$ with $u \in C$ and $v \notin C$. It is well-known [15], [18] that the minimum $s$-$t$ cut problem is the dual of the maximum $s$-$t$ problem and, in particular, that the capacity of the minimum $s$-$t$ cut is equal to the value of the maximum $s$-$t$ flow, as well as, that given a maximum $s$-$t$ flow one can easily obtain the corresponding minimum $s$-$t$ cut.

### B. Undirected Graphs

Although the focus of our results is on directed graphs, it will be crucial for us to consider undirected graphs too. To this end, we view an undirected graph $G = (V, E, \boldsymbol{u})$ as a directed one in which the ordered pair $(u, v) \in E$ does not denote an arc anymore, but an (undirected) *edge* $(u, v)$ and the order just specifies an *orientation* of that edge from $u$ to $v$. (Even though we use the same notation for these two different types of graphs, we will always make sure that it is clear from the context whether we deal with directed graph that has arcs, or with undirected graph that has edges.) From this perspective, the definitions of $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ that we introduced above for directed graphs transfer over to undirected setting almost immediately. The only (but very crucial) difference is that in undirected graphs a feasible flow can have some of $f_e$s being negative - this corresponds to the flow flowing in the direction that is opposite to the edge orientation. As a result, the feasibility condition (2) becomes

$$|f_e| \leq u_e, \quad \text{for each arc } e \in E. \tag{3}$$

Also, the set $E^+(v)$ (resp. $E^-(v)$) denotes now the set of incident edges that are oriented towards (resp. away) from $v$, and $E(v) := E^+(v) \cup E^-(v)$ is just the set of all edges incident to $v$, regardless of their orientation.

Finally, given a directed graph $G = (V, E, \boldsymbol{u})$, by its *projection* $\bar{G}$ we understand an undirected graph that arises from treating each arc of $G$ as an edge with the corresponding orientation. Note that if $G$ had two arcs $(u, v)$ and $(v, u)$ then $\bar{G}$ will have two parallel edges $(u, v)$ and $(v, u)$ that have opposite orientation and, possibly, different capacities.

### C. Electrical Flows and Potentials

A notion that will play a fundamental role in this paper is the notion of electrical flows. Here, we just briefly review

some of the key properties that we will need later. For an in-depth treatment we refer the reader to [6].

Consider an undirected graph $G$ and some vector of resistances $\boldsymbol{r} \in \mathbb{R}^m$ that assigns to each edge $e$ its *resistance* $r_e > 0$. For a given $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ in $G$, let us define its *energy* (with respect to $\boldsymbol{r}$) $\mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f})$ to be

$$\mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f}) := \sum_e r_e f_e^2 = \boldsymbol{f}^T \boldsymbol{R} \boldsymbol{f}, \tag{4}$$

where $\boldsymbol{R}$ is an $m \times m$ diagonal matrix with $R_{e,e} = r_e$, for each edge $e$.

For a given undirected graph $G$, a demand vector $\boldsymbol{\sigma}$, and a vector of resistances $\boldsymbol{r}$, we define an *electrical $\boldsymbol{\sigma}$-flow* in $G$ (that is *determined* by resistances $\boldsymbol{r}$) to be the $\boldsymbol{\sigma}$-flow that minimizes the energy $\mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f})$ among all $\boldsymbol{\sigma}$-flows in $G$. As energy is a strictly convex function, one can easily see that such a flow is unique. Also, we emphasize that we do *not* require here that this flow is feasible with respect to capacities of $G$ (cf. (3)). Furthermore, whenever we consider electrical flows in the context of a directed graph $G$, we will mean an electrical flow – as defined above – in the (undirected) projection $\bar{G}$ of $G$.

One of very useful properties of electrical flows is that it can be characterized in terms of vertex potentials inducing it. Namely, one can show that a $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ in $G$ is an electrical $\boldsymbol{\sigma}$-flow determined by resistances $\boldsymbol{r}$ iff there exist *vertex potentials* $\phi_v$ (that we collect into a vector $\boldsymbol{\phi} \in \mathbb{R}^n$) such that, for any edge $e = (u, v)$ in $G$ that is oriented from $u$ to $v$,

$$f_e = \frac{\phi_v - \phi_u}{r_e}. \tag{5}$$

In other words, a $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ is an electrical $\boldsymbol{\sigma}$-flow iff it is *induced* via (5) by some vertex potential $\boldsymbol{\phi}$. (Note that orientation of edges matters in this definition.)

Using vertex potentials, we are able to express the energy $\mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f})$ (see (4)) of an electrical $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ in terms of the potentials $\boldsymbol{\phi}$ inducing it as

$$\mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f}) = \sum_{e=(u,v)} \frac{(\phi_v - \phi_u)^2}{r_e}. \tag{6}$$

One of the consequences of this characterization of electrical flows via vertex potentials is that one can view the energy of an electrical $\boldsymbol{\sigma}$-flow as being a result of optimization not over all the $\boldsymbol{\sigma}$-flows but rather over certain set of vertex potentials. Namely, we have the following lemma that, for completeness, we prove in the full version of the paper [39].

**Lemma II.1.** *For any graph $G = (V, E)$, any vector of resistances $\boldsymbol{r}$, and any demand vector $\boldsymbol{\sigma}$,*

$$\frac{1}{\mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f}^*)} = \min_{\boldsymbol{\phi} | \boldsymbol{\sigma}^T \boldsymbol{\phi} = 1} \sum_{e=(u,v) \in E} \frac{(\phi_v - \phi_u)^2}{r_e},$$

*where $\boldsymbol{f}^*$ is the electrical $\boldsymbol{\sigma}$-flow determined by $\boldsymbol{r}$ in $G$. Furthermore, if $\boldsymbol{\phi}^*$ are the vertex potentials corresponding to $\boldsymbol{f}^*$ then the minimum is attained by taking $\boldsymbol{\phi}$ to be equal to $\tilde{\boldsymbol{\phi}} := \boldsymbol{\phi}^* / \mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f}^*)$.*

Note that the above lemma provides a convenient way of lowerbounding the energy of an electrical $\boldsymbol{\sigma}$-flow. One just needs to expose any vertex potentials $\boldsymbol{\phi}$ such that $\boldsymbol{\sigma}^T \boldsymbol{\phi} = 1$ and this will immediately constitute an energy lowerbound. Also, another basic but useful property of electrical $\boldsymbol{\sigma}$-flows is captured by the following fact.

**Fact II.2** (Rayleigh Monotonicity)**.** *For any graph $G = (V, E)$, demand vector $\boldsymbol{\sigma}$ and any two vectors of resistances $\boldsymbol{r}$ and $\boldsymbol{r}'$ such that $r_e \geq r'_e$, for all $e \in E$, we have that if $\boldsymbol{f}$ (resp. $\boldsymbol{f}'$) is the electrical $\boldsymbol{\sigma}$-flow determined by $\boldsymbol{r}$ (resp. $\boldsymbol{r}'$) then*

$$\mathcal{E}_{\boldsymbol{r}}(\boldsymbol{f}) \geq \mathcal{E}_{\boldsymbol{r}'}(\boldsymbol{f}').$$

### *D. Laplacian Solvers*

A very important algorithmic property of electrical flows is that one can compute very good approximations of them in nearly-linear time. Below, we briefly describe the tools enabling that.

To this end, let us recall that electrical $\boldsymbol{\sigma}$-flow is the (unique) $\boldsymbol{\sigma}$-flow induced by vertex potentials via (5). So, finding such a flow boils down to computing the corresponding vertex potentials $\boldsymbol{\phi}$. It turns out that computing these potentials can be cast as a task of solving certain type of linear system called *Laplacian* systems. To see that, let us define the *edge-vertex incidence matrix* $\boldsymbol{B}$ being an $n \times m$ matrix with rows indexed by vertices and columns indexed by edges such that

$$\boldsymbol{B}_{v,e} = \begin{cases} 1 & \text{if } e \in E^+(v), \\ -1 & \text{if } e \in E^-(v), \\ 0 & \text{otherwise.} \end{cases}$$

Now, we can compactly express the flow conservation constraints (1) of a $\boldsymbol{\sigma}$-flow $\boldsymbol{f}$ (that we view as a vector in $\mathbb{R}^m$) as

$$\boldsymbol{B}\boldsymbol{f} = \boldsymbol{\sigma}.$$

On the other hand, if $\boldsymbol{\phi}$ are some vertex potentials, the corresponding flow $\boldsymbol{f}$ induced by $\boldsymbol{\phi}$ via (5) (with respect to resistances $\boldsymbol{r}$) can be written as

$$\boldsymbol{f} = \boldsymbol{R}^{-1} \boldsymbol{B}^T \boldsymbol{\phi},$$

where again $\boldsymbol{R}$ is a diagonal $m \times m$ matrix with $R_{e,e} := r_e$, for each edge $e$.

Putting the two above equations together, we get that the vertex potentials $\boldsymbol{\phi}$ that induce the electrical $\boldsymbol{\sigma}$-flow determined by resistances $\boldsymbol{r}$ are given by a solution to the following linear system

$$\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{\phi} = \boldsymbol{L}\boldsymbol{\phi} = \boldsymbol{\sigma}, \tag{7}$$

where $\boldsymbol{L} := \boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T$ is the (weighted) *Laplacian* $\boldsymbol{L}$ of $G$ (with respect to the resistances $\boldsymbol{r}$). One can easily check that

$\boldsymbol{L}$ is an $n \times n$ matrix indexed by vertices of $G$ with entries given by

$$L_{u,v} = \begin{cases} \sum_{e \in E(v)} 1/r_e & \text{if } u = v, \\ -1/r_e & \text{if } e = (u,v) \in E, \text{ and} \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

One can see that the Laplacian $\boldsymbol{L}$ is not invertible, but – as long as, the underlying graph is connected – it's null-space is one-dimensional and spanned by all-ones vector. As we require our demand vectors $\boldsymbol{\sigma}$ to have its entries sum up to zero (otherwise, no $\boldsymbol{\sigma}$-flow can exist), this means that they are always orthogonal to that null-space. Therefore, the linear system (7) has always a solution $\boldsymbol{\phi}$ and one of these solutions[5] is given by

$$\boldsymbol{\phi} = \boldsymbol{L}^{\dagger}\boldsymbol{\sigma},$$

where $\boldsymbol{L}^{\dagger}$ is the Moore-Penrose pseudo-inverse of $\boldsymbol{L}$.

Now, from the algorithmic point of view, the crucial property of the Laplacian $\boldsymbol{L}$ is that it is symmetric and *diagonally dominant*, i.e., for any $v \in V$, $\sum_{u \neq v} |L_{u,v}| \leq L_{v,v}$. This enables us to use fast approximate solvers for symmetric and diagonally dominant linear systems to compute an approximate electrical $\boldsymbol{\sigma}$-flow. Namely, building on the work of Spielman and Teng [48], [49], Koutis et al. [31], [32] designed an SDD linear system solver that implies the following theorem. (See also recent work of Kelner et al. [27] that presents an even simpler nearly-linear-time Laplacian solver.)

**Theorem II.3.** *For any $\varepsilon > 0$, any graph $G$ with $n$ vertices and $m$ edges, any demand vector $\boldsymbol{\sigma}$, and any resistances $\boldsymbol{r}$, one can compute in $\widetilde{O}(m \log m \log \varepsilon^{-1})$ time vertex potentials $\tilde{\boldsymbol{\phi}}$ such that $\|\tilde{\boldsymbol{\phi}} - \boldsymbol{\phi}^*\|_{\boldsymbol{L}} \leq \varepsilon \|\boldsymbol{\phi}^*\|_{\boldsymbol{L}}$, where $\boldsymbol{L}$ is the Laplacian of $G$, $\boldsymbol{\phi}^*$ are potentials inducing the electrical $\boldsymbol{\sigma}$-flow determined by resistances $\boldsymbol{r}$, and $\|\boldsymbol{\phi}\|_{\boldsymbol{L}} := \sqrt{\boldsymbol{\phi}^T \boldsymbol{L} \boldsymbol{\phi}}$.*

To understand the type of approximation offered by the above theorem, observe that $\|\boldsymbol{\phi}\|_{\boldsymbol{L}}^2 = \boldsymbol{\phi}^T \boldsymbol{L} \boldsymbol{\phi}$ is just the energy of the flow induced by vertex potentials $\boldsymbol{\phi}$. Therefore, $\|\tilde{\boldsymbol{\phi}} - \boldsymbol{\phi}^*\|_{\boldsymbol{L}}$ is the energy of the electrical flow $\bar{\boldsymbol{f}}$ that "corrects" the vertex demands of the electrical $\tilde{\boldsymbol{\sigma}}$-flow induced by potentials $\tilde{\boldsymbol{\phi}}$, to the ones that are dictated by $\boldsymbol{\sigma}$. So, in other words, the above theorem tells us that we can quickly find an electrical $\tilde{\boldsymbol{\sigma}}$-flow $\tilde{\boldsymbol{f}}$ in $G$ such that $\tilde{\boldsymbol{\sigma}}$ is a slightly perturbed version of $\boldsymbol{\sigma}$ and $\tilde{\boldsymbol{f}}$ can be corrected to the electrical $\boldsymbol{\sigma}$-flow $\boldsymbol{f}^*$ that we are seeking, by adding to it some electrical flow $\bar{\boldsymbol{f}}$ whose energy is at most $\varepsilon$ fraction of the energy of the flow $\boldsymbol{f}^*$. (Note that electrical flows are linear, so we indeed have that $\boldsymbol{f}^* = \tilde{\boldsymbol{f}} + \bar{\boldsymbol{f}}$.) As we will see, this kind of approximation is completely sufficient for our purposes.

[5]Note that the linear system (7) will have many solutions, but each two of them are equivalent up to a translation. So, as the formula (5) is translation-invariant, each of these solutions will yield the same unique electrical $\boldsymbol{\sigma}$-flow.

### *E. Bipartite $\boldsymbol{b}$-Matchings*

A fundamental graph problem that constitutes both an application of our results, as well as, one of the tools we use to establish them, is the *(maximum-cardinality) bipartite $\boldsymbol{b}$-matching problem*. In this problem, we are given an undirected bipartite graph $G = (V, E)$ with $V = P \cup Q$ – where $P$ and $Q$ are the two sets of bipartition – as well as, a *demand vector* $\boldsymbol{b}$ that assigns to every vertex $v$ an integral and positive demand $b_v$. Our goal is to find a maximum cardinality *multiset* $M$ of the edges of $G$ that forms a *$\boldsymbol{b}$-matching*. That is, we want to find a multi-set $M$ of edges of $G$ that is of maximum cardinality subject to a constraint that, for each vertex $v \in V$, the number of edges of $M$ that are incident to $v$ is at most $b_v$. (When $b_v = 1$ for every vertex $v$, we will simply call such $M$ a *matching*.)

We say that a $\boldsymbol{b}$-matching $M$ is *perfect* iff every vertex in $V$ has exactly $b_v$ edges incident to it in $M$. Note that a perfect $\boldsymbol{b}$-matching - if it exists in $G$ - has to necessarily be of maximum cardinality. Also, if a graph has a perfect $\boldsymbol{b}$-matching then it must be that $\sum_{v \in P} b_v = \sum_{v \in Q} b_v$. Now, by the *perfect bipartite $\boldsymbol{b}$-matching problem* we mean a task in which we need to either find the perfect $\boldsymbol{b}$-matching in $G$ or conclude that it does not exist.

Finally, by a *fractional* solution to a $\boldsymbol{b}$-matching problem, we understand an $|E|$-dimensional vector $\boldsymbol{x}$ that allocates non-negative value of $x_e$ to each edge $e$ and is such that for every vertex $v$ of $G$, the sum $\sum_{e \in E(v)} x_e$ of (fractional) incident edges in $\boldsymbol{x}$ is at most $b_v$. Also, we define the *size* of a fractional $\boldsymbol{b}$-matching $\boldsymbol{x}$ to be $|\boldsymbol{x}|_1$.

An interesting class of graphs that is guaranteed to always have a perfect matching are bipartite graphs that are $d$-*regular*, i.e., that have the degree of each vertex equal to $d$. A remarkable algorithm of Goel et al. [20] shows that one can find a perfect matching in such graphs in time that is proportional only to number of its vertices and not edges. (Note that a $d$-regular bipartite graph has exactly $\frac{dn}{2}$ edges and thus this number can be much higher than $n$ when $d$ is large.) In particular, they prove the following theorem that we will use later.

**Theorem II.4** (see Theorem 4 in [20])**.** *Given an $n \times n$ doubly-stochastic matrix $\boldsymbol{M}$ with $m$ non-zero entries, one can find a perfect matching in the support of $M$ in $O(n \log^2 n)$ expected time with $O(m)$ preprocessing time.*

## III. From Flows to Matchings, and Back

As we already mentioned, our results stem from exploiting the interplay between the maximum $s$-$t$ flow and bipartite $\boldsymbol{b}$-matching problem, as well as, from understanding the performance of interior-point methods – when applied to these two problems – via the structure of corresponding electrical flows. To highlight these elements, we decompose the proof of our main theorem (Theorem I.1) into three natural parts.

*Reducing Maximum Flow to $\boldsymbol{b}$-Matching:* First, we focus on analyzing the relationship between the maximum $s$-$t$ flow and the (maximum-cardinality) bipartite $\boldsymbol{b}$-matching problem. It is well-known that the latter can be reduced to the former in a simple way. As it turns out, however, one can also go the other way – there is a simple, combinatorial reduction from the maximum flow problem to the task of finding a perfect bipartite $\boldsymbol{b}$-matching.[6]

Before making this precise, let us introduce one definition. Consider a $\boldsymbol{b}$-matching problem instance corresponding to a bipartite graph $G = (V, E)$ with $P$ and $Q$ ($V = P \cup Q$) being two sides of the bipartition. For any edge $e = (p, q) \in E$, let us define the *thickness* $d(e)$ of that edge to be $d(e) := \min\{b_p, b_q\}$. (So, $d(e)$ is an upper bound on the value of $x_e$ in any feasible $\boldsymbol{b}$-matching $\boldsymbol{x}$.) We say that a $\boldsymbol{b}$-matching instance is *balanced* iff

$$\sum_{e \in E} d(e) \leq 4|\boldsymbol{b}|_1. \tag{9}$$

Now, in the full version of the paper [39], we establish the following result.

**Theorem III.1.** *If one can solve a balanced instance of a perfect bipartite $\boldsymbol{b}$-matching problem in a (bipartite) graph with $\bar{n}$ vertices and $\bar{m}$ edges in $T(\bar{n}, \bar{m}, |\boldsymbol{b}|_1)$ time, then one can solve the maximum $s$-$t$ flow problem in a graph $G = (V, E, \boldsymbol{u})$ with $m$ arcs and capacity vector $\boldsymbol{u}$ in $\widetilde{O}((m + T(\Theta(m), 4m, 4|\boldsymbol{u}|_1)) \log |\boldsymbol{u}|_1)$ time.*

This connection between maximum flows and bipartite matchings is useful in two ways. Firstly, it enables us to reduce the main problem we want to solve – the maximum $s$-$t$ flow problem with unit capacities – to a seemingly simpler one: the perfect bipartite $\boldsymbol{b}$-matching problem. Secondly, the fact that this reduction works also for fractional instances provides us with an ability to lift our $\boldsymbol{b}$-matching rounding procedure that we develop later (see Theorem III.3) to the maximum flow setting (see Corollary III.4).

*The Algorithm for Near-Perfect $\boldsymbol{b}$-Matching Problem:* Once the above reduction is established, we can proceed to designing an improved algorithm for the perfect bipartite $\boldsymbol{b}$-matching problem. This algorithm consists of two parts.

The first one – constituting the technical core of our paper – is related to the (fractional) near-perfect bipartite $\boldsymbol{b}$-matching problem, a certain relaxation of the perfect bipartite $\boldsymbol{b}$-matching problem. To describe this task formally, let us call a $\boldsymbol{b}$-matching $\boldsymbol{x}$ *near-perfect* if its size $|\boldsymbol{x}|_1$ is at least $\frac{|\boldsymbol{b}|_1}{2} - \widetilde{O}(m^{\frac{3}{7}})$, i.e., it is within $\widetilde{O}(m^{\frac{3}{7}})$ additive factor of the size of a perfect $\boldsymbol{b}$-matching. Now, given a bipartite graph $G = (P \cup Q, E)$ and demand vector $\boldsymbol{b}$, the *near-perfect $\boldsymbol{b}$-matching problem* is a task of either finding a *near-perfect $\boldsymbol{b}$-matching* in $G$ or concluding that no *perfect $\boldsymbol{b}$-matching* exists in that graph.

Our goal is to design an algorithm that solves this near-perfect $\boldsymbol{b}$-matching problem in $\widetilde{O}(m^{\frac{10}{7}})$ time. To this end, in the full version of the paper [39], we prove the following theorem.

**Theorem III.2.** *Let $G = (V, E)$ with $V = P \cup Q$ be an undirected bipartite graph with $n$ vertices and $m$ edges and let $\boldsymbol{b}$ be a demand vector that corresponds to a balanced $\boldsymbol{b}$-matching instance with $|\boldsymbol{b}|_1 = O(m)$. In $\widetilde{O}(m^{\frac{10}{7}})$ time, one can either find a fractional near-perfect $\boldsymbol{b}$-matching $\boldsymbol{x}$ or conclude that no perfect $\boldsymbol{b}$-matching exists in $G$.*

(Observe that whenever we have an instance of maximum $s$-$t$ flow problem that has $\bar{m}$ arcs and unit capacities, $|\boldsymbol{u}|_1$ is exactly $\bar{m}$. So, if we apply the reduction from Theorem III.1 to that instance then the resulting $\boldsymbol{b}$-matching problem instance will be balanced, have $m \leq 4\bar{m}$ edges, as well as, $|\boldsymbol{b}|_1 \leq 4|\boldsymbol{u}|_1 = 4\bar{m} \leq 2m$. Therefore, we will be able to apply the above Theorem III.2 to it.)

At a very high level, our algorithm for the near-perfect $\boldsymbol{b}$-matching problem is inspired by the way the existing interior-point method path-following algorithms (see, e.g., [54], [53], [7]) can be used to solve it. Basically, our algorithm is an iterative method that starts with some initial, far-from-optimal solution and then gradually improves this maintained solution to near-optimality (pushing it along so-called central path) using appropriate electrical flows as a guidance. We then show how to tie the convergence rate of this process to the structure of the guiding electrical flows. At that point, one can use a simple energy-bounding argument to establish a generic convergence bound that yields an (unsatisfactory) $\widetilde{O}(m^{\frac{3}{2}})$-time algorithm.

To improve upon this bound and deliver the desired $\widetilde{O}(m^{\frac{10}{7}})$-time algorithm, we show how one can appropriately "shape" these guiding electrical flows to make their guidance more effective and thus guarantee faster convergence. Very roughly speaking, it turns out there is a way of changing the maintained solution to make it essentially the same from the point of view of our $\boldsymbol{b}$-matching instance, while dramatically improving the quality of corresponding electrical flows that guide it.

Our way of executing this idea is based on a careful composition of two techniques. One of them corresponds to perturbing, in a certain way, the arcs that are most significantly distorting the structure of electrical flow – this technique can be viewed as a refinement of edge removal technique of Christiano et al. [9]. The other technique corresponds to preconditioning the whole solution by adding additional, auxiliary, arcs to it. These arcs are chosen so to significantly improve the conductance properties of the solution (when viewed as a graph with resistances) while not leading to too significant deformation of the final obtained solution.

[6] One can view this as one possible explanation of why the techniques used in the context of bipartite matchings and maximum flows are so similar.

*Rounding Near-Perfect $\boldsymbol{b}$-Matchings:* Finally, our final step on our way towards solving the perfect $\boldsymbol{b}$-matching problem (and thus the maximum $s$-$t$ flow problem) is related to turning the approximate and fractional answer returned by the algorithm from Theorem III.2 into an exact and integral one. To this end, note that if that algorithm returned a near-perfect $\boldsymbol{b}$-matching that was integral, there would be a standard way to either turn it into a perfect $\boldsymbol{b}$-matching or conclude that no such perfect $\boldsymbol{b}$-matching exists. Namely, one could just use repeated augmenting path computations. It is well-known that given an integral $\boldsymbol{b}$-matching, one can perform, in $O(m)$ time, an augmenting path computation that either results in increasing the size of our $\boldsymbol{b}$-matching by one, or concludes that no further augmentation is possible (and thus no perfect $\boldsymbol{b}$-matching exists). So, as our initial near-perfect $\boldsymbol{b}$-matching has size at least $\frac{|\boldsymbol{b}|_1}{2} - \widetilde{O}(m^{\frac{3}{7}})$, after at most $\widetilde{O}(m^{\frac{3}{7}})$ iterations, i.e., in time $\widetilde{O}(m^{\frac{10}{7}})$, we would get the desired answer.

Unfortunately, the above approach can fail completely once our near-perfect $\boldsymbol{b}$-matching is fractional. This is so, as in this case we do not have any meaningful lowerbound on the progress on the size of the $\boldsymbol{b}$-matching brought by the augmenting path computation.

Therefore, to deal with this issue, we develop the last ingredient of our algorithm: a nearly-linear time procedure that allows one to round fractional $\boldsymbol{b}$-matchings. More precisely, in the full version of the paper [39], building on the work of Goel et al. [20], we establish the following theorem.

**Theorem III.3.** *Let $G = (V, E)$ be an undirected bipartite graph with $m$ edges and let $\boldsymbol{b}$ be a demand vector, if $\boldsymbol{x}$ is a fractional $\boldsymbol{b}$-matching in $G$ of size $k = |\boldsymbol{x}|_1$ then one can find in $\widetilde{O}(m)$ time an integral $\boldsymbol{b}$-matching in $G$ of size $\lfloor k \rfloor$.*

Clearly, if we apply the above rounding method to the fractional near-perfect matching $\boldsymbol{x}$ computed by the algorithm from Theorem III.2, it will give us an integral $\boldsymbol{b}$-matching $\boldsymbol{x}^*$ whose size is still at least $\frac{|\boldsymbol{b}|_1}{2} - \widetilde{O}(m^{\frac{3}{7}})$. So, the augmenting path-based approach we outlined above will let us obtain the desired integral and exact answer to the perfect $\boldsymbol{b}$-matching problem within the desired time bound.

In the light of all the above, we see that combining all the above pieces indeed yields an $\widetilde{O}(m^{\frac{10}{7}})$-time algorithm for the perfect bipartite $\boldsymbol{b}$-matching problem in graphs with $|\boldsymbol{b}|_1 = O(m)$. Now, using the reduction from Theorem III.1, this gives us the analogous algorithm for the maximum $s$-$t$ flow problem in unit-capacity graphs and that, in turn, results in an algorithm for the bipartite matching problem. So, both Theorem I.1 and Theorem I.2 hold.

*Rounding $s$-$t$ Flows:* Finally, we mention the other byproduct of our techniques – the fast rounding procedure for flows. Namely, using the reduction described in Theorem III.1 and the rounding from Theorem III.3 we can obtain a fast rounding procedure not only for fractional $\boldsymbol{b}$-matchings but also for fractional $s$-$t$ flows. Specifically, the proof of the following corollary appears in the full version of the paper [39].

**Corollary III.4.** *Let $G = (V, E, \boldsymbol{u})$ be a directed graph with capacities and let $\boldsymbol{f}$ be some feasible fractional $s$-$t$ flow in $G$ of value $F$. In $\widetilde{O}(m)$ time, we can obtain out of $\boldsymbol{f}$ an integral $s$-$t$ flow $\boldsymbol{f}^*$ of value $\lfloor F \rfloor$ that is feasible in $G$.*

Again, we note that a very similar rounding result was independently obtained by Khanna et al. [28].

## ACKNOWLEDGMENT

We are grateful to Andrew Goldberg, Jonathan Kelner, Lap Chi Lau, Gary Miller, Richard Peng, Seth Pettie, Daniel Spielman, and Shang-Hua Teng for a number of helpful discussions on this topic.

## REFERENCES

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., 1974.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network flows: theory, algorithms, and applications*. Prentice-Hall, 1993.

[3] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, and M. R. Reddy, *Applications of Network Optimization*, ser. Handbooks in Operations Research and Management Science. North-Holland, 1995, vol. 7.

[4] H. Alt, N. Blum, K. Mehlhorn, and M. Paul, "Computing a maximum cardinality matching in a bipartite graph in time $O(n^{1.5}\sqrt{m/\log n})$," *Inf. Process. Lett.*, vol. 37, no. 4, pp. 237–240, 1991.

[5] S. Arora, E. Hazan, and S. Kale, "The multiplicative weights update method: a meta-algorithm and applications," *Theory of Computing*, vol. 8, no. 1, pp. 121–164, 2012.

[6] B. Bollobas, *Modern Graph Theory*. Springer, 1998.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[8] J. R. Bunch and J. E. Hopcroft, "Triangular factorization and inversion by fast matrix multiplication," *Mathematics of Computation*, vol. 28(125), pp. 231–236, 1974.

[9] P. Christiano, J. Kelner, A. Mądry, D. Spielman, and S.-H. Teng, "Electrical flows, Laplacian systems, and faster approximation of maximum flow in undirected graphs," in *STOC'11: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, 2011, pp. 273–281.

[10] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," *Journal of Symbolic Computation*, vol. 9, pp. 251–280, 1990.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. The MIT Press, 2009.

[12] S. I. Daitch and D. A. Spielman, "Faster approximate lossy generalized flow via interior point algorithms," in *STOC'08: Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, 2008, pp. 451–460.

[13] J. Edmonds, "Paths, trees, and flowers," *Canadian Journal of Mathematics*, vol. 17, pp. 449–467, 1965.

[14] J. Egerváry, "Matrixok kombinatorius tulajdonságairól," *Matematikai és Fizikai Lapok*, vol. 38, pp. 16–28, 1931.

[15] P. Elias, A. Feinstein, and C. E. Shannon, "A note on the maximum flow through a network," *IRE Transactions on Information Theory*, vol. 2, 1956.

[16] S. Even and R. E. Tarjan, "Network flow and testing graph connectivity," *SIAM Journal on Computing*, vol. 4, no. 4, pp. 507–518, 1975.

[17] T. Feder and R. Motwani, "Clique partitions, graph compression and speeding-up algorithms," *Journal of Computer and System Sciences*, vol. 51(2), pp. 261–272, 1995.

[18] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian Journal of Mathematics*, vol. 8, pp. 399–404, 1956.

[19] H. N. Gabow and R. E. Tarjan, "Faster scaling algorithms for general graph matching problems," *Journal of the ACM*, vol. 38, no. 4, pp. 815–853, 1991.

[20] A. Goel, M. Kapralov, and S. Khanna, "Perfect matchings in O(n log n) time in regular bipartite graphs," in *STOC'10: Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, 2010, pp. 39–46.

[21] A. V. Goldberg and A. V. Karzanov, "Maximum skew-symmetric flows and matchings," *Mathematical Programming*, vol. 100, no. 3, pp. 537–568, 2004.

[22] A. V. Goldberg and S. Rao, "Beyond the flow decomposition barrier," *Journal of the ACM*, vol. 45, no. 5, pp. 783–797, 1998.

[23] N. J. A. Harvey, "Algebraic algorithms for matching and matroid problems," *SIAM Journal on Computing*, vol. 39, no. 2, pp. 679–702, 2009.

[24] A. J. Hoffman, "Some recent applications of the theory of linear inequalities to extremal combinatorial analysis," in *Proceedings of Symposia in Applied Mathematics*, vol. 10, 1960, pp. 113–127.

[25] J. Hopcroft and R. Karp, "An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs," *SIAM Journal on Computing*, vol. 2, no. 4, pp. 225–231, 1973.

[26] J. A. Kelner, Y. T. Lee, L. Orecchia, and A. Sidford, "An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations," 2013. [Online]. Available: http://arxiv.org/abs/1304.2338

[27] J. A. Kelner, L. Orecchia, A. Sidford, and Z. A. Zhu, "A simple, combinatorial algorithm for solving SDD systems in nearly-linear time," in *STOC'13: Proceedings of the 45th Annual ACM Symposium on the Theory of Computing*, 2013, pp. 911–920.

[28] S. Khanna, T. C. Kwok, and L. C. Lau, 2013, personal communication.

[29] V. King, S. Rao, and R. Tarjan, "A faster deterministic maximum flow algorithm," *Journal of Algorithms*, vol. 17, no. 3, pp. 447–474, 1994.

[30] D. König, "Graphok és matrixok," *Matematikai és Fizikai Lapok*, vol. 38, pp. 116–119, 1931.

[31] I. Koutis, G. L. Miller, and R. Peng, "Approaching optimality for solving SDD systems," in *FOCS'10: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, 2010, pp. 235–244.

[32] ——, "A nearly $m \log n$-time solver for SDD linear systems," in *FOCS'11: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, 2011, pp. 590–598.

[33] L. C. Lau, 2013, personal communication.

[34] Y. T. Lee, S. Rao, and N. Srivastava, "A new approach to computing maximum flows using electrical flows," in *STOC'13: Proceedings of the 45th Annual ACM Symposium on the Theory of Computing*, 2013, pp. 755–764.

[35] L. Lovász, "On determinants, matchings and random algorithms," *Fundamentals of Computation Theory*, vol. 565–574, 1979.

[36] L. Lovász and D. M. Plummer, *Matching Theory*. Elsevier Science, 1986.

[37] A. Mądry, "Fast approximation algorithms for cut-based problems in undirected graphs," in *FOCS'10: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, 2010, pp. 245–254.

[38] ——, "From graphs to matrices, and back: New techniques for graph algorithms," Ph.D. dissertation, Massachusetts Institute of Technology, 2011.

[39] ——, "Navigating central path with electrical flows: from flows to matchings, and back," 2013, available at http://arxiv.org/abs/1307.2205.

[40] S. Micali and V. V. Vazirani, "An $O(\sqrt{|V|} \cdot |E|)$ algoithm for finding maximum matching in general graphs," in *FOCS'80: Proceedings of the 21st Annual IEEE Symposium on Foundations of Computer Science*, 1980, pp. 17–27.

[41] M. Mucha, "Finding maximum matchings via Gaussian elimination," Ph.D. dissertation, University of Warsaw, 2005.

[42] M. Mucha and P. Sankowski, "Maximum matchings via Gaussian elimination," in *FOCS'04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, 2004, pp. 248–255.

[43] J. B. Orlin, "Max flows in o(nm) time, or better," in *STOC'13: Proceedings of the 45th Annual ACM Symposium on the Theory of Computing*, 2013, pp. 765–774.

[44] M. O. Rabin and V. V. Vazirani, "Maximum matchings in general graphs through randomization," *J. Algorithms*, vol. 10, no. 4, pp. 557–567, Dec. 1989.

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

[46] J. Sherman, "Breaking the multicommodity flow barrier for $O(\sqrt{\log n})$-approximations to sparsest cuts," in *FOCS'09: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, 2009, pp. 363–372.

[47] ——, "Nearly maximum flows in nearly linear time," in *FOCS'13: Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science*, 2013.

[48] D. A. Spielman and S.-H. Teng, "Solving sparse, symmetric, diagonally-dominant linear systems in time $O(m^{1.31})$," in *FOCS'03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, 2003.

[49] ——, "Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems," in *STOC'04: Proceedings of the 36th Annual ACM Symposium on the Theory of Computing*, 2004, pp. 81–90.

[50] W. T. Tutte, "The factorization of linear graphs," *Journal of the London Mathematical Society*, vol. 22, pp. 107–111, 1947.

[51] V. Vassilevska Williams, "Multiplying matrices faster than Coppersmith-Winograd," in *STOC'12: Proceedings of the 44th Annual ACM Symposium on the Theory of Computing*, 2012.

[52] V. V. Vazirani, "A theory of alternating paths and blossoms for proving correctness of the $O(\sqrt{|V||E|})$ general graph matching algorithms," *Combinatorica*, vol. 14 (1), pp. 71–109, 1994.

[53] S. J. Wright, *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics, 1997.

[54] Y. Ye, *Interior Point Algorithms: Theory and Analysis*. John Wiley & Sons, 1997.