DIMACS Technical Report 94-19
April 1994

COORDINATION COMPLEXITY OF PARALLEL PRICE-DIRECTIVE DECOMPOSITION

by

Michael D. Grigoriadis[1] and Leonid G. Khachiyan[1]
Department of Computer Science
Rutgers University
New Brunswick, NJ 08903

* Research supported by the National Science Foundation under grant CCR-9208539.
[1] Permanent member of DIMACS.

DIMACS is a cooperative project of Rutgers University, Princeton University, AT&T Bell Laboratories, and Bellcore.

DIMACS is an NSF Science and Technology Center funded under ContractSTC-88-09648; and also receives support from the New Jersey Commission on Science and Technology.

ABSTRACT

Parallel price directive decomposition (PDD) methods for the approximate solution of block-angular convex resource sharing problems are considered. This general model in structured optimization consists of K nonempty disjoint compact sets called "blocks" and M nonnegative-valued convex "coupling" inequalities. It has a number of applications in combinatorial optimization, network flows, scheduling, communication networks, engineering design, and finance.

This paper studies the coordination complexity of PDD approximation methods, i.e., the number of iterations required to solve the general resource sharing problem to a fixed relative accuracy, as a function of K and M. First, a simple PDD method based on the classical logarithmic potential function is presented and analyzed. For a fixed accuracy, its coordination complexity is shown to be $O(M \ln M)$, which is within a logarithmic factor of the best possible bound for any PDD method that works with the original blocks. An important property of the logarithmic-potential PDD method is that its coordination complexity depends neither on the "width" nor on the dimension of the blocks. Second, polylogarithmically-matching upper and lower coordination complexity bounds are presented for a wider class of PDD methods in which each block can be partially restricted by the coupling constraints. The upper bound for this class of PDD is obtained by a combined method, which uses either the logarithmic or the exponential potential function, depending on the number of coupling constraints per block.

KEY WORDS. Approximation algorithm, block angular programs, coordination complexity, exponential potential function, fully polynomial time approximation scheme, Lagrangian relaxation, logarithmic potential function, parallel computation, price directive decomposition, structured optimization

1. Introduction.

In this paper we study *price directive decomposition* (PDD) approximation methods for solving general resource sharing problems having the *block angular* structure. This structure consists of K nonempty disjoint convex compact sets B^k, $k = 1, \ldots, K$, called *blocks*, and a set of M *coupling* or *resource* constraints. The resource sharing problem is to compute a block-feasible solution that uses the smallest scalar multiple of a given M-vector of resources. There are many applications of this general model and of its special cases, such as in combinatorial optimization, network flows, scheduling, communication networks, engineering design, and finance. The methods we shall consider are based on logarithmic or exponential potential function reduction techniques, which can be implemented in parallel.

1.2. The problem. We consider the approximate solution of convex block-angular resource-sharing problems of the form

$$\lambda^* = \min \left\{ \lambda \;\middle|\; \begin{array}{rl} \displaystyle\sum_{k=1}^{K} f_m^k(x^k) \le \lambda, & m = 1, \ldots, M \\ x^k \in B^k, & k = 1, \ldots, K \end{array} \right\},$$

where $f_m^k : B^k \to \mathbb{R}_+$ are nonnegative continuous convex functions of the *block variable* x^k in the nonempty convex compact block B^k for each k. We shall use the following notation: $x = (x^1, \ldots, x^K)$, $B = B^1 \times \ldots \times B^K$, $f^k = (f_1^k, \ldots, f_M^k) : \; B^k \to \mathbb{R}_+^M$, $f = \sum_{k=1}^K f^k : \; B \to \mathbb{R}^M$, and e for the vector of all ones. We shall assume throughout that $M \ge 2$.

With this notation, the above resource sharing problem can be written as

$$\lambda^* = \lambda(x^*) = \min_{x \in B} \{\lambda(x) = \max\{f_1(x), \ldots, f_M(x)\}\},$$

or simply written as follows:

$$(\mathcal{P}) \qquad \lambda^* = \min\{\lambda \mid f(x) \le \lambda e, \; x \in B\}.$$

We shall compute an *ε-approximate solution* of $(\mathcal{P})$ for a given *relative tolerance* $\varepsilon \in (0, 1]$, that is, compute an $x \in B$ such that

$$f(x) \le (1 + \varepsilon)\lambda^* e,$$

1.2. Price directive decomposition (PDD). Suppose that a point $x_0 \in B$ is known and let $\lambda(x_0) \le \mu \le +\infty$ be a fixed scalar. Then, by the nonnegativity of the functions f^k, any solution x of $(\mathcal{P})$ also belongs to $B_\mu = B_\mu^1 \times \ldots \times B_\mu^K$, where

$$(1.1) \qquad B_\mu^k = \{ \; x^k \in B^k \mid f^k(x^k) \le \mu e \; \}, \qquad k = 1, \ldots, K.$$

From von Neumann's saddlepoint theorem (see, e.g., Rockafellar [27], p. 393),

$$\lambda^* = \min_{x \in B_\mu} \max_{p \in P} L(x,p) = \max_{p \in P} \min_{x \in B_\mu} L(x,p),$$

where $P = \{ p \in \mathbb{R}^M \mid e^T x = 1,\ p \geq 0 \}$ is the unit simplex, and $L(x,p) = \sum_{k=1}^K L^k(x^k,p) = \sum_{k=1}^K p^T f^k(x^k) = p^T f(x)$ is the Lagrangian of $(\mathcal{P})$. An immediate consequence of this theorem is the familiar duality relation:

$$(\mathcal{D}) \qquad \lambda^* = \max \left\{ g_\mu(p) = \sum_{k=1}^K g_\mu^k(p) \;\middle|\; p \in P \right\},$$

where

$$(\mathcal{P}^k) \qquad g_\mu^k(p) = \min \left\{ L^k(x^k,p) = p^T f^k(x^k) \mid x^k \in B_\mu^k \right\}.$$

Thus, for a fixed scalar μ and a fixed *price vector* $p \in P$ for the coupling constraints, the computation of $g_\mu(p)$ decomposes into K independent *block problems* $(\mathcal{P}^k)$. This observation forms the basis of the well-known class of *Lagrangian* or *price directive* decomposition (PDD) methods (see e.g. [21]). Such methods maintain $x \in B$, $p \in P$, and the scalar $\lambda(x) \leq \mu \leq +\infty$, and repeat the following three steps at each iteration $0, 1, \ldots$, until an appropriate stopping criterion is satisfied:

Step 1 (Compute prices). Use a deterministic or randomized procedure to compute a price vector p and a scalar μ from the information $I = I^0, I^1, \ldots, I^r = \{\hat{x}^{kr}, p^r, \mu^r\}$ accumulated for all iterations $0, 1, \ldots, r$ so far.

Step 2 (Block optimization). Use a "block solver" to compute an optimal solution $\hat{x}^k \in B_\mu^k$ of each block problem $(\mathcal{P}^k)$.

Step 3 (New iterate). Replace the current approximate solution $x^k \in B^k$ by a new convex combination of the previous solutions x^{kr} kept on record.

This general scheme can be implemented in various ways. The block solver can clearly process the problems $(\mathcal{P}^k)$ in parallel. We shall refer to the task of computing the prices p, the scalar μ, and the manner in which the convex combination is to be formed for each block in Step 3 as a *coordination round*, performed by a "coordinator". In most instances, the coordinator can be implemented in parallel as well. We shall use the term *coordination complexity* to mean the total number of iterations performed by a PDD method.

It is important to note that the way the coordinator assigns values for μ at each round affects the constitution of the block problems $(\mathcal{P}^k)$. PDD methods have mostly been studied for coordinators who always assign $\mu = +\infty$ and hence define the $(\mathcal{P}^k)$ over the original block constraints B_k. To distinguish this class of *unrestricted* PDD methods from the more general case whose coordinator can vary μ at will in the interval $[\lambda(x), +\infty]$ at any iterate x, we shall refer to the latter class as *restricted* PDD.

1.3. Background. Price directive decomposition has been studied for almost four decades. We shall mention only a few of the many contributions in this area, mainly those that differ in the way p is computed.

Among the earliest such approaches is Brown's "statistical" method for polyhedral games [5] analyzed by Robinson [28] (also see Shapiro [31] and Khachiyan [18]), the more general scheme of Everett III [7], and several other "two-level" price decomposition methods, including the Dantzig–Wolfe decomposition principle [6]. (See, e.g., Lasdon [21] and the recent book by Ahuja et al. [1] for a number of applications in network optimization and design.) The PDD model also includes various approaches based on nondifferentiable optimization for maximizing $g(p)$, such as space-dilation methods (Shor [32]), the method of inscribed ellipsoids (Tarasov et al. [34], Khachiyan [19]), the method of volumetric centers by Vaidya [36], with improvements for structured problems by Bertsimas and Orlin [3], and cutting plane methods based on analytic centers (see Atkinson and Vaidya [2], Goffin et al. [13], [14]). Coordinators for these methods require all or a substantial part of the history I. For instance, [2], [34], and [36], as well as implementations of [6], maintain the information I for $\Omega(M)$ past iterations.

Also included in this subclass are much simpler, "oblivious" methods that minimize a potential function Φ: $f(x) \to \mathbb{R}$, smooth with respect to f, as an approximation to the objective function of the problem. These methods maintain only the current iterate $x \in B$, determine the vector $p = p(f(x))$ by linearizing Φ with respect to $f(x)$ at x, and select an appropriate step length $\tau \in (0,1]$ to obtain the new iterate $x' = (1-\tau)x + \tau\hat{x}$ in B as to approximately minimize the potential on the blocks. In this respect, they are reminiscent of successive (under)relaxation and Frank–Wolfe [11] type methods. An early example is the method of Fratta et al. [12] — presented in the context of routing in communication networks — which minimizes a *rational* potential function for the multicommodity flow problem,

Exponential potential functions were introduced in optimization as early as 1951 by Motzkin [23] who suggested the use of gradient descent methods for solving linear systems of inequalities. Polyak studied convergence properties and applications of exponential potential functions to minimax optimization problems [25] (also see Bertsekas [4] and Schnabel [30]). Shahrokhi and Matula [29] were the first to obtain a fully polynomial time approximation scheme for the uniform concurrent flow problem (a special linear case of $(\mathcal{P})$) based on such an exponential potential function and unrestricted PDD. Their bounds were substantially improved by Tardos [35], Klein et al. [20], and Leighton et al. [22], and further extended to linear $(\mathcal{P})$ by Plotkin et al. [26]. The best currently known coordination complexity bounds for both linear and nonlinear problems $(\mathcal{P})$ were given by Grigoriadis and Khachiyan [16]. It is important to note that these coordination complexity bounds for exponential-potential PDD were shown for explicitly or implicitly μ-restricted block optimizations. The results of this paper imply that they cannot hold for unrestricted PDD.

Logarithmic potential (or barrier) functions were introduced to linear and convex

optimization by Frisch in the mid 1950's [8], [9]. Leaving aside the well-known application of these functions to polynomial-time interior point methods for linear and convex programming (see e.g., Goldfarb and Todd [15], Nesterov and Nemirovskii [24]), we mention here their recent use within the context of restricted PDD methods by Schultz and Meyer [33]. Their strategy involves a one-sided box constraint as a block restriction and a K-dimensional search in the coordination step that minimizes the potential over all convex combinations of K block directions. A convergence analysis and attractive parallel computational results are reported in [33] for a set of large, practical minimum-cost multicommodity flow problems with $K << M$. On the other hand, an exponential-potential restricted PDD method was found to be even more efficient in solving the same class of problems [17].

1.4. Results. In this paper we shall focus on the basic question: "How does the coordination complexity of the class of all unrestricted or restricted PDD methods for solving $(\mathcal{P})$ grow as a function of K and M for a fixed $\varepsilon \in (0,1]$?" Our results can be summarized as follows.

In Section 2 we examine unrestricted PDD ($\mu = +\infty$). We first develop a simple logarithmic potential-based PDD method and bound its coordination complexity by

$$N_L = O\left(M\left(\varepsilon^{-2}\ln\varepsilon^{-1} + \ln M\right)\right) \tag{1.2}$$

iterations. In addition to these N_L parallel block optimizations, each coordination round of this method requires $O(M \ln\ln(M/\varepsilon))$ arithmetic operations, or $O(\ln M \ln\ln(M/\varepsilon))$ arithmetic operations in parallel on an $M/\ln M$-processor EREW PRAM.

We see that the coordination complexity of such a PDD is $O(M \ln M)$ for a fixed ε. On the other hand, in Section 2.8 we exhibit a simple instance of $(\mathcal{P})$ whose 1–approximate solution by any unrestricted PDD method requires

$$N = \Omega(M)$$

iterations, even with an all-powerful coordinator. This shows that for a fixed ε, the coordination complexity of the logarithmic-potential PDD is optimal up to a $\ln M$ factor. An important aspect of these bounds is that they are independent of the size (diameter, width) of the blocks B_k. And a useful property of the method is that it provides an ε-approximate optimal dual vector p when it halts.

The optimal value of problem $(\mathcal{P})$ can also be computed in $O(M \ln 1/\varepsilon)$ iterations by the inscribed ellipsoid (Tarasov et al. [34]) or by Vaidya's method of volumetric centers [36], and in $O(M \ln^2 1/\varepsilon)$ iterations by the cutting plane method based on analytic centers (Atkinson and Vaidya [2]). The theoretically fastest of these methods, [36], requires $O(\mathcal{M}(M))$ arithmetic operations per coordination round, where $\mathcal{M}(M)$ is the complexity of matrix multiplication for M-order matrices. With the best currently-known bound of $\mathcal{M}(M) = O(M^{2.38})$, and disregarding the large hidden constant factors, its $O(M \cdot \mathcal{M}(M))$ total coordination work is still substantially higher than that for the

logarithmic potential PDD method, which grows quadratically in M, up to logarithmic factors.

In Section 3 we discuss restricted PDD ($\lambda(x) \leq \mu \leq +\infty$). We use an exponential-potential restricted PDD method to solve problem $(\mathcal{P})$ with M coupling constraints and K blocks to a relative accuracy of ε in

$$N_E = O\left(K \ln M \left(\varepsilon^{-2} + \ln M\right)\right) \tag{1.3}$$

iterations [16]; in particular, its coordination complexity is $O(K \ln^2 M)$ for a fixed ε. We use this method for $K \leq M/\ln M$, where each coordination round requires $O(M)$ operations, or $O(\ln M)$ operations in parallel on $M/\ln M$ processors.

Since unrestricted PDD is a special case of restricted PDD — or because the iteration bound of logarithmic-potential PDD can be shown to hold for any $\lambda(x) \leq \mu \leq +\infty$ — the coordination complexity bound N_L in (1.2) applies to all restricted PDD as well. Hence the upper bound of

$$N = \min\{N_L, N_E\} = O\left(\min\{M \ln M, K \ln^2 M\}\right)$$

on the coordination complexity of problem $(\mathcal{P})$ for any fixed ε, which we obtain by combining the logarithmic- and exponential-potential restricted PDD methods.

In Section 3 we also show a lower bound of

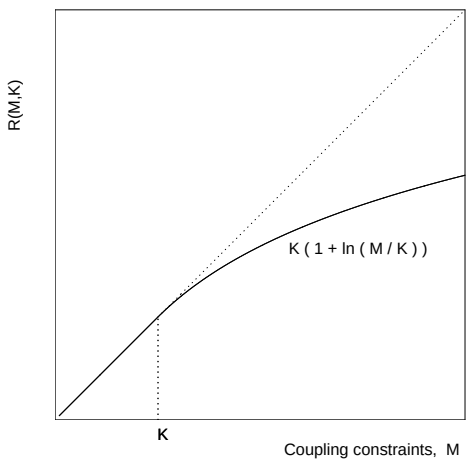
$$N = \Omega(R(M, K))$$

iterations for computing a 1-approximate solution of $(\mathcal{P})$ in the class of all restricted PDD methods, where

$$R(M,K) = \begin{cases} M & \text{if } M \leq K \\ K\left(1 + \ln(M/K)\right) & \text{if } M > K. \end{cases} \tag{1.4}$$

(See Figure 1.) Our upper bound on the coordination complexity of the combined logarithmic-exponential restricted PDD matches the lower bound for restricted PDD to within a factor of $\ln^2 M/(1 + \ln\ln M)$ for all values of M and K, and to within a factor of $\ln M$ for $M \leq K$ or $M >> K$.

Figure 1. Lower bounds $R(M,K)$ for restricted PDD



2. Unrestricted PDD and the logarithmic-potential method.

In Sections 2.1–6, we develop a logarithmic potential function unrestricted PDD method for problem $(\mathcal{P})$ and bound its coordination complexity. The last Section 2.8 gives a simple lower bound on the coordination complexity of unrestricted PDD methods.

2.1. Logarithmic potential and price vector. In stating our unresrticted PDD algorithm and its analysis, we shall work with the standard logarithmic potential function [10],[15] of the form

$$\Phi(\theta, f) = \theta - \frac{t}{M} \sum_{m=1}^{M} \ln\left(\theta - f_m(x)\right), \qquad x \in B, \tag{2.1}$$

where $\theta \in \mathbb{R}_+$ is a variable and t is a fixed positive parameter, specified later in Section 2.2. This potential function has a number of useful properties which we list below.

1) For a fixed $f \in R_+^M$, the function $\Phi(\theta, f)$ is convex in θ and has the *barrier property*: $\Phi(\theta, f) \to +\infty$ for $\theta \to \lambda(f) + 0$ or for $\theta \to +\infty$, where $\lambda(f) = \max\{f_1, \ldots, f_M\}$. Let $\theta(f)$ denote the unique minimizer of the potential on $\theta \in (\lambda(f), +\infty)$ for a fixed f, i.e.,

$$\theta(f) = \operatorname{argmin}\{\Phi(\theta, f) \mid \lambda(f) < \theta < +\infty\}. \tag{2.2}$$

Clearly, $\theta(f)$ can be determined from the first-order optimality condition $\partial\Phi(\theta, f)/\partial\theta = 0$, or equivalently, by solving the equation

$$\frac{t}{M} \sum_{m=1}^{M} \frac{1}{\theta - f_m} = 1. \tag{2.3}$$

Since the left-hand side of (2.3) decreases monotonically with increasing values of θ, the implicit function $\theta(f)$ is the unique root of (2.3) in the interval $(\lambda(f), +\infty)$.

2) The smooth function $\theta : f \to \mathbb{R}_+^1$ approximates the piecewise linear function $\lambda : f \to \mathbb{R}^1$ from above and below as follows:

LEMMA 1. $\lambda(f) + t/M \le \theta(f) \le \lambda(f) + t$.

Proof. The first inequality should be obvious from (2.3). The second inequality follows from

$$1 = \frac{t}{M} \sum_{m=1}^{M} \frac{1}{\theta(f) - f_m} \le \frac{t}{M} \sum_{m=1}^{M} \frac{1}{\theta(f) - \lambda(f)} = \frac{t}{\theta(f) - \lambda(f)}. \qquad \square$$

3) Define $\phi(f) = \Phi(\theta(f), f)$. Then, for small $t > 0$, the function $\theta(f)$ also closely approximates $\phi(f)$ as shown in the next lemma.

LEMMA 2. $\theta(f) - t\ln\theta(f) \;\le\; \phi(f) \;\le\; \theta(f) - t\ln t \quad$ *for any* $f \in \mathbb{R}^M_+$.

Proof. By the nonnegativity of f,

$$\begin{aligned}\theta(f) &= \phi(f) + \frac{t}{M}\sum_{m=1}^{M} \ln\left(\theta(f) - f_m\right) \\ &\le \phi(f) + \frac{t}{M}\sum_{m=1}^{M} \ln\theta(f) \\ &= \phi(f) + t\ln\theta(f),\end{aligned}$$

which gives the first inequality in the lemma. To show the second inequality, observe from (2.3) that

$$1 = \frac{t}{M}\sum_{m=1}^{M}\frac{1}{\theta(f) - f_m} \;\ge\; t\left(\prod_{m=1}^{M}\frac{1}{\theta(f) - f_m}\right)^{1/M},$$

and consequently $0 \ge \ln t - \frac{1}{M}\sum_{m=1}^{M}\ln\left(\theta(f) - f_m\right)$. Multiplying this inequality by t, we obtain $0 \ge t\ln t - \theta(f) + \phi(f)$. □

COROLLARY 1. $\theta(f) \le \left(\phi(f) - t\right)/\left(1 - t\right)$ *for* $t \in (0,1)$.

Proof. Using the inequality $\ln\theta \le \theta - 1$ and Lemma 2, $\theta(f) - t(\theta(f) - 1) \;\le\; \theta(f) - t\ln\theta(f) \;\le\; \phi(f)$. □

4) Define the price vector $p = p(f) \in \mathbb{R}^M$, with elements

$$p_m(f) = \frac{t}{M}\frac{1}{\theta(f) - f_m}, \qquad m = 1, \ldots, M. \tag{2.4}$$

Clearly, $p \ge 0$ and $e^T p = 1$ from (2.3). Furthermore,

$$\begin{aligned}p^T f = \frac{t}{M}\sum_{m=1}^{M}\frac{f_m}{\theta(f) - f_m} &= \frac{t}{M}\sum_{m=1}^{M}\left(-1 + \frac{\theta(f)}{\theta(f) - f_m}\right) \\ &= -t + \theta(f)\frac{t}{M}\sum_{m=1}^{M}\frac{1}{\theta(f) - f_m} = -t + \theta(f)e^T p. \\ &= -t + \theta(f).\end{aligned} \tag{2.5}$$

For convenience we shall henceforth write $\lambda(x)$, $\theta(x)$, and $\phi(x)$ for $\lambda(f(x))$, $\theta(f(x))$, and $\phi(f(x))$, respectively.

2.2. Algorithm $\mathcal{A}_L$ for solving $(\mathcal{P})$ to a given absolute accuracy. In this section we state a simple logarithmic potential function descent algorithm that computes a solution x of the resource sharing problem $(\mathcal{P})$ to within a given *absolute* accuracy $\sigma \in (0, 1/3]$. The algorithm requires that a point $x_0 \in B$ with $\lambda(x_0) \le 1$ be given. Since

the algorithm is an unrestricted PDD method with $\mu = +\infty$, we assume with no loss of generality a single block B. Later we shall use this algorithm as a subroutine to solve problem $(\mathcal{P})$ to a given *relative* accuracy $\epsilon > 0$.

Algorithm $\mathcal{A}_L(f, x, \sigma)$:
Step 0. (Initialization.) Set $x = x_0$ and select

$$t = \frac{3\sigma}{7}, \quad \text{and} \quad \tau = \frac{1}{M}\frac{4t^2}{25 + 10t}. \tag{2.6}$$

Step 1. (Compute prices.) Set $f = f(x) \in \mathbb{R}_+^M$ and compute the price vector $p = p(f) \in \mathbb{R}_+^M$ by (2.4).
Step 2. (Block optimization.) Compute an $\hat{x} \in B$ such that

$$p^T f(\hat{x}) \;\leq\; \left(1 + \frac{\sigma}{7}\right) \min \{\; p^T f(y) \mid y \in B \;\}. \tag{2.7}$$

Step 3. (Stopping criterion.) If

$$p^T \hat{f} \geq \theta(f) - 2t, \tag{2.8}$$

where $\hat{f} \equiv f(\hat{x}) \in \mathbb{R}_+^M$, then report x and stop.
Step 4. (New iterate.) Set[1]

$$x' = (1 - \tau)x + \tau\hat{x}.$$

Replace x with x' and go to Step 1 to start a new iteration. □

We shall now prove Algorithm $\mathcal{A}_L$ correct and state a bound on its coordination complexity for for the fixed values of the parameter t and step length τ given by (2.6).

THEOREM 1. *For any initial point $x_0 \in B$ such that $\lambda(x_0) \leq 1$ and for any $\sigma \in (0, 1/3]$,* Algorithm $\mathcal{A}_L$ *halts in*

$$N = O\left(M\left(\frac{\lambda(x_0) - \lambda^*}{\sigma^3} \;+\; \frac{\ln 1/\sigma}{\sigma^2}\right)\right) \tag{2.9}$$

iterations with an $x \in B$ such that

$$\lambda(x) \leq \lambda^* + \sigma. \tag{2.10}$$

[1] The step length τ can be determined in practice by a line search minimizing Φ for fixed θ, but our subsequent analysis assumes the constant τ specified in Step 1.

Proof. We first show the correctness of the stopping criterion (2.8) in Step 3 of Algorithm $\mathcal{A}_L$. Let x^* be an optimal solution of problem $(\mathcal{P})$. By the solution of the block problem (2.7) in Step 2,

$$\begin{aligned} p^T\hat{f} \equiv p^T f(\hat{x}) &\le (1+\sigma/7)\min\{p^T f(y) \mid y \in B\} \le (1+\sigma/7)\, p^T f(x^*) \\ &\le (1+\sigma/7)\max_m f_m(x^*) e^T p = (1+\sigma/7)\,\lambda(x^*) \\ &\le \lambda^* + \sigma/7. \end{aligned} \tag{2.11}$$

If the current iterate $x \in B$ satisfies the stopping criterion (2.8), then (2.11) implies that $\sigma/7 + \lambda^* \ge \theta(x) - 2t$. But $\theta(x) > \lambda(x)$ by Lemma 1, and therefore

$$\lambda(x) < \lambda^* + \sigma/7 + 2t = \lambda^* + \sigma,$$

which proves (2.10).

In order to prove the bound on the number of iterations of Algorithm $\mathcal{A}_L$ we need the next two lemmas.

LEMMA 3. *Suppose that $\phi(x) \le \phi(x_0)$ for an iterate x of* Algorithm $\mathcal{A}_L$. *Then,*

$$p^T\left(f + \hat{f}\right) \le 5/2, \tag{2.12}$$

where $p = p(x)$, $f = f(x)$, $\phi(x) = \Phi(\theta, f)$, and $\hat{f} = f(\hat{x})$.

Proof. First, $p^T\hat{f} \le \lambda^* + \sigma/7 \le \lambda(x_0) + \sigma/7 \le 1 + \sigma/7 \le 22/21$ by (2.11) and by our assumption $\lambda(x_0) \le 1$. So, it suffices to show that $p^T f \le 5/2 - 22/21 = 61/42$, or equivalently (by (2.5)) show that

$$\theta(x) - t \le 61/42. \tag{2.13}$$

Since $\phi(x) \le \phi(x_0)$ by assumption, Corollary 1 gives

$$\theta(x) \le \frac{\phi(x) - t}{1-t} \le \frac{\phi(x_0) - t}{1-t}\ .$$

On the other hand, Lemma 2 gives

$$\phi(x_0) \le \theta(f(x_0)) - t\ln t, \tag{2.14}$$

and since $\theta(x_0) \le \lambda(x_0) + t$ by Lemma 1,

$$\theta(x) \le \frac{\lambda(x_0) - t\ln t}{1-t} \le \frac{1 - t\ln t}{1-t}.$$

Therefore,

$$\theta(x) - t \le 1 + \frac{t^2 - t\ln t}{1-t}\ .$$

But $(t^2 - t\ln t)\,/\,(1-t)$ is a monotonically increasing function of $t \in (0,1)$ and Algorithm $\mathcal{A}_L$ uses $t = 3\sigma/7 \le 1/7$. Thus,

$$\theta(x) - t \le 1 + \frac{(1/7)^2 - (1/7)\ln 1/7}{1 - 1/7} = 1.3\ldots \le 61/42 = 1.4\ldots. \qquad \square$$

LEMMA 4. *For any two consecutive iterates $x, x' \in B$ of* Algorithm $\mathcal{A}_L$,

$$\phi(x') \le \phi(x) - \frac{t^3}{13M}\ . \tag{2.15}$$

Proof. Denote $f = f(x)$ and $\hat{f} = f(\hat{x})$ as before, and let $f' = f(x')$ and $\lambda' = \lambda(f')$. Then,

$$\begin{aligned}
\phi(x') \equiv \Phi\left(\theta(f'), f'\right) &= \min_{\lambda' < \theta' < +\infty} \Phi(\theta', f') = \min_{0 < \theta' < +\infty} \Phi(\theta', f') \\
&\le \Phi\left(\theta(f), f'\right) \\
&= \Phi\left(\theta(f), f\left((1-\tau)x + \tau\hat{x}\right)\right) \\
&\le \Phi\left(\theta(f), (1-\tau)f + \tau\hat{f}\right),
\end{aligned} \tag{2.16}$$

where the last inequality follows from the fact that $\Phi(\theta, f)$ is a monotonically increasing function with respect to each component f_m of f and from the convexity of each $f_m(\cdot)$. To show the finiteness of the last term above, observe that

$$\begin{aligned}
&\Phi\left(\theta(f), (1-\tau)f + \tau\hat{f}\right) \\
&\quad = \theta(f) - \frac{t}{M}\sum_{m=1}^{M} \ln\left(\theta(f) - f_m + \tau(f_m - \hat{f}_m)\right) \\
&\quad = \theta(f) - \frac{t}{M}\sum_{m=1}^{M} \ln\left(\theta(f) - f_m\right) - \frac{t}{M}\sum_{m=1}^{M} \ln\left(1 + \tau\frac{f_m - \hat{f}_m}{\theta(f) - f_m}\right) \\
&\quad = \phi(x) - \frac{t}{M}\sum_{m=1}^{M} \ln\left(1 + \tau\frac{f_m - \hat{f}_m}{\theta(f) - f_m}\right) \\
&\quad = \phi(x) - \frac{t}{M}\sum_{m=1}^{M} \ln\left(1 + \frac{\tau M}{t} p_m\left(f_m - \hat{f}_m\right)\right),
\end{aligned} \tag{2.17}$$

and for each $m = 1, \ldots, M$,

$$\begin{aligned}
\frac{\tau M}{t} p_m\left(f_m - \hat{f}_m\right) &\ge -\frac{\tau M}{t} p_m \hat{f}_m \ge -\frac{\tau M}{t} p^T \hat{f} \\
&\ge -\left(\lambda^* + \frac{\sigma}{7}\right)\frac{\tau M}{t} \ge -\frac{22}{21}\,\frac{\tau M}{t} = -\frac{22}{21}\,\frac{4t}{25 + 10t} > -1.
\end{aligned}$$

Now, (2.16) and (2.17) imply

$$\phi(x') - \phi(x) \leq -\frac{t}{M} \sum_{m=1}^{M} \ln\left(1 + \frac{\tau M}{t} p_m \left(f_m - \hat{f}_m\right)\right) = L + R,$$

with

$$L = -\tau p^T (f - \hat{f})$$

and

$$R = \frac{t}{M} \sum_{i=2}^{\infty} \frac{1}{i} \left(\frac{-\tau M}{t}\right)^i \sum_{m=1}^{M} p_m^i \left(f_m - \hat{f}_m\right)^i.$$

But,

$$\begin{aligned} \left| \frac{1}{i} \left(\frac{\tau M}{t}\right)^i \sum_{m=1}^{M} p_m^i (f_m - \hat{f}_m)^i \right| &\leq \frac{1}{i} \left(\frac{\tau M}{t}\right)^i \sum_{m=1}^{M} p_m^i (f_m + \hat{f}_m)^i \\ &\leq \frac{1}{i} \left(\frac{\tau M}{t} \sum_{m=1}^{M} p_m (f_m + \hat{f}_m)\right)^i = \frac{1}{i} \left(\frac{\tau M}{t} p^T (f + \hat{f})\right)^i, \end{aligned}$$

where the first inequality follows from the nonnegativity of each f_m.

By induction on the number of iterations of Algorithm $\mathcal{A}_L$ we can assume that $\phi(x) \leq \phi(x_0)$, see (2.15). Now, from Lemma 3 we have $p^T(f + \hat{f}) \leq 5/2$, and hence,

$$R \leq \frac{t}{M} \sum_{i=2}^{\infty} \frac{1}{i} \left(\frac{5M\tau}{2t}\right)^i = -\frac{t}{M} \left[\ln\left(1 - \frac{5M\tau}{2t}\right) + \frac{5M\tau}{2t}\right]. \tag{2.18}$$

On the other hand,

$$p^T (f - \hat{f}) = \theta(f) - t - p^T \hat{f} \geq t\,,$$

for otherwise x would satisfy the stopping criterion (2.8) and the algorithm would halt before computing the iterate x'. This implies that $L \leq -t\tau$.

This bound on L, along with (2.18), yields

$$\phi(x') - \phi(x) \;\leq\; L + R \;\leq\; -t \left\{\tau + \frac{1}{M} \left[\ln\left(1 - \frac{5M\tau}{2t}\right) + \frac{5M\tau}{2t}\right]\right\}.$$

The value of $\tau = M^{-1}\ 4t^2/(10t+25)$, with which Algorithm $\mathcal{A}_L$ is initialized, maximizes the expression in the curly brackets above. For this τ we have

$$\phi(x') - \phi(x) \ \leq\ -\frac{t}{M}\left\{\frac{2t}{5} - \ln\left(1+\frac{2t}{5}\right)\right\}.$$

Since $(u-\ln(1+u))/u^2$ is a monotonically decreasing function of $u = 2t/5$ and $t = 3\sigma/7 \leq 1/7$, we obtain

$$2t/5 - \ln(1+2t/5) \geq \alpha(2t/5)^2$$

with $\alpha = (2/35 - \ln(1+2/35))/(2/35)^2 = .48\ldots$. Hence,

$$-\frac{t}{M}\left\{\frac{2t}{M} - \ln\left(1+\frac{2t}{5}\right)\right\} \ \leq\ -\alpha\left(\frac{2}{5}\right)^2\frac{t^3}{M} \ <\ -\frac{1}{13}\frac{t^3}{M},$$

which proves the lemma. □

We are now in a position to complete the proof of Theorem 1. By (2.14) and Lemma 1,

$$\phi(x_0) \leq \theta(x_0) - t\ln t \leq \lambda(x_0) + t - t\ln t.$$

But for any iterate x of Algorithm $\mathcal{A}_L$, Corollary 1 implies that

$$\phi(x) \geq t + (1-t)\theta(x) \geq t + (1-t)\lambda^* = \lambda^* + (1-\lambda^*)t \geq \lambda^*.$$

Hence,

$$\phi(x_0) - \phi(x) \leq \lambda(x_0) - \lambda^* + t\ln(e/t). \tag{2.19}$$

Since each iteration of the algorithm decreases the value of $\phi(x)$ by at least $t^3/13M$ (Lemma 4), the total number of iterations, N, cannot exceed

$$13M\left(\frac{\lambda(x_0)-\lambda^*}{t^3} + \frac{\ln(e/t)}{t^2}\right) = O\left(M\left(\frac{\lambda(x_0)-\lambda^*}{\sigma^3} + \frac{\ln 1/\sigma}{\sigma^2}\right)\right),$$

which coincides with the expression claimed in (2.9). □

2.3. Solving $(\mathcal{P})$ to a given relative accuracy with Algorithm $\mathcal{A}_L$. The next theorem generalizes the above iteration bound for any starting point $x_0 \in B$ and for a given *relative* accuracy $\varepsilon > 0$. Its proof invokes a *ternary search* procedure $\mathcal{TS}$, similar to that in [16], which calls Algorithm $\mathcal{A}_L$ as a subroutine to gradually decrease the value of $t \equiv 3\sigma/7$.

$\mathcal{TS}$ maintains the interval $[\underline{\lambda}, \bar{\lambda}]$, its length $\delta = \bar{\lambda} - \underline{\lambda}$, and the parameters $\lambda_1 = \underline{\lambda} + \delta/3$, $\lambda_2 = \underline{\lambda} + 2\delta/3$, $\nu = 1/(2\underline{\lambda} + \bar{\lambda})$, and $\sigma = \nu\delta/3$. Initially, it sets $\underline{\lambda} = 0$, $x = x_0$, and $\bar{\lambda} = \lambda(x)$, and then repeats the following two steps until $\delta \leq \varepsilon\underline{\lambda}$ is satisfied:

Step 1. Call Algorithm $\mathcal{A}_L(\nu f, x, \sigma)$ to compute a point $y \in B$ such that $\nu\lambda(y) \leq \nu\lambda^* + \sigma$, that is, $\lambda(y) \leq \lambda^* + \delta/3$.

Step 2. If $\lambda(y) \leq \lambda_2$, replace x by y and $\bar{\lambda}$ by $\lambda(y)$; otherwise ($\lambda(y) > \lambda_2$), replace $\underline{\lambda}$ by λ_1. □

THEOREM 2. *For any initial point $x_0 \in B$ and a number $\varepsilon \in (0,1]$, the ternary search procedure with* Algorithm $\mathcal{A}_L$ *halts in a total of*

$$N = O\left(M\left(\varepsilon^{-2}\ln\varepsilon^{-1} + \ln\frac{\lambda(x_0)}{\lambda^*}\right)\right)$$

iterations of Algorithm $\mathcal{A}_L$ *with an $x \in B$ such that $\lambda(x) \leq (1+\varepsilon)\lambda^*$.*

Proof. Since $\sigma \leq 1/3$ and $\nu\lambda(x_0) \leq 1$, the call to Algorithm $\mathcal{A}_L$ in Step 1 of $\mathcal{TS}$ is legitimate. By Theorem 1 it produces a solution $y \in B$ such that $\lambda(y) \leq \lambda^* + \delta/3$. The replacement of $\underline{\lambda}$ by λ_1 in Step 2 is also correct since $\lambda_2 < \lambda(y)$ implies that $\lambda^* > \lambda_1 = \underline{\lambda} + \delta/3$. Hence, $\mathcal{TS}$ correctly generates a sequence of intervals $[\underline{\lambda}, \lambda(x)]$ each of which contains λ^* and shrinks by at least a factor of $2/3$ from one iteration of $\mathcal{TS}$ to the next.

In order to bound the total number of iterations of Algorithm $\mathcal{A}_L$ induced by $\mathcal{TS}$, observe that at each step of $\mathcal{TS}$ we have

$$\frac{\nu\lambda(x) - \nu\lambda^*}{\sigma} = \frac{\nu(\bar{\lambda} - \lambda^*)}{\sigma} \leq \nu\delta/\sigma = 3. \tag{2.20}$$

Since Algorithm $\mathcal{A}_L$ works with the scaled vector function νf, the inequality in (2.20) implies that the second term in the bound (2.9) dominates the first. Thus, by Theorem 1, the number of iterations Algorithm $\mathcal{A}_L$ performs for a step of $\mathcal{TS}$ is bounded by $O(M\sigma^{-2}\ln 1/\sigma)$ for the value of σ applicable for that step.

First consider those steps of $\mathcal{TS}$ for which $\underline{\lambda} = 0$ (and consequently $\sigma = 1/3$). The number of such steps is $O(\ln\lambda(x_0)/\lambda^*))$ and hence the number of iterations of Algorithm $\mathcal{A}_L$ they induce can be bounded by $O(M(\ln\lambda(x_0)/\lambda^*)))$.

Next, consider those steps of $\mathcal{TS}$ for which $\underline{\lambda} > 0$. For such steps, σ and ε are of the same order. Thus, the total number of iterations of Algorithm $\mathcal{A}_L$ they induce is dominated by those induced by the last step of $\mathcal{TS}$.

The bound on the total number N of iterations of Algorithm $\mathcal{A}_L$ in the claim is obtained by combining these two estimates. □

Finally, we can assume without loss of generality that $\lambda(x_0)/\lambda^* \leq M$ since such an initial $x_0 \in B$ can be computed by solving the block problem (2.7) with the price vector $p = e/M$. Theorem 2 then implies the iteration bound (1.2) claimed in the Introduction.

2.4. Computing the duals. A useful property of Algorithm $\mathcal{A}_L$ is that in addition to a primal solution $x \in B$ satisfying (2.10), that is,

$$\lambda(x) = \max\{p^T f(x) \mid p \in P\} \leq \lambda^* + \sigma,$$

the algorithm also produces a dual solution $p \in P$ satisfying

$$g(p) = \min\{p^T f(x) \mid x \in B\} \geq \lambda^* - \sigma.$$

The latter inequality follows from (2.11), (2.8), Lemma 1, the assumption $\lambda^* \leq 1$, and the value of $t = 3\sigma/7$ from (2.6) as follows:

$$\min\{p^T f(x) \mid x \in B\} \geq \frac{p^T \hat{f}}{1+\sigma/7} \geq \frac{\theta(f)-2t}{1+\sigma/7} \geq \frac{\lambda(f)-2t}{1+\sigma/7} \geq \frac{\lambda^*-2t}{1+\sigma/7} \geq \lambda^* - \sigma.$$

Since the last step of procedure $\mathcal{TS}$ guarantees an absolute accuracy of $\lambda^*\varepsilon$ for the primal solution $x \in B$, the above observation implies that it guarantees the same absolute accuracy for the dual solution $p \in P$. Hence, the last price vector p in the stopping criterion (2.8) can be used as an ε-approximate optimal dual solution: $g(p) = \min\{p^T f(x) \mid x \in B\} \geq (1-\varepsilon)\lambda^*$.

2.5. Complexity of the coordination round. In stating Algorithm $\mathcal{A}_L$ and its analysis we have implicitly assumed that the root of equation (2.3) can be computed exactly. It is easy to see that computing this root, and the vector p, to an accuracy of $O(\ln M/\varepsilon)$ bits guarantees the iteration bound given by Theorem 3. A straightforward way to compute $\theta(f)$ to this accuracy is to use binary search on the interval $\theta \in \lambda(f) + [t/M, t]$ given by Lemma 1. This would require $O(\ln(M/\varepsilon))$ computations of the sum $\sum_{m=1}^{M} 1/(\theta - f_m)$ per iteration of Algorithm $\mathcal{A}_L$, resulting in $O(M \ln(M/\varepsilon))$ arithmetic operations per iteration. The corresponding parallel time on $M/\ln M$ processors is $O(\ln M \ln(M/\varepsilon))$.

Although the above approach may be adequate in practice, the bound can be improved using Newton's method to compute $\min_\theta \Phi(\theta, f)$. For a fixed f, the function $\Phi(\theta, f)$ is t/M–self-concordant [24] in θ and hence each Newton iteration decreases Φ by a factor of t/M unless θ is in the region of quadratic convergence

$$\frac{\left|M/t - \sum_{m=1}^{M} 1/(\theta - f_m)\right|}{\sqrt{\sum_{m=1}^{M} 1/(\theta - f_m)^2}} \leq \frac{1}{2} \tag{2.21}$$

(see Section 2.2.3 in Nesterov and Nemirovskii [24]). From (2.19) it follows that the total decrease in Φ over all iterations of Algorithm $\mathcal{A}_L$ within a step of procedure $\mathcal{TS}$ is bounded by $\nu\lambda(x_0) - \nu\lambda^* + t\ln(e/t) \leq 1 + t\ln(e/t) \leq 1 + \min\{t\ln(e/t) \mid t \in (0, 1/7]\} = O(1)$. Hence, the total number of Newton iterations outside the region of quadratic convergence (2.21) can be bounded by $O(M/t)$ for all iterations of Algorithm $\mathcal{A}_L$ within a step of $\mathcal{TS}$. On the other hand, since Algorithm $\mathcal{A}_L$ spends $O(Mt^{-2}\ln 1/t)$ iterations per stept of $\mathcal{TS}$, the amortized number of Newton iterations outside (2.21) per iteration of Algorithm $\mathcal{A}_L$ is $O(t/\ln 1/t) = O(1)$. It follows that the amortized number of all Newton iterations per iteration of Algorithm $\mathcal{A}_L$ is $O(\ln\ln(M/t))$, i.e., $O(\ln\ln(M/\varepsilon))$. The complexity of each Newton iteration is of the same order as that for binary search. Thus, the coordination task can be implemented to run in $O(M\ln\ln(M/\varepsilon))$ arithmetic operations, or in $O(\ln M \ln\ln(M/\varepsilon))$ parallel time on $M/\ln M$ processors. So, the sequential running time of each coordination round is essentially linear in M.

2.6. Block optimization. In Algorithm $\mathcal{A}_L$, the block problem (2.7) is stated over the original block B, that is, without imposing the constraints $f(x) \leq \mu e$. For the block-structured case $B = B_1 \times \cdots \times B_K$, each iteration of Algorithm $\mathcal{A}_L$ decomposes into K independent problems: compute an $\hat{x}^k \in B^k$ such that

$$p^T f^k(\hat{x}^k) \leq \left(1 + \frac{\sigma}{7}\right) \min\{\, p^T f^k(y^k) \mid y^k \in B^k \,\}, \tag{2.22}$$

which can be solved in parallel as mentioned earlier. By the choice of $\sigma = \nu\delta/3 = (\bar{\lambda} - \underline{\lambda})/(2\underline{\lambda} + \bar{\lambda})$ in procedure $\mathcal{TS}$ (Section 2.3), the relative optimization accuracy demanded of the block solver for solving (2.22) is within a constant multiplicative factor of the desired overall relative accuracy $\varepsilon \in (0, 1]$, a reasonable requirement.

2.7. Overall complexity of logarithmic-potential unrestricted PDD. The following theorem summarizes our results on the logarithmic-potential PDD method developed in the foregoing five subsections.

THEOREM 3. *For a given relative tolerance $\varepsilon \in (0, 1]$, price directive decomposition with the logarithmic potential function computes an ε-approximate solution x of the resource sharing problem $(\mathcal{P})$ in*

$$N = O\left(M\left(\varepsilon^{-2} \ln \varepsilon^{-1} + \ln M\right)\right)$$

iterations. Each coordination round requires $O(M \ln\ln(M/\varepsilon))$ operations, or $O(\ln M \ln\ln(M/\varepsilon))$ operations in parallel on an $M/\ln M$-processor EREW PRAM. *Each iteration also requires K parallel unrestricted block optimizations performed to a relative accuracy of $O(\varepsilon)$.*

2.8. Lower bound on iterations for unrestricted PDD. The number of coordination steps of the overall logarithmic-potential descent method for a fixed relative error ε is $O(M \ln M)$, using a simple coordinator of complexity $O(M \ln\ln M)$ operations, as discussed above. We now give a trivial resource-sharing problem whose solution to a fixed relative accuracy of $\varepsilon = 1$ requires $\Omega(M)$ coordination steps, even with an exact block solver and with an all-powerful coordinator.

Consider the problem of minimizing the maximum arc load in a two-node network with M parallel arcs which must carry one unit of exogenous flow, i.e.,

$$\min\{\lambda \mid x_i \leq \lambda,\ i = 1, \ldots, M,\ x = (x_1, \ldots, x_M) \in B\}, \tag{2.23}$$

where the block $B = \{x \in R^M \mid e^T x = 1,\ x \geq 0\}$.

PROPOSITION 1. *Any unrestricted PDD method requires $\Omega(M)$ iterations to solve problem (2.23) to a relative accuracy of $\varepsilon = 1$.*

Proof. For any $p \in \mathbb{R}^M$ the optimal solution of the block problem (2.7) is one of the vertices of the unit simplex B. This means that any approximate flow x obtained after N iterations, being a convex combination of all iterates x obtained so far, cannot

have more than N positive components. Hence, the maximum arc load $\lambda(x) \geq 1/N$ for the current flow x. But since $\lambda^* = 1/M$, we have $1/N \leq \lambda(x) \leq (1+\varepsilon)\lambda^* = 2/M$ for any 1-approximate solution x, which readily implies $N \geq M/2 = \Omega(M)$ for $\varepsilon = 1$. □

Stated differently, the iteration bound of the logarithmic-potential PDD for a fixed relative accuracy $\varepsilon > 0$ cannot be improved by more than a logarithmic factor in M in the entire class of unrestricted PDD methods, even with an all-powerful coordinator. By the same token, the model of unrestricted PDD in Section 1.2 is too general to study the asymptotic number of iterations required to solve the general resource sharing problem $(\mathcal{P})$ with M bounding constraints for $\varepsilon \to 0$. This is because in this general model, any problem $(\mathcal{P})$ can be solved exactly ($\varepsilon = 0$) in M unrestricted PDD iterations. Accordingly, a meaningful asymptotic analysis would require bounding the complexity of the coordinator, or the information available to it, or both. We plan to address some of these aspects in a future paper.

3. Restricted PDD and the exponential-potential method.

In this section we briefly state the exponential-potential restricted PDD method for $(\mathcal{P})$ along with its complexity bounds developed in [16]. Then, we obtain a lower bound on the coordination complexity of any restricted PDD method for solving $(\mathcal{P})$ to a fixed relative accuracy of $\varepsilon = 1$. Finally, we combine the logarithmic- and the exponential-potential PDD methods so that the coordination complexity of the resulting method matches the lower bound of all restricted PDD methods within a polylogarithmic factor in M.

3.1. Exponential potential and price vector. The exponential potential PDD method for problem $(\mathcal{P})$ uses the convex (in both x and f) function

$$\Phi(f) = \ln \sum_{m=1}^{M} \exp \frac{f_m(x)}{t}, \qquad x \in B,$$

where t is a small positive parameter. This smooth function $\Phi(f) : f \to \mathbb{R}^1_+$ approximates the piecewise convex function $\lambda(f)$ as follows:

$$\frac{\lambda(f)}{t} \leq \Phi(f) \leq \frac{\lambda(f)}{t} + \ln M.$$

Therefore,

$$\frac{\lambda^*}{t} \leq \Phi^* \leq \frac{\lambda^*}{t} + \ln M,$$

where $\Phi^* = \min_{x \in B} \Phi(f(x))$ and λ^* is the unknown optimal value of $(\mathcal{P})$ as before. These two inequalities immediately imply the following lemma, where we denote $\Phi(f(x))$ by $\Phi(x)$.

LEMMA 5. *Let* $t = \sigma/(\delta + \ln M)$, *where* $\sigma > 0$ *is a given absolute tolerance and* δ *is an arbitrary positive scalar, If* $x \in B$ *minimizes* $\Phi(x)$ *over* B *to an absolute accuracy of* δ, *i.e.* $\Phi(x) \leq \Phi^*(x) + \delta$, *then* x *minimizes* $\lambda(x)$ *over* B *to an absolute accuracy* σ, *i.e.,* $\lambda(x) \leq \lambda^* + \sigma$.

We can thus replace the problem of approximately minimizing $\lambda(x)$ on $x \in B$ by the problem of approximately minimizing $\Phi(x)$.

The exponential potential PDD method we state in the next section uses as the price vector $p = p(f)$ the gradient of $\Phi(f)$ with respect to f:

$$p_m = \frac{\exp\{f_m/t\}}{\sum_{i=1}^M \exp\{f_i/t\}} \equiv \frac{\exp\{(f_m - \lambda(f))/t\}}{\sum_{i=1}^M \exp\{(f_i - \lambda(f))/t\}} , \tag{3.1}$$

where the rightmost expression provides a normalized version of p with nonnegative exponents to be used in practical implementations.

3.2. Algorithm $\mathcal{A}_E$ for solving $(\mathcal{P})$ to a given absolute accuracy. The algorithm we present in this subsection uses the values of t and δ in accordance with Lemma 5 to solve problem $(\mathcal{P})$ to a given absolute accuracy of $\sigma \in (0, 1/3]$. We assume, as before, that a point $x_0 \in B$ for which $\lambda(x_0) \leq 1$ is given.

Algorithm $\mathcal{A}_E(f, x_0, \sigma)$:

Step 0. (Initialization.) Set $x = x_0$ and select

$$\delta = 6 \ln M, \quad t = \frac{\sigma}{7 \ln M} , \quad \tau = \frac{\sigma t}{6\mu} ,$$

where μ is any number in $[\lambda(x_0), 1]$.

Step 1. (Compute prices.) Set $f = f(x) \in \mathbb{R}_+^M$ and compute the price vector $p = p(f) \in \mathbb{R}_+^M$ by (3.1).

Step 2. (Block optimization.) For each block $k = 1, \ldots, K$, perform the following two steps:

Step 2.1. Compute an $\hat{x}^k \in B_\mu^k = \{y^k \in B^k \mid f^k(y^k) \leq \mu e\}$ such that

$$p^T f^k(\hat{x}^k) \leq \left(1 + \frac{\sigma}{7}\right) \min\left\{ p^T f^k(y^k) \mid y^k \in B_\mu^k \right\} . \tag{3.2}$$

Step 2.2. For the step length $\tau^k = \tau$, compute[2]

$$\begin{aligned} \Delta_\Phi^k(\tau^k) &= \Phi(y(k, \tau^k)) - \Phi(x) \\ &= \ln \sum_{m=1}^M p_m \exp\left\{ \frac{f_m^k\left((1-\tau^k)x^k + \tau^k \hat{x}^k\right) - f_m^k(x^k)}{t} \right\} , \end{aligned}$$

[2] The step length τ^k can be determined in practice by a line search minimizing Φ on $\tau^k \in (0, 1]$.

where $y(k,\tau^k)$ is the point obtained from x by replacing x^k by $(1-\tau^k)x^k+\tau^k\hat{x}^k$.

Step 3. (Stopping criterion.) If the inequality $\sum_{k=1}^{K}\Delta_\Phi^k(\tau^k)\le -\sigma^2/28\mu$ is not satisfied, report x and stop.

Step 4. (New iterate.) Select a k that minimizes $\Delta_\Phi^k(\tau^k)$ over $k=1,\dots,K$. Replace x by $y(k,\tau^k)$. Go to Step 1 to start a new iteration. □

The coordination complexity of this algorithm is given by the theorem below.

THEOREM 4. [16] *For any initial point $x_0\in B$ such that $\lambda(x_0)\le 1$, and for any $\sigma\in(0,1/3]$, Algorithm $\mathcal{A}(f,x_0,\sigma)$ halts in*

$$N=O\left(\frac{K}{\sigma^2}\ln M\ln\left(2+\frac{\lambda(x_0)-\lambda^*}{\sigma}\right)\right) \tag{3.3}$$

iterations with an $x\in B$ such that $\lambda(x)\le\lambda^*+\sigma$.

3.3. Solving $(\mathcal{P})$ to a given relative accuracy and the complexity of exponential-potential PDD. Applying the ternary search procedure $\mathcal{TS}$ of Section 2.3 along with Algorithm $\mathcal{A}_E$ of Section Section 3.2, we readily obtain the following theorem.

THEOREM 5. [16] *Given any initial point $x_0\in B$ and a number $\varepsilon\in(0,1]$, problem $(\mathcal{P})$ can be solved to a relative accuracy of ε in*

$$N=O\left(K\ln M\left(\varepsilon^{-2}+\ln\frac{\lambda(x_0)}{\lambda^*}\right)\right). \tag{3.4}$$

iterations of the exponential-potential PDD method.

As before, the term $\ln(\lambda(x_0)/\lambda^*)$ can be replaced by $O(\ln M)$ by initially solving the block problems (3.2) with $\sigma=1$ and $p=e/M$. This justifies the following result, where an "operation" means either an arithmetic operation or exponentiation of a nonnegative number.

THEOREM 6. *The exponential-potential PDD method computes an ε-approximate solution of $(\mathcal{P})$ in*

$$N_E=O\left(M\left(\varepsilon^{-2}\ln\varepsilon^{-1}+\ln M\right)\right)$$

iterations. Moreover, for $K=O(M/\ln M)$, the complexity of a coordination round of this method is $O(M)$ operations, or $O(\ln M)$ operations in parallel on an $M/\ln M$-processor EREW PRAM. *Each iteration requires K parallel restricted block optimizations performed to a relative accuracy of $O(\varepsilon)$.*

3.4. Overall complexity of combined logarithmic-exponential restricted PDD. For $K=O(M/\ln M)$ the above iteration bound N_E is better than that of the logarithmic-potential PDD method, which can formally be viewed as restricted PDD with $\mu=+\infty$. (Another possibility is to to use the logarithmic-potential method with

μ-restricted block optimizations, in which case the analysis of Sections 2.1–2 fully applies and gives the same iteration bound.)

The *combined* restricted PDD uses either the logarithmic- or the exponential-potential restricted PDD method in order to minimize the total iteration bound. The complexity of the resulting *logarithmic-exponential* PDD method is summarized in Theorem 7, which follows from Theorems 3 and 6. (For the sake of simplicity we have slightly increased the bounds of these two theorems below.)

THEOREM 7. *Consider the logarithmic-exponential restricted PDD method which uses the logarithmic potential function if $K > M/\ln M$ and the exponential potential otherwise. The so-combined method solves the general resource sharing problem $(\mathcal{P})$ to any given relative accuracy of $\varepsilon \in (0,1]$ in*

$$N = \min\{\ M, K\ln M\ \}\ O(\varepsilon^{-2}\ln\varepsilon^{-1} + \ln M) \tag{3.5}$$

iterations. Each coordination round of the method can be executed sequentially in $O(M\ln\ln(M/\varepsilon))$ amortized operations, or in $O(\ln\ M\ln\ln(M/\varepsilon))$ operations in parallel on an $M/\ln M$-processor EREW PRAM. *Each iteration requires K parallel restricted block optimizations performed to a relative accuracy of $O(\varepsilon)$.*

3.5. Lower bound on iterations for restricted PDD. In this section we give a lower bound based on a simple instance of $(\mathcal{P})$, and we shall hence assume a restricted PDD method with en *exact* block solver.

Consider the following simple concurrent flow problem on a two-node network with M parallel arcs to accommodate a unit exogenous flow for each of K commodities, so as to minimize the maximum arc load, i.e.,

$$\min\left\{\lambda \;\middle|\; \sum_{k=1}^{K} x_m^k \le \lambda,\quad m=1,\ldots,M,\quad x^k \in B^k\right\}, \tag{3.6}$$

where x_m^k is the flow of commodity k on arc m, and

$$B^k = B = \left\{x^k \;\middle|\; \sum_{m=1}^{M} x_m^k = 1,\quad x^k \ge 0\right\}$$

is the unit simplex for $k = 1,\ldots,K$.

PROPOSITION 2. *Any restricted PDD method $(\lambda(x) \le \mu \le +\infty)$ requires $\Omega(R(M,K))$ iterations to solve problem (3.6) to a relative accuracy of $\varepsilon = 1$, where $R(M,K)$ is defined by (1.4).*

Proof. Let $M(r)$ be the number of arcs that carry positive total flow at iteration $r = 1,2,\ldots$, i.e.,

$$M(r) = \left|\left\{m \in \{1,\ldots,M\} \;\middle|\; x_m(r) = \sum_{k=1}^{K} x_m^k(r) > 0\right\}\right|.$$

Since the network accommodates K units of flow on $M(r)$ arcs, $\lambda(x(r)) \geq K/M(r)$ and hence

$$\mu(r) \geq K/M(r). \tag{3.7}$$

Next, the coordinator passes the same price vector p and the same scalar $\mu(r)$ to each block $k = 1, \ldots, K$. Therefore, the block solvers produce identical optimal solutions $\hat{x}^1 = \ldots = \hat{x}^K$ at each iteration, obtained by minimizing $p^T y^k$ over the $\mu(r)$-restricted block

$$B_\mu^k = \left\{ y^k \;\middle|\; \sum_{m=1}^{M} y_m^k = 1, \quad y^k \leq \mu(r)e, \quad y^k \geq 0 \right\}.$$

Moreover, due to the special structure of B_μ^k, we can assume without loss of generality that $\hat{x}^k$ has at most $\lceil 1/\mu(r) \rceil$ positive components. This readily implies that, at each iteration r, the number of arcs with positive total flow can increase by at most $\lceil 1/\mu(r) \rceil$. Hence, from (3.7), we obtain the recurrence

$$M(r+1) \leq M(r) + \lceil M(r)/K \rceil \tag{3.8}$$

with $M(1) = 1$. (Nevertheless, observe that in Step 3 of the general PDD scheme (Section 1.2), the coordinator may select distinct convex combinations or step lengths within each block so that the x^k need not be the same for all commodities k.)

Since the optimal value λ^* of problem (3.6) is K/M, we must have $M(r) \geq M/2$ for any 1-approximate solution $x(r)$ of this problem. Hence any restricted PDD method requires at least

$$N = \min\{r \mid M(r) \geq M/2\} \tag{3.9}$$

iterations to produce a 1-approximate solution to (3.6), where $M(r)$ satisfies the recurrence (3.8). By induction on r we immediately obtain

$$M(r) \leq r \quad \text{for} \quad r \leq K, \tag{3.10}$$

which implies

$$N \geq M/2 \quad \text{for} \quad M \leq 2K. \tag{3.11}$$

In particular, $N = \Omega(M)$ for $M \leq K$, which proves the proposition for $M \leq K$. Furthermore, (3.11) also implies that

$$N \geq \frac{M}{2} \geq \frac{K}{2} \geq \frac{K}{2(1+\ln 2)} \left(1 + \ln \frac{M}{K}\right) = \Omega\left(K\left(1 + \ln \frac{M}{K}\right)\right)$$

for $K \leq M \leq 2K$, which shows the proposition for all $M \leq 2K$.

It remains to show that the bound $N = \Omega\left(K\left(1+\ln(M/K)\right)\right)$ also holds for $M > 2K$. To this end, define $Q = \min\{r \mid M(r) \geq K\}$. By (3.10) we have $K \leq Q$. We can also assume with no loss of generality that $Q < +\infty$, for otherwise $M(r) < K$ for all r and hence $\lambda(x(r)) \geq K/M(r) > 1 > 2K/M = 2\lambda^*$. This means that the algorithm would never achieve the required relative accuracy of $\varepsilon = 1$ and there would be nothing to prove. Hence Q is well defined.

For $M(r) \geq K$, we have from (3.8) that

$$M(r+1) \leq M(r) + \lceil M(r)/K \rceil \leq M(r) + 1 + M(r)/K \leq M(r)\left(1 + \frac{2}{K}\right),$$

which implies $M(r) \leq K \exp\left(2(r-Q)/K\right)$ for $r \geq Q$. Now from (3.10) it follows that

$$N \geq Q + \frac{K}{2}\ln\frac{M}{2K} \geq K + \frac{K}{2}\ln\frac{M}{2K} \geq \frac{K}{2}\left(1 + \ln\frac{M}{K}\right)$$

for $M > 2K$. □

For a fixed ε, the ratio of the upper bound (3.5) on iterations to the lower bound given by (1.4) is $O\left((\ln M)\min\{M, K\ln M\}/R(M,K)\right)$, which is bounded by $O\left(\ln^2 M/(1+\ln\ln M)\right)$ for all values of K and M.

REFERENCES

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows* (Prentice-Hall, NJ, 1993).

[2] D.S. Atkinson and P.M. Vaidya, "A cutting plane algorithm that uses analytic centers", in *Nondifferentiable and large-scale optimization*, Mathematical Programming – B, J.L. Goffin and J.P. Vial, Eds., (to appear).

[3] D. Bertsimas and J.B. Orlin, "A technique for speeding up the solution of the Lagrangean dual", *Mathematical Programming* 63 (1994), pp. 23–45.

[4] D. Bertsikas, "Approximation procedures based on the method of multipliers", *JOTA* 25 (1977), pp. 443-449.

[5] G.W. Brown, "Iterative solution of games by fictitious play", in *Activity Alnalysis and Production*, T.C. Koopmans, Ed., Cowles Commission Monograph 13 (John Wiley, NY, 1951).

[6] G.B. Dantzig and P. Wolfe, "The decomposition algorithm for linear programming", *Econometrica* 29 (1961), pp. 767–778.

[7] H. Everett III, "Generalized Lagrange multiplier method for solving problems of optimum allocation of resources", *Oper. Res.* 11 (1963), pp. 399–417.

[8] K.R. Frisch, "Principles of linear programming – with particular reference to the double gradient form of the logarithmic potential method", Memorandum of October 18, 1954, University Institute of Economics, Oslo, Norway.

[9] K.R. Frisch, "The logarithmic potential method of convex programming", Memorandum of May 13, 1955, University Institute of Economics, Oslo, Norway.

[10] A.V. Fiacco and G.P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques* (Wiley, NY, 1968).

[11] M. Frank and P. Wolfe, "An algorithm for quadratic programming", *Naval Res. Logistics Quarterly* 3 (1956), pp. 149–154.

[12] L. Fratta, M. Gerla, and L. Kleinrock, "The flow-deviation method: an approach to store-and-forward communication network design", *Networks* 3 (1973), pp. 97–133.

[13] J.L. Goffin, A. Haurie and J.P. Vial, "Decomposition and nondifferentiable optimization with the projective algorithm", *Management Science* 38 (1992), pp. 284-302.

[14] J.L. Goffin, A. Haurie, J.P. Vial, and D.L. Zhu, "Using central prices in the decomposition of linear programs", *European J. of Operational Research* 64 (1993), pp. 393-409.

[15] D. Goldfarb and M.J. Todd, "Chapter II. Linear Programming", pp. 73–170 in *Handbook of OR & MS Vol 1*, G.L. Nemhauser et al., Eds (Elsevier Science (North-Holland), Amsterdam, 1989).

[16] M.D. Grigoriadis and L.G. Khachiyan, "Fast approximation schemes for convex programs with many blocks and coupling constraints", *SIAM J. Optimization* 4 (1994), pp. 86-107.

[17] M.D. Grigoriadis and L.G. Khachiyan, "An exponential function reduction method for block-angular convex programs", LCSR-TR-211, Laboratory for Computer Science Research, Department of Computer Science, Rutgers University, New Brunswick, NJ, May 1993.

[18] L.G. Khachiyan, "Convergence rate of the game processes for solving matrix games", *Zh. Vȳchisl. Mat. & Mat. Fiz.* 17-6 (1977) pp. 1421–1431. English translation in *U.S.S.R. Comput. Math. & Math. Phys.*, 17 (1978), pp. 78–88.

[19] L.G. Khachiyan, "Optimal algorithms in convex programming, decomposition, and sorting", pp. 161-205 in *Computers and Decision Problems*, Ju. Zhuravlev, Ed., (Science Publishers – Nauka, Moscow, 1989, in Russian).

[20] P. Klein, C. Stein, and E. Tardos, "Leighton–Rao might be practical: faster approximation algorithms for concurrent flow with uniform capacities", in *Proc. 22nd Annual ACM Symp. on Theory of Computing*, pp. 310–321, May 1990.

[21] L.S. Lasdon, *Optimization Theory for Large Systems* (The MacMillan Company, NY, 1970).

[22] T. Leighton, F. Makedon, S. Plotkin, C. Stein, E. Tardos, and S. Tragoudas, "Fast approximation algorithms for multicommodity flow problems", pp. 101–111 in *Proc. 23rd Annual ACM Symp. on Theory of Computing*, May 1991.

[23] T.S. Motzkin, "New technique for linear inequalities andoptimization", in *Project SCOOP Symp. on Linear Inequalities and Programming*, Planning Research Division, U.S. Air Force, Washington, DC, 1952.

[24] Y. Nesterov and A. Nemirovskii, *Interior Point Polynomial Algorithms in Convex Programming* (SIAM, Philadelphia 1993).

[25] R.A. Polyak, "Smooth optimization methods for minimax problems", *SIAM J. Control and Optimization* 26 (1988), pp. 1274-1286.

[26] S.A. Plotkin, D.B. Shmoys, and E. Tardos, "Fast approximation algorithms for fractional packing and covering problems", in *Proc. 32nd Annual Symp. on Foundations of Computer Science*, pp. 495–504, October 1991.

[27] T.R. Rockafellar, *Convex Analysis* (Princeton Univ. Press, Princeton, NJ, 1970).

[28] J. Robinson, "An iterative method of solving a game", *Ann. of Math.* 54 (1951), pp. 196–301.

[29] F. Shahrokhi and D.W. Matula, "The maximum concurrent flow problem", *J. ACM* 37 (1990), pp. 318–334.

[30] R.B. Schnabel, "Determining feasibility of a set of nonlinear inequality constraints", *Math. Progr. Study* 16 (1982), pp. 137-148.

[31] H.N. Shapiro, "Note on a computation method in the thoery of games", *Comm. Pure and Applied Math.* 11 (1982), pp. 587-593.

[32] N.Z. Shor, *Minimization Methods for Nondifferentiable Functions* (Springer Series in Computational Mathematics 3, Springer, Berlin, 1985).

[33] G.L. Schultz and R.R. Meyer, "An interior point method for block-angular optimization", *SIAM J. Optimization* 1 (1991), pp. 583–602.

[34] S.P. Tarasov, L.G. Khachiyan, and I.I. Erlich, "The method of inscribed ellipsoids", *Soviet Math. Dokl.* 37 (1988), pp. 226-230.

[35] E. Tardos, "Improved approximation algorithm for concurrent multicommodity flows", Tech. Rep. 872, School of OR/IE, Cornell Univ., Ithaca, NY, October 1989.

[36] P.M. Vaidya, "A new algorithm for minimizing convex functions over convex sets", in *Proc. 30th Annual IEEE Symp. on Foundations of Computer Science*, pp. 338-343 (1989).