



Resource Sharing Revisited: Local Weak Duality and Optimal Convergence

Daniel Blankenburg  

Research Institute for Discrete Mathematics, Universität Bonn, Germany

Abstract

We revisit the (block-angular) min-max resource sharing problem, which is a well-known generalization of fractional packing and the maximum concurrent flow problem. It consists of finding an ℓ_∞ -minimal element in a Minkowski sum $\mathcal{X} = \sum_{C \in \mathcal{C}} X_C$ of non-empty closed convex sets $X_C \subseteq \mathbb{R}_{\geq 0}^{\mathcal{R}}$, where \mathcal{C} and \mathcal{R} are finite sets. We assume that an oracle for approximate linear minimization over X_C is given.

We improve on the currently fastest known FPTAS in various ways. A major novelty of our analysis is the concept of local weak duality, which illustrates that the algorithm optimizes (close to) independent parts of the instance separately. Interestingly, this implies that the computed solution is not only approximately ℓ_∞ -minimal, but among such solutions, also its second-highest entry is approximately minimal.

Based on a result by Klein and Young [21], we provide a lower bound of $\Omega\left(\frac{|\mathcal{C}|+|\mathcal{R}|}{\delta^2} \log |\mathcal{R}|\right)$ required oracle calls for a natural class of algorithms. Our FPTAS is optimal within this class – its running time matches the lower bound precisely, and thus improves on the previously best-known running time for the primal as well as the dual problem.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Resource sharing, Dantzig-Wolfe-type algorithms, Decreasing minimization

Digital Object Identifier 10.4230/LIPIcs.ESA.2022.20

Related Version Faster Primal-Dual Convergence for Min-Max Resource Sharing and Stronger Bounds via Local Weak Duality

Full Version: <https://arxiv.org/abs/2111.06820>

Acknowledgements We thank Jens Vygen for many fruitful discussions and Sebastian Pokutta for helpful suggestions.

1 Introduction

1.1 Problem description

Dividing a limited set of *resources* among *customers* is an overarching theme of numerous problems in discrete and continuous optimization. A common formulation of such problems is known as *min-max resource sharing* in the literature. In this work, we consider the *block-angular min-max resource sharing problem* as it was first studied by Grigoriadis and Khachiyan [13]. The problem consists of choosing a feasible resource allocation for every customer, such that the maximum accumulated resource usage is minimized. Formally, it can be described as follows:

There is a finite set of customers \mathcal{C} and a finite set of resources \mathcal{R} . We denote their sizes by $n := |\mathcal{C}|$ and $m := |\mathcal{R}|$. For each customer $C \in \mathcal{C}$, there is a non-empty closed convex set $X_C \subseteq \mathbb{R}_{\geq 0}^m$ of *feasible resource allocations*, also referred to as *block*. The set of *feasible solutions* of the (block-angular) min-max resource sharing problem is given as the Minkowski-sum $\mathcal{X} := \sum_{C \in \mathcal{C}} X_C$.



© Daniel Blankenburg;
licensed under Creative Commons License CC-BY 4.0
30th Annual European Symposium on Algorithms (ESA 2022).

Editors: Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman; Article No. 20; pp. 20:1–20:14
Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Further, we assume that we are given, for some constant $\sigma \geq 1$, a σ -approximate *block-solver*, which is an approximate linear minimization oracle for non-negative price vectors. It is specified by functions $f_C : \mathbb{R}_{\geq 0}^m \rightarrow X_C$ for all $C \in \mathcal{C}$ that satisfy

$$\forall y \in \mathbb{R}_{\geq 0}^m : \quad \langle y, f_C(y) \rangle \leq \sigma \text{opt}_C(y), \quad (1)$$

where $\text{opt}_C(y) := \min_{x \in X_C} \langle y, x \rangle$.

The (block-angular) *min-max resource sharing problem* is to compute resource allocations $x_C \in X_C$ for every customer $C \in \mathcal{C}$, such that $x := \sum_{C \in \mathcal{C}} x_C$ attains

$$\lambda^* := \min_{x \in \mathcal{X}} \|x\|_{\infty}. \quad (2)$$

We abbreviate this problem as *resource sharing problem* in the following. In this work, we consider fully polynomial-time approximation schemes relative to σ . For $\delta > 0$, we construct a solution $x \in \mathcal{X}$ with $\|x\|_{\infty} \leq \sigma(1 + \delta)\lambda^*$ within a number of oracle calls that is polynomial in n, m , and δ^{-1} .

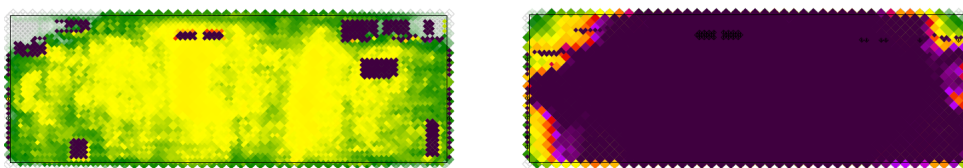
Algorithms that interact with the feasible region only via a linear minimization oracle are known as algorithms of the *Dantzig-Wolfe type*. An iteration consists of choosing a price vector $y \in \mathbb{R}_{\geq 0}^m$ and querying the linear minimization oracle of a customer $C \in \mathcal{C}$ with y . The computed solution is a convex combination of the solutions returned from the oracle. At their core, Dantzig-Wolfe-type algorithms are primal-dual algorithms. In the case of the resource sharing problem, the dual is to find $q \in \Delta_m := \{p \in [0, 1]^m : \|p\|_1 = 1\}$, such that $\min_{x \in \mathcal{X}} \langle q, x \rangle = \max_{p \in \Delta_m} \min_{x \in \mathcal{X}} \langle p, x \rangle$. Strong duality is implied by von Neumann's minimax theorem:

$$\max_{p \in \Delta_m} \min_{x \in \mathcal{X}} \langle p, x \rangle = \min_{x \in \mathcal{X}} \max_{p \in \Delta_m} \langle p, x \rangle = \min_{x \in \mathcal{X}} \|x\|_{\infty} = \lambda^*. \quad (3)$$

The resource sharing formulation can be used to model a large variety of packing problems in combinatorial optimization. Prominent examples include multicommodity and concurrent flow problems, where the linear minimization tasks correspond to shortest path problems, or fractional Steiner tree packing problems, where the linear minimization tasks correspond to minimum weight Steiner tree problems and can be implemented, for example, with a fast 2-approximation.

Using a linear minimization oracle to interact with the feasible region is a standard tool in convex optimization, most often encountered in the form of conditional gradient descent (a.k.a. Frank-Wolfe algorithm [9]), that has regained interest in current research as it leads to surprisingly fast algorithms in practice [17, 19]. Apart from efficiency, utilizing a linear minimization oracle has the advantage that solutions are constructed explicitly as an (often sparse) combination of extreme points. This is highly useful if one is interested in an integral solution to the packing problem, since, in practice, a close to optimal integral solution can often be recovered by applying e.g. randomized rounding to the sparse fractional solution [28].

Resource sharing has a long history of incremental generalizations and improvements since Shahrokhi and Matula [29] introduced the idea to use a linear optimization oracle with an exponential weight function in the context of multicommodity flows. At their heart, most, if not all, steps in this line of work can be interpreted as variants of the now well-known multiplicative weight update method. Since we also present a variant of that algorithm and improve on the best-known running time in this setting, our work can be regarded as another step in that line. But more than that, we study an aspect that – to the best of our knowledge – is absent from any treatment of multiplicative-weight-based algorithms even in special cases such as multicommodity flow. We are interested in the stability of such algorithms and in



(a) High congestion only in local hotspots. (b) High congestion in almost every part.

■ **Figure 1** Comparison of two fractional routing results. Congestion of global routing tiles is visualized by color (lowest congestion in green, highest in violet).

deriving guarantees that go beyond the min-max objective. For example, one might alter a given instance \mathcal{X} by adding a completely independent resource and consider $\mathcal{X} \times \{\Lambda\}$ with $\Lambda \gg \min_{x \in \mathcal{X}} \|x\|_\infty$. In this case, ℓ_∞ -guarantees of existing algorithms do not provide any insight into the quality of their computed solution on the perturbed instance when restricted to \mathcal{X} . We introduce techniques that allow a comprehensive treatment of such situations.

To provide more motivation for this pursuit, we consider the example of global routing in VLSI design, where a resource sharing formulation has proven successful in theory and practice [12, 16, 24]. Highly simplified, the problem consists of a Steiner tree packing problem in a grid graph. One seeks to find a collection of (fractional) Steiner trees that minimize the maximum edge overload. Figure 1 depicts two fractional routing results on an industrial microprocessor. The maximum edge congestion is the same in both cases, so both solutions have the same objective value w.r.t. the resource sharing problem. As indicated in Figure 1a, however, the maximum congestion might be caused by local effects that are beyond reach for global optimization. A practicable algorithm must be resilient to such local hotspots and produce solutions that are close to optimal on the set of remaining resources. The solution in Figure 1a is clearly to be preferred over that in Figure 1b. In practice, the ideal outcome might be a solution $x \in \mathcal{X}$ that minimizes the maximal entry, and among such solutions, it minimizes the second-highest entry, and among those, it minimizes the third-highest, and so on. This concept is known as *decreasingly minimal* [6].

► **Definition 1.** Let $x \in \mathbb{R}^m$. We denote by x_\downarrow the vector x with entries sorted in decreasing order. We introduce the decreasing preorder, by defining for $x, y \in \mathbb{R}^m$, $x \leq_{dec} y$ if either $x_\downarrow = y_\downarrow$ or there exists $k \in \{1, \dots, m\}$ such that $(x_\downarrow)_k < (y_\downarrow)_k$ and $(x_\downarrow)_j = (y_\downarrow)_j$ for all $j < k$. Given a set $X \subseteq \mathbb{R}^m$, we say that an element $x \in X$ is decreasingly minimal if $x \leq_{dec} y$ holds for all $y \in X$.

1.2 Our contribution

In this work, we present a fully polynomial-time approximation scheme for the primal as well as the dual of the resource sharing problem. Our algorithm is an extension of the algorithm by Müller, Radke, and Vygen [24].

We introduce the novel concept of *local weak duality* (Definition 2) to analyze the *core algorithm* (Algorithm 1) which is a variant of the multiplicative weight update method. Given local weak duality, we can prove stronger bounds on $\max_{r \in S} x_r$ for the computed solution $x \in \mathcal{X}$ and certain subsets $S \subseteq \mathcal{R}$. That provides a theoretical counterpart to the empirical observation that the algorithm optimizes (close to) independent parts of the instance separately. Moreover, it allows concluding that – with an exact block solver – our algorithm always computes a solution that is approximately decreasingly minimal on the two highest entries. Such guarantees were not known even for special cases of the problem and

could, in principle, be transferred to many other settings where the multiplicative weight update method is employed. We also provide a negative result, namely that our algorithm will not compute close to decreasingly minimal solutions on the three highest entries in general.

Furthermore, we present an iterative scaling scheme in conjunction with an amortized running time analysis that results in the currently fastest known constant factor approximation for the resource sharing problem with $\mathcal{O}((n+m) \log m)$ many oracle calls. This is then utilized as an initial scaling technique to design an FPTAS improving on the best-known running time for the resource sharing problem. Also, dual convergence, which follows immediately with our analysis, was not shown in prior work.

Moreover, we discuss the limits of algorithms in this setting. We study a class of natural extensions of the core algorithm and prove, using a result by Klein and Young [21], that any algorithm from this class requires $\Omega(\frac{n+m}{\delta^2} \log m)$ oracle calls to compute a $(1+\delta)$ -approximate solution (for a range of parameters as described in Theorem 13). This matches the running time of our algorithm precisely. As this is independent of the choice of the prices, this proves that – in a certain sense – multiplicative price updates are optimal, and that no warm-start analysis (i.e. reducing the running time by starting with a close to optimal dual solution) of such algorithms is possible.

1.3 Related work

The resource sharing problem originates from the maximum concurrent flow problem. For this special case, Shahrokhi and Matula [29] introduced the idea to use Dantzig-Wolfe-type algorithms with an exponential weight function. Another important special case of this problem is *fractional packing*, which can be described as follows. Given a polyhedron $P \subseteq \mathbb{R}^k$, a matrix $A \in \mathbb{R}^{m \times k}$ that satisfies $Ax \geq 0$ for all $x \in P$, and a vector $b \in \mathbb{R}_{>0}^m$, find an $x \in P$ that satisfies $Ax \leq b$. The width of the instance is defined as $\rho := \max_{x \in P} \max_{i=1, \dots, m} (Ax)_i / b_i$. Plotkin, Shmoys, and Tardos [26] studied this problem in the context of Dantzig-Wolfe-type algorithms. Their results were generalized and improved multiple times [5, 11, 13, 18, 20, 22, 24]. An important idea in this line of work is a step-size technique introduced by Garg and Könemann [11] and extended by Fleischer [5], which is used to design algorithms with width-independent running times.

The general version of the block-angular min-max resource sharing problem, as it is the subject of this work, was studied first by Grigoriadis and Khachiyan [13]. In their formulation, one is given non-empty closed convex blocks B_C for each $C \in \mathcal{C}$, and resource allocation functions $g^{(C)} : B_C \rightarrow \mathbb{R}_{\geq 0}^m$, which are convex and non-negative in each coordinate. Then, one seeks to compute $\min\{\max_{r \in \mathcal{R}} \sum_{C \in \mathcal{C}} g^{(C)}(b_C)_r : b_C \in B_C \forall C \in \mathcal{C}\}$. Their optimization oracle solves $\min_{b_C \in B_C} \langle y, g^{(C)}(b_C) \rangle$ for a given price vector $y \in \mathbb{R}_{\geq 0}^m$. It is easy to see that this formulation fits into our framework by defining X_C as the convex hull of $g^{(C)}(B_C)$. The currently fastest known algorithm for the general case is due to Müller, Radke, and Vygen [24]. They present a sequential algorithm that computes a solution of value at most $\sigma(1+\delta)$ within $\mathcal{O}((n+m) \log m (\delta^{-2} + \log \log m))$ oracle calls. All of the mentioned approaches can be interpreted as variants of the multiplicative weights update method [2].

Klein and Young [21] studied lower bounds on the number of iterations that are required by any Dantzig-Wolfe-type algorithm to compute a $(1+\delta)$ -approximate solution for fractional packing. They provide an asymptotic width-dependent bound of $\Omega\left(\min\left\{\rho \frac{\log m}{\delta^2}, m^{1/2-\gamma}\right\}\right)$ (for any fixed $\gamma \in (0, 1/2)$). This matches known upper bounds precisely for a range of parameters. If the width of the instance is unbounded, it is easy to see that $\Omega(m)$ oracle

■ **Algorithm 1** Core Resource Sharing Algorithm with parameters ϵ, T .

```

1:  $y \leftarrow \mathbb{1} \in \mathbb{R}^m$ 
2: for  $t = 1, \dots, T$  do
3:   for  $C \in \mathcal{C}$  do
4:      $\alpha \leftarrow 0$ 
5:      $s_C^{(t)} \leftarrow 0$ 
6:     while  $\alpha < 1$  do
7:        $b \leftarrow f_C(y)$ 
8:        $\xi \leftarrow \min\{1 - \alpha, 1/\|b\|_\infty\}$ 
9:        $y_r \leftarrow y_r \exp(\epsilon \xi b_r) \quad \forall r \in \mathcal{R}$ 
10:       $s_C^{(t)} \leftarrow s_C^{(t)} + \xi b$ 
11:       $\alpha \leftarrow \alpha + \xi$ 
12:    end while
13:  end for
14:   $s^{(t)} \leftarrow \sum_{C \in \mathcal{C}} s_C^{(t)}$ 
15: end for
16: return  $\frac{1}{T} \sum_{t=1}^T s^{(t)}$ 

```

calls are required to compute a constant-factor approximation [14]. If one is not restricted to algorithms of the Dantzig-Wolfe-type, then it is known how to avoid the δ^{-2} dependence on the running time for the case of fractional packing [1, 3].

Decreasingly minimal solutions to optimization problems appear under many different names in the literature, such as lexicographically optimal [10, 23], egalitarian [4], fair [8, 27] and more recently in a line of work by Frank and Murota – who study the integral case – as decreasingly minimal [6, 7]. We are going to use their notation in the following. It is known how to find such solutions with linear programming techniques [25, 27]. In our case, since \mathcal{X} is a non-empty closed convex subset of $\mathbb{R}_{\geq 0}^m$, it contains a unique decreasingly minimal element [27]. A concept related to decreasing minimization is that of *majorization* [15]. If the set of feasible solutions contains a least majorized element, it is also the decreasingly minimal element and can be extracted as the minimum of non-decreasing separable convex functions [30]. We are not aware of results w.r.t. decreasingly minimal solutions in the context of Dantzig-Wolfe-type algorithms.

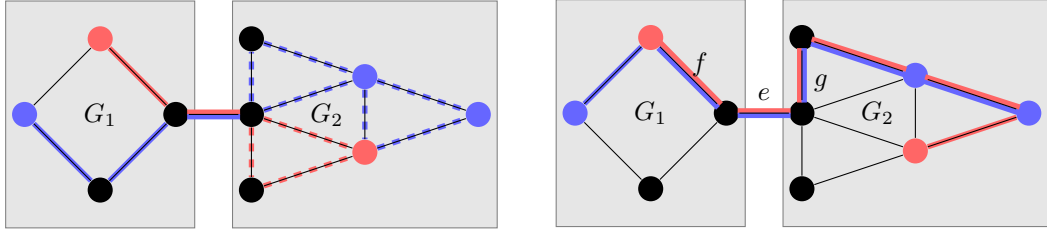
1.4 Outline

The core algorithm (Algorithm 1, [24]) starts with the uniform price vector $y = \mathbb{1} \in \mathbb{R}^m$ and, after every oracle call, it updates the prices $y_r \leftarrow y_r \exp(\epsilon \xi b_r)$, where $\epsilon > 0$ is a fixed parameter of the algorithm. It runs in $T \in \mathbb{N}$ phases (iterations of the outer loop). At the end, the average solution over all phases $x^{(T)} := \frac{1}{T} \sum_{t=1}^T s^{(t)}$ is returned. The restricted step sizes in Line 8 were first proposed by Garg and Könemann [11] in the case of multicommodity flows.

The standard tool to prove primal convergence of algorithms that are based on the multiplicative weight update method is weak duality: For $y \in \mathbb{R}_{\geq 0}^m$, it holds that

$$\sum_{C \in \mathcal{C}} \langle y, f_C(y) \rangle \leq \sigma \min_{x \in \mathcal{X}} \langle y, x \rangle \leq \sigma \|y\|_1 \min_{x \in \mathcal{X}} \|x\|_\infty = \sigma \|y\|_1 \lambda^*. \quad (4)$$

This is sufficient to derive bounds on $\|x^{(T)}\|_\infty$. However, we aim to prove stronger bounds on $\max_{r \in S} x_r^{(T)}$ for certain subsets of resources $S \subseteq \mathcal{R}$. To this end, we introduce the concept of *local weak duality*, which generalizes weak duality. We briefly describe the intuition behind



(a) The decreasingly minimal solution λ with a decomposition $\lambda = \lambda^{(blue)} + \lambda^{(red)}$. Here $\lambda^{(blue)}$ ($\lambda^{(red)}$) assigns value 1 to all edges that are marked with solid blue (red) lines and value $1/2$ to all edges that are marked with dashed blue (red) lines.

(b) Example of a bad oracle. Consider edge prices $y \in \mathbb{R}_{\geq 0}^{E(G)}$, given by $y_e = 2$, $y_f = y_g = 1$, and zero otherwise. Then, a 2-approximation oracle may return the solution above, not satisfying local weak duality on $E(G_1)$ or $E(G_2)$ for any value $\mu < 2$.

■ **Figure 2** An instance of fractional Steiner tree packing in a graph G that consists of two subgraphs G_1 and G_2 which are joined by an edge. The goal is to find fractional Steiner trees that connect the blue, respectively red terminals, such that the maximum edge load is minimized.

this notion. Given $y \in \mathbb{R}_{\geq 0}^m$, one may consider the *local* objective value $\sum_{C \in \mathcal{C}} \sum_{r \in S} y_r f_C(y)_r$ of the oracles on S (e.g. the cost of the paths restricted to a subset of edges in the multicommodity flow case). A local analog to weak duality is given if this objective value can be bounded by $\mu \sum_{r \in S} y_r$ for some $\mu > 0$, independently of y_r for $r \in \mathcal{R} \setminus S$ (the prices on the remaining edges). The following definition includes a different price vector for every customer, which is necessary to deal with the sequential price updates of the core algorithm. Then, the upper bound on the local objective value is defined using the point-wise maximum of these prices.

► **Definition 2.** We say that an instance \mathcal{X} of the resource sharing problem satisfies local weak duality w.r.t. a subset of resources $S \subseteq \mathcal{R}$ and $\mu \geq 0$ if for any collection of non-negative price vectors $(y^{(C)})_{C \in \mathcal{C}} \subseteq \mathbb{R}_{\geq 0}^m$ it holds that

$$\sum_{C \in \mathcal{C}} \sum_{r \in S} y_r^{(C)} f_C(y^{(C)})_r \leq \mu \sum_{r \in S} \max_{C \in \mathcal{C}} y_r^{(C)}. \quad (5)$$

This definition generalizes weak duality, in the sense that every instance satisfies local weak duality w.r.t. $S = \mathcal{R}$ and $\mu = \sigma \lambda^*$.

Let us briefly present an exemplary application. Figure 2 displays an instance of the fractional Steiner tree packing problem with unit capacities. The goal is to find fractional Steiner trees in G that connect the blue, respectively the red terminals such that the maximum edge usage is minimized. We have $\mathcal{C} = \{blue, red\}$ and $\mathcal{R} = E(G)$. The figure also depicts an optimum solution $\lambda = \lambda^{(red)} + \lambda^{(blue)}$ to the resource sharing problem as described in the caption of Figure 2a. In this case, λ is also the decreasingly minimal solution. As the figure indicates, G can be decomposed into two subgraphs G_1 and G_2 that are joined by an edge e . It is clear that e is the bottleneck, meaning that $\lambda_e = \min_{x \in \mathcal{X}} \|x\|_\infty = 2$. However, the (local) maximum usage on G_1 and G_2 is lower, it holds $\max_{e \in E(G_1)} \lambda_e = 1$ and $\max_{e \in E(G_2)} \lambda_e = \frac{1}{2}$. We observe that the Steiner tree problems decompose into two independent subproblems in G_1 and G_2 , i.e. the blue, respectively red terminals in G_1 and G_2 need to be connected to the contained endpoint of e . Let us discuss two examples of how to apply local weak duality to this instance for different oracles.

- (i) The oracle is exact. Thus it also solves the subproblems in G_1 and G_2 exactly. Now consider price vectors $y^{(red)}, y^{(blue)} \in \mathbb{R}_{\geq 0}^{E(G)}$. Since $\lambda^{(blue)}$ and $\lambda^{(red)}$ are feasible solutions for the *blue*, respectively the *red* customer, we get:

$$\sum_{e \in E(G_1)} y_e^{(blue)} f_{blue}(y^{(blue)})_e \leq \sum_{e \in E(G_1)} y_e^{(blue)} \lambda_e^{(blue)} \leq \sum_{e \in E(G_1)} \lambda_e^{(blue)} \max\{y_e^{(blue)}, y_e^{(red)}\}.$$

Applying the analogous inequality to the red customer and using that $\lambda = \lambda^{(blue)} + \lambda^{(red)}$, we can derive

$$\sum_{e \in E(G_1)} y_e^{(blue)} f_{blue}(y^{(blue)})_e + \sum_{e \in E(G_1)} y_e^{(red)} f_{red}(y^{(red)})_e \leq \sum_{e \in E(G_1)} \lambda_e \max\{y_e^{(blue)}, y_e^{(red)}\}.$$

The right hand side can be bounded by $\max_{e \in E(G_1)} \lambda_e \sum_{e \in E(G_1)} \max\{y_e^{(blue)}, y_e^{(red)}\}$. Therefore, local weak duality is satisfied w.r.t. $S = E(G_1)$ and $\mu = \max_{e \in E(G_1)} \lambda_e = 1$. The very same argument can be applied to show that local weak duality is also satisfied w.r.t. $S = E(G_2)$ and $\mu = \max_{e \in E(G_2)} \lambda_e = \frac{1}{2}$.

- (ii) The oracle is a 2-approximation given by a path-decomposition algorithm. In this case, we know that it will solve the subproblems in G_1 exactly (as these are shortest path problems) and the approximation guarantee of 2 also holds for the subinstance in G_2 . Thus, with the same argument as above, local weak duality is satisfied with regard to $S = E(G_1)$ and $\mu = 1$, and with regard to $S = E(G_2)$ and $\mu = 2 \cdot \frac{1}{2} = 1$.

In general, we cannot assume local weak duality with $\mu < 2$ for all 2-approximation oracles as described in Figure 2b for this instance. This illustrates that our notion of local weak duality is not only a property of the convex region that we consider but also a property of the oracle. This instance is an example of a product case, which we briefly discuss in Section 3.

In Section 2.1, we analyze the core algorithm with new techniques. We prove primal-dual convergence and that, under local weak duality, $\max_{r \in S} x_r^{(T)}$ is bounded by μ plus an additive error that is linear in δ . Dual convergence was not shown in [24]. The core algorithm is sufficient to obtain fast convergence in normalized settings, i.e. when λ^* is known up to a constant factor. To derive a FPTAS for the general problem one needs to extend this algorithm by a prior constant factor approximation. In [24] this is done with a scaling/binary search approach. We present a faster constant-factor approximation in Section 2.2. This allows to transfer all results from the normalized to the general setting without increasing the asymptotic number of oracle calls. We summarize this in the following main theorem.

► **Theorem 3 (Main Theorem).** *Let \mathcal{X} be an instance of the resource sharing problem and $\delta \in (0, 1]$. One can compute a primal solution $x \in \mathcal{X}$ and a dual solution $z \in \Delta_m$ satisfying*

$$\|x\|_\infty \leq (1 + \delta)\sigma\lambda^*, \quad \min_{x \in \mathcal{X}} \langle z, x \rangle \geq (1 - \delta) \frac{\lambda^*}{\sigma}, \quad (6)$$

and, for all $S \subseteq \mathcal{R}$, $\mu \geq 0$ such that (5) holds,

$$\max_{r \in S} x_r \leq \mu + \delta \max\{\lambda^*, \mu\}, \quad (7)$$

using $\mathcal{O}\left(\frac{n+m}{\delta^2} \log m\right)$ oracle calls, and further operations taking polynomial time.

In Section 3, we show how to apply this result to product settings and that the computed solution is close to decreasingly minimal on the two highest entries in the following sense.

► **Corollary 4.** *Let $\lambda \in \mathcal{X}$ be decreasingly minimal. If the block solver is exact, i.e. $\sigma = 1$, then the returned solution $x \in \mathcal{X}$ satisfies*

$$(x_{\downarrow})_1 \leq (\lambda_{\downarrow})_1 + \delta\lambda^* \quad \text{and} \quad (x_{\downarrow})_2 \leq (\lambda_{\downarrow})_2 + \delta\lambda^*. \quad (8)$$

This analysis is best possible for the general case. In the full version we show that an analogous version of Corollary 4 does not hold for the three highest entries. Furthermore, it is not possible to reduce the additive constant of $\delta\lambda^*$ to $\delta(\lambda_{\downarrow})_2$ in the second inequality of (8). In a sense, an additive error of $\mathcal{O}(\delta\lambda^*)$ is best possible in our framework, because the core algorithm is known to make an additive error of $\mathcal{O}(\delta)$ and we scale the instance such that $\lambda^* \in [c, C]$ for some constants $0 < c < C$ before applying it.

In Section 4, we study a class of generalizations of the core algorithm. These can be described as standard block-coordinate descent algorithms. This class contains all algorithms that run in a number of phases T , and in each phase $1, \dots, T$ construct a solution for every customer by using *any* choice of prices and the restricted step size rule as in Line 8 of the core algorithm. Our algorithm is optimal within this class in the sense that any such algorithm requires $\Omega(\frac{n+m}{\delta^2} \log m)$ many oracle calls to construct a $(1 + \delta)$ -approximate solution (Theorem 13), even if $\lambda^* = 1$ is known. This follows from a result by Klein and Young [21] and matches the upper bound provided by our algorithm precisely.

2 Proof of the main theorem

2.1 Analysis on normalized instances

First, we prove the main theorem for normalized instances, meaning those for which $\lambda^* \in [c, 1]$ is known for a constant $c > 0$. Later, in Section 2.2, we show how to remove this assumption. We write $x^{(t)} := \frac{1}{t} \sum_{p=1}^t s^{(p)} \in \mathcal{X}$ for the current solution and $y^{(t)}$ for the price vector y after phase t of the core algorithm is completed. Note that $y_r^{(t)} = \exp(\epsilon \sum_{p=1}^t s_r^{(p)}) = \exp(\epsilon t x_r^{(t)})$ holds for all $r \in \mathcal{R}$ and $t = 1, \dots, T$. This can be used to deduce a simple bound for any $S \subseteq \mathcal{R}$:

$$\max_{r \in S} x_r^{(t)} = \max_{r \in S} \frac{1}{\epsilon t} \log y_r^{(t)} \leq \frac{1}{\epsilon t} \log \sum_{r \in S} y_r^{(t)}. \quad (9)$$

Especially, $\|x^{(t)}\|_{\infty} \leq \frac{1}{\epsilon t} \log \|y^{(t)}\|_1$ holds. We denote the dual objective values that correspond to the price vectors $y^{(t)}$ by

$$\Theta_t := \min_{x \in \mathcal{X}} \frac{\langle y^{(t)}, x \rangle}{\|y^{(t)}\|_1} = \frac{\sum_{C \in \mathcal{C}} \text{opt}_C(y^{(t)})}{\|y^{(t)}\|_1}. \quad (10)$$

We provide a bound that relates the number of oracle calls to the price vector $y^{(t)}$. This bound improves (slightly) on the bound that was shown in [24], which was $tn + \frac{m}{\epsilon} \log \|y^{(t)}\|_1$.

► **Lemma 5.** *The number of oracle calls of the core algorithm until termination of phase $t = 1, \dots, T$ is bounded by*

$$tn + \frac{m}{\epsilon} \log \frac{\|y^{(t)}\|_1}{m}. \quad (11)$$

A proof can be found in the full version. The following lemma states a bound on $\sum_{r \in S} y_r^{(t)}$, which implies a bound on $\max_{r \in S} x_r^{(t)}$ according to Inequality (9). In the case $S = \mathcal{R}$, this provides a bound on the primal error and the number of oracle calls. Again, we refer to the full version for a proof of the lemma.

► **Lemma 6** (Upper bound on the price increase). *Let \mathcal{X} be an instance of the resource sharing problem that satisfies local weak duality w.r.t. $S \subseteq \mathcal{R}$ and $\mu > 0$. Let $\eta := \exp(\epsilon) - 1$. Assume that $\eta\mu < 1$. Then for all $t = 0, 1, \dots, T$, it holds that*

$$\sum_{r \in S} y_r^{(t)} \leq |S| \exp\left(\frac{t\eta\mu}{1 - \eta\mu}\right). \quad (12)$$

As pointed out earlier, every instance satisfies local weak duality w.r.t. \mathcal{R} and $\sigma\lambda^*$. So the bound $\|y^{(t)}\|_1 \leq m \exp(t\eta\sigma\lambda^*/(1 - \eta\sigma\lambda^*))$ that was stated in [24] follows. In this case, however, it is possible to insert the definition of the dual values to obtain the primal-dual bound $\|y^{(t)}\|_1 \leq m \exp\left(\frac{\eta\sigma}{1 - \eta\sigma\lambda^*} \sum_{p=1}^t \Theta_p\right)$, which can be used to prove also dual convergence of the average dual solution.

► **Theorem 7** (Bound on the primal and dual error). *Assume $\epsilon \in (0, 1]$, $\eta\sigma\lambda^* < 1$. For every $t = 1, \dots, T$ the primal solution $x^{(t)}$ and the average dual solution $z^{(t)} := \frac{1}{t} \sum_{p=1}^t \frac{y^{(p)}}{\|y^{(p)}\|_1} \in \Delta_m$ satisfy*

$$\|x^{(t)}\|_\infty \leq \frac{\log m}{\epsilon t} + \frac{1 + \epsilon}{1 - \eta\sigma\lambda^*} \sigma\lambda^*, \quad \min_{x \in \mathcal{X}} \langle z^{(t)}, x \rangle \geq \frac{1 - \eta\sigma\lambda^*}{\sigma(1 + \epsilon)} \left(\lambda^* - \frac{\log m}{\epsilon t} \right). \quad (13)$$

Moreover, if \mathcal{X} satisfies local weak duality w.r.t. $S \subseteq \mathcal{R}$ and $\mu > 0$, then

$$\max_{r \in S} x_r^{(t)} \leq \frac{\log |S|}{\epsilon t} + \frac{1 + \epsilon}{1 - \eta\sigma\lambda^*} \mu. \quad (14)$$

For normalized instances, the main theorem follows with a straightforward calculation from Theorem 7 by choosing $\epsilon = \frac{\delta}{8\sigma}$ and $T = \left\lceil \frac{\log m}{2\sigma\epsilon^2} \right\rceil$. Omitted proofs can be found in the full version.

2.2 Constant-factor approximation in $\mathcal{O}((n + m) \log m)$ oracle calls

In the following, we assume $\lambda^* \in [1, \sigma m]$, which can be guaranteed in n oracle calls, by computing for each customer a solution to the uniform price vector $x := \sum_{C \in \mathcal{C}} f_C(\mathbb{1}) \in \mathcal{X}$ and scaling the instance with $\frac{\sigma m}{\|x\|_\infty}$. Our constant-factor approximation, Algorithm 2, is similar to the core algorithm, but it works with adaptive parameters, may discard the solutions of some phases, and restart them. This is done by checking a bound on the sum of the prices, in Line 13, after every oracle call. The algorithm maintains a guess on λ^* , denoted by Λ , which influences the convex coefficient ξ . A violation of the bound indicates that the guess was too low. This leads to a restart of the phase with $\epsilon \leftarrow \epsilon/2$ and $\Lambda \leftarrow 2\Lambda$.

Let $K^* := \lceil \log \lambda^* \rceil$. We denote by K the number of restarts of the algorithm (i.e. times the if-statement in Line 13 is satisfied). Further, $t_1 \leq \dots \leq t_K$ denote the (not necessarily distinct) indices of the phases in which the restarts occur. For a phase t , we write $\epsilon^{(t)}$ for the ϵ -value with which it was completed successfully. As before, we write $x^{(t)} := \frac{1}{t} \sum_{p=1}^t s^{(p)}$ for the current solution after phase t . By construction of the algorithm, we have that $y_r^{(t)} = \exp\left(\sum_{p=1}^t \epsilon^{(p)} s_r^{(p)}\right)$. Note that $\epsilon^{(p)}$ is decreasing for $p = 1, \dots, t$. Therefore, one can bound the maximum usage of the current solution at termination of phase t analogous to previously by $\|x^{(t)}\|_\infty \leq \frac{1}{\epsilon^{(t)} t} \log \|y^{(t)}\|_1$. We only sketch the analysis of the algorithm in the following. Note that after the k 'th restart, $\epsilon = \frac{1}{2^{k4\sigma}}$. So, for larger k , the price updates $y_r \leftarrow \exp(\epsilon \xi b_r)$ get smaller and smaller. Similarly to the proof of Lemma 6, one can show that after K^* restarts, ϵ is small enough to guarantee that the price bound will not be violated again.

■ **Algorithm 2** Constant-factor approximation for the resource sharing problem.

```

1:  $y \leftarrow \mathbb{1} \in \mathbb{R}^m$ 
2:  $\epsilon \leftarrow \frac{1}{4\sigma}$  ▷ Current epsilon for the price update
3:  $\Lambda \leftarrow 1$  ▷ Current guess on  $\lambda^*$ 
4: for  $t = 1, \dots, T := \lceil \log m \rceil$  do
5:    $y^{(t-1)} \leftarrow y$  ▷ Store the last price vector
6:   for  $C \in \mathcal{C}$  do
7:      $\alpha \leftarrow 0, s_C^{(t)} \leftarrow 0$ 
8:     while  $\alpha < 1$  do
9:        $b \leftarrow f_C(y)$ 
10:       $\xi \leftarrow \min\{1 - \alpha, \Lambda / \|b\|_\infty\}$  ▷ Decide the “amount” with which  $b$  is taken
11:       $y_r \leftarrow y_r \exp(\epsilon \xi b_r) \quad \forall r \in \mathcal{R}$ 
12:       $s_C^{(t)} \leftarrow s_C^{(t)} + \xi b, \alpha \leftarrow \alpha + \xi$ 
13:      if  $\|y\|_1 > m \exp(t)$  then ▷ Check price bound
14:         $y \leftarrow y^{(t-1)}$  ▷ Reset the prices to those of the start of this phase
15:         $\epsilon \leftarrow \epsilon/2, \Lambda \leftarrow 2\Lambda$  ▷ Reduce  $\epsilon$ , increase the guess on  $\lambda^*$ 
16:        go to 6 ▷ Restart phase, solutions from this phase are discarded
17:      end if
18:    end while
19:  end for
20:   $s^{(t)} \leftarrow \sum_{C \in \mathcal{C}} s_C^{(t)}$ 
21: end for
22: return  $\frac{1}{T} \sum_{t=1}^T s^{(t)}$ 

```

► **Lemma 8.** *The total number of restarts K is bounded by K^* .*

This means that the number of restarts is bounded by $\lceil \log \lambda^* \rceil \leq \lceil \log \sigma m \rceil \in \mathcal{O}(\log m)$. Analogously to Lemma 5, one can deduce the following bound on the number of oracle calls, which depends on the indices of the phases that are restarted.

► **Lemma 9.** *There is a constant $c_1 > 0$, such that the number of oracle calls is bounded by $Tn + c_1 m \left(T + \sum_{i=1}^K t_i \right)$.*

To use this bound one exploits that (due to the price bound in Line 13)

$$\lambda^* \leq \|x^{(t)}\|_\infty \leq \frac{1}{\epsilon^{(t)} t} \log \|y^{(t)}\|_1 \leq \frac{\log m}{\epsilon^{(t)} t} + \frac{1}{\epsilon^{(t)}} \quad (15)$$

holds for every successfully completed phase t . Thus if $\lambda^* \gg 1$, then $\epsilon^{(t)}$ must be small already for low values of t . This implies that many restarts occurred in the early phases. Indeed, we provide upper bounds on the t_i , which allow estimating $\sum_{i=1}^K t_i$ by a geometric series resulting in $\sum_{i=1}^K t_i \in \mathcal{O}(\log m)$.

► **Lemma 10.** *There are constants $c_2, c_3 > 0$, such that $t_i \leq c_2 + 2^{i-K^*} c_3 \log m$ holds $\forall i = 1, \dots, K$. Thus the number of oracle calls is in $\mathcal{O}((n+m) \log m)$.*

Lemma 8 states that phase T is completed with $\epsilon^{(T)} \geq \frac{1}{2^{K^*} 4\sigma} \geq \frac{1}{\lambda^* 8\sigma}$. Inserting this into Inequality (15) shows that $\|x^{(T)}\|_\infty \leq 16\sigma\lambda^*$. The analysis is summarized in the following theorem.

► **Theorem 11.** *Algorithm 2 computes a solution $x^{(T)} \in \mathcal{X}$ to the resource sharing problem that satisfies $\|x^{(T)}\|_\infty \leq 16\sigma\lambda^*$ using $\mathcal{O}((n+m) \log m)$ oracle calls.*

Combining Algorithm 2 with the core algorithm proves the main theorem. Omitted proofs from this section can be found in the full version.

3 Decreasing Minimality on the two highest entries

In this section, we discuss the implications of local weak duality. In particular, we prove that our algorithm computes solutions that are close to decreasingly minimal on the two largest entries. Before that, let us consider a simple application, which is the *product case*. Assume that the instance of the resource sharing problem consists of multiple independent parts, that is, there exists a (perhaps unknown) partition of the resources $\mathcal{R} = \mathcal{R}_1 \cup \dots \cup \mathcal{R}_k$ on which the customers act “independently”. More precisely, we assume that each of the convex sets X_C can be decomposed into a product $X_C = X_1^{(C)} \times \dots \times X_k^{(C)}$ where $X_i^{(C)} \subseteq \mathbb{R}_{\geq 0}^{|\mathcal{R}_i|}$ for $i = 1, \dots, k$. Let us write $\mathcal{X}_i := \sum_{C \in \mathcal{C}} X_i^{(C)}$ for $i = 1, \dots, k$. Note that we have $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_k$. Let $C \in \mathcal{C}$ and $y^{(C)} \in \mathbb{R}_{\geq 0}^m$. In this case, \mathcal{X} satisfies local weak duality w.r.t. \mathcal{R}_j and $\min_{x_j \in \mathcal{X}_j} \|x_j\|_\infty$ for every $j = 1, \dots, k$. A proof can be found in the full version. So, indeed, according to the main theorem, Theorem 3, every independent part of the instance is optimized separately, i.e. the primal solution $x^{(T)}$ satisfies $\max_{r \in \mathcal{R}_i} x_r^{(T)} \leq \min_{x_i \in \mathcal{X}_i} \|x_i\|_\infty + \delta \lambda^*$ for all $i = 1, \dots, k$. We already saw a natural example of a product setting in Figure 2. As discussed in the beginning, especially in these product cases it is reasonable to assume local weak duality also for approximate block solvers (with their approximation guarantee). A more surprising observation is that any instance \mathcal{X} with an exact block solver satisfies local weak duality w.r.t. to all but one coordinate and the value of the second-highest entry in the decreasingly minimal solution.

► **Lemma 12.** *Let \mathcal{X} be an instance of the resource sharing problem with $\sigma = 1$. Let $\lambda \in \mathcal{X}$ be the decreasingly minimal element. Assume that $(\lambda_\downarrow)_1 > (\lambda_\downarrow)_2$ holds. Let $r^* \in \mathcal{R}$ be the coordinate of the unique maximum entry, i.e. $\lambda_{r^*} = (\lambda_\downarrow)_1$. Then \mathcal{X} satisfies local weak duality w.r.t. $S := \mathcal{R} \setminus \{r^*\}$ and $\mu := (\lambda_\downarrow)_2$.*

A proof of this statement can be found in the full version. Note that if $(\lambda_\downarrow)_1 = (\lambda_\downarrow)_2$ Corollary 4 is trivial due to primal convergence. In the case $(\lambda_\downarrow)_1 > (\lambda_\downarrow)_2$, Corollary 4 follows due to local weak duality with an application of the main theorem.

One might conjecture that an analogous version to Corollary 4 is valid also for the third-highest entry and so on (meaning that the computed solution $x^{(T)}$ satisfies $(x_\downarrow^{(T)})_3 \leq (\lambda_\downarrow)_3 + \delta \lambda^*$ etc.). This is not the case in general. We provide a counterexample in the full version.

4 Limits of standard Dantzig-Wolfe-type algorithms

In this section, we study a class of algorithms that can be described by the meta-algorithm Algorithm 3. This class contains all algorithms that process the customers in a number T of phases and compute a solution $s_C^{(t)} \in X_C$ for every customer in each phase $t = 1, \dots, T$. We allow to choose any price vector for the oracle queries and allow to return any convex combination at the end of the algorithm. However, we fix the restricted step size rule in Line 7. On the one hand, this class can be interpreted as a natural generalization of the core algorithm. On the other hand, it includes standard conditional gradient methods that work with this restricted step size rule. Using this step-size rule is explained by the fact that otherwise $\|\xi b\|_\infty > 1$ holds. Since ξb is added to the current solution of the given phase and the goal is to find an element with small ℓ_∞ -norm, it is a reasonable approach to restrict the step-size towards elements with large ℓ_∞ -norm. Note that the bound $1/\|b\|_\infty$ in Line 7 can be replaced by $C/\|b\|_\infty$ for any other constant $C > 0$ without affecting the asymptotic lower bound. We prove that any such algorithm needs to use $\Omega(\frac{n+m}{\delta^2} \log m)$ many oracle calls to terminate with a $(1 + \delta)$ -approximate solution.

■ **Algorithm 3** Standard block-coordinate descent with restricted steps.

```

1: for  $t = 1, \dots, T$  do
2:   for  $C \in \mathcal{C}$  do
3:      $\alpha \leftarrow 0, s_C^{(t)} \leftarrow 0$ 
4:     while  $\alpha < 1$  do
5:       Choose  $y \in \mathbb{R}_{\geq 0}^m$ 
6:        $b \leftarrow f_C(y)$ 
7:        $\xi \leftarrow \min\{1 - \alpha, 1/\|b\|_\infty\}$ 
8:        $s_C^{(t)} \leftarrow s_C^{(t)} + \xi b$ 
9:        $\alpha \leftarrow \alpha + \xi$ 
10:    end while
11:   end for
12:    $s^{(t)} \leftarrow \sum_{C \in \mathcal{C}} s_C^{(t)}$ 
13: end for
14: return a convex combination of  $s^{(1)}, \dots, s^{(T)}$ 

```

► **Theorem 13.** *For every $\gamma \in (0, 1/2)$ there exist constants $K_\gamma, \tau_\gamma > 0$, such that for every $m > K_\gamma$, $n \in \mathbb{N}$, there exists an instance of the resource sharing problem with $n+2$ customers, $2m$ resources and $\lambda^* = 1$, such that for any $\delta \in (0, 1/10)$, any version of Algorithm 3 requires*

$$\tau_\gamma(n+m) \min \left\{ \frac{\log m}{\delta^2}, m^{1/2-\gamma} \right\} \quad (16)$$

oracle calls to compute a $(1+\delta)$ -approximate solution.

For a proof we refer to the full version.

5 Conclusion

In this work, we have presented an FPTAS for the primal and the dual of the resource sharing problem and improved on the best-known running time in terms of number of oracle calls.

We were able to show that our algorithm has the natural property to optimize (close to) independent parts of the instance separately by introducing the notion of local weak duality. This implied that our algorithm computes solutions that are close to decreasingly minimal on the two largest entries. Local weak duality provides a theoretical understanding of the empirically observed resilience to local effects of multiplicative-weight-based algorithms. Extending the algorithm to achieve (approximate) decreasing minimality on the third-highest entry and beyond is subject to future work.

In the last section, we have shown that further improvements, if possible, require different types of algorithms. A mere change of the price update rule is not enough to achieve faster convergence. This also implied that no warm-start analysis of any version of Algorithm 3 is possible.

References

- 1 Zeyuan Allen-Zhu and Lorenzo Orecchia. Nearly linear-time packing and covering lp solvers: Achieving width-independence and -convergence. *Mathematical Programming*, 175, February 2018. doi:10.1007/s10107-018-1244-x.

- 2 Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012. doi:10.4086/toc.2012.v008a006.
- 3 Daniel Bienstock and Garud Iyengar. Approximating fractional packings and coverings in $o(1/\epsilon)$ iterations. *SIAM J. Comput.*, 35:825–854, 2006.
- 4 Bhaskar Dutta and Debraj Ray. A concept of egalitarianism under participation constraints. *Econometrica*, 57:615–35, February 1989. doi:10.2307/1911055.
- 5 Lisa K. Fleischer. Approximating fractional multicommodity flow independent of the number of commodities. *SIAM Journal on Discrete Mathematics*, 13(4):505–520, 2000. doi:10.1137/S0895480199355754.
- 6 András Frank and Kazuo Murota. Discrete decreasing minimization, part i: Base-polyhedra with applications in network optimization, 2019. arXiv:1808.07600.
- 7 András Frank and Kazuo Murota. Discrete decreasing minimization, part ii: Views from discrete convex analysis, 2020. arXiv:1808.08477.
- 8 András Frank and Kazuo Murota. Fair integral flows, 2020. arXiv:1907.02673.
- 9 Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956. doi:10.1002/nav.3800030109.
- 10 Satoru Fujishige. Lexicographically optimal base of a polymatroid with respect to a weight vector. *Mathematics of Operations Research*, 5(2):186–196, 1980. URL: <http://www.jstor.org/stable/3689149>.
- 11 Naveen Garg and Jochen Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM J. Comput.*, 37:630–652, January 2007. doi:10.1109/SFCS.1998.743463.
- 12 Michael Gester, Dirk Müller, Tim Nieberg, Christian Panten, Christian Schulte, and Jens Vygen. Bonnrout: Algorithms and data structures for fast and good vlsi routing. *ACM Trans. Des. Autom. Electron. Syst.*, 18(2), April 2013. doi:10.1145/2442087.2442103.
- 13 M. Grigoriadis and L. Khachiyan. Fast approximation schemes for convex programs with many blocks and coupling constraints. *SIAM J. Optim.*, 4:86–107, 1994.
- 14 Michael D. Grigoriadis and Leonid G. Khachiyan. Coordination complexity of parallel price-directive decomposition. *Mathematics of Operations Research*, 21(2):321–340, 1996. doi:10.1287/moor.21.2.321.
- 15 G. H. Hardy, George Polya, and J. E. Littlewood. *Inequalities*. The University press Cambridge [Eng.], 1934.
- 16 Stephan Held, Dirk Müller, Daniel Rotter, Rudolf Scheifele, Vera Traub, and Jens Vygen. Global routing with timing constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(2):406–419, 2018. doi:10.1109/TCAD.2017.2697964.
- 17 Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 427–435, Atlanta, Georgia, USA, 17–19 June 2013. PMLR. URL: <http://proceedings.mlr.press/v28/jaggi13.html>.
- 18 Klaus Jansen and Hu Zhang. Approximation algorithms for general packing problems and their application to the multicast congestion problem. *Mathematical Programming*, 114:183–206, 2008.
- 19 Thomas Kerdreux. *Accelerating conditional gradient methods*. PhD thesis, Université Paris sciences et lettres, June 2020.
- 20 Rohit Khandekar. *Lagrangian relaxation based algorithms for convex programming problems*. PhD thesis, Indian Institute of Technology Delhi, 2004.
- 21 Philip Klein and Neal E. Young. On the number of iterations for dantzig–wolfe optimization and packing-covering approximation algorithms. *SIAM Journal on Computing*, 44(4):1154–1172, 2015. doi:10.1137/12087222X.

- 22 Christos Koufogiannakis and Neal E. Young. A nearly linear-time ptas for explicit fractional packing and covering linear programs. *Algorithmica*, 70(4):648–674, 2014. Journal version of [2007]. doi:10.1007/s00453-013-9771-6.
- 23 Nimrod Megiddo. A good algorithm for lexicographically optimal flows in multi-terminal networks. *Bulletin of the American Mathematical Society*, 83:407–409, 1977.
- 24 Dirk Müller, Klaus Radke, and Jens Vygen. Faster min–max resource sharing in theory and practice. *Mathematical Programming Computation*, 3:1–35, 2011.
- 25 Dritan Nace and James Orlin. Lexicographically minimum and maximum load linear programming problems. *Operations Research*, 55:182–187, February 2007. doi:10.1287/opre.1060.0341.
- 26 Serge Plotkin, David Shmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20, January 2001. doi:10.1109/SFCS.1991.185411.
- 27 Bozidar Radunovic and Jean-Yves Le Boudec. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Transactions on Networking*, 15(5):1073–1083, 2007. doi:10.1109/TNET.2007.896231.
- 28 Prabhakar Raghavan and Clark D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, December 1987. doi:10.1007/BF02579324.
- 29 Farhad Shahrokhi and David W. Matula. The maximum concurrent flow problem. *J. ACM*, 37(2):318–334, April 1990. doi:10.1145/77600.77620.
- 30 Arie Tamir. Least majorized elements and generalized polymatroids. *Mathematics of Operations Research*, 20(3):583–589, 1995. doi:10.1287/moor.20.3.583.