# APPROXIMATION ALGORITHMS FOR GENERAL PACKING PROBLEMS WITH MODIFIED LOGARITHMIC POTENTIAL FUNCTION *

Klaus Jansen and Hu Zhang
*Institute of Computer Science and Applied Mathematics,*
*University of Kiel, Germany*
kj,hzh@informatik.uni-kiel.de

**Abstract** In this paper we present an approximation algorithm based on a Lagrangian decomposition via a logarithmic potential reduction to solve a general packing or min-max resource sharing problem with $M$ nonnegative convex constraints on a convex set $B$. We generalize a method by Grigoriadis et al to the case with weak approximate block solvers (i.e. with only constant, logarithmic or even worse approximation ratios). We show that the algorithm needs at most $O(M(\varepsilon^{-2}\ln\varepsilon^{-1}+\ln M))$ calls to the block solver, a bound independent of the data and the approximation ratio of the block solver. For small approximation ratios the algorithm needs at most $O(M(\varepsilon^{-2}+\ln M))$ calls to the block solver.

## 1. Introduction.

We consider the following general *packing problem* or *convex min-max resource-sharing problem* of the form:

$$(P) \qquad \lambda^* = \min\{\lambda | f(x) \leq \lambda e, x \in B\},$$

where $f : B \to \mathbb{R}^M$ is a vector of $M$ nonnegative continuous convex functions defined on a nonempty convex compact set $B$, and $e$ is the vector of all ones. Without loss of generality we assume $\lambda^* > 0$. The functions $f_m$,

*This research was supported in part by the DFG - Graduiertenkolleg, Effiziente Algorithmen und Mehrskalenmethoden, by the EU Thematic Network APPOL, Approximation and Online Algorithms, IST-1999-14084 and by the EU Research Training Network ARACNE, Approximation and Randomized Algorithms in Communication Networks, HPRN-CT-1999-00112.

$1 \leq m \leq M$, are the packing or coupling constraints. In addition we denote $\lambda(x) = \max_{1 \leq m \leq M} f_m(x)$ for any given $x \in B$. There are many applications of the general packing problem. Typical examples are scheduling on unrelated machines, job shop scheduling, network embeddings, Held-Karp bound for TSP, multicommodity flows, maximum concurrent flow, spreading metrics, approximation of metric space, and graph partitioning [1, 2, 3, 7, 9, 10].

Grigoriadis and Khachiyan [4, 5] proposed an elegant method to compute an $\varepsilon$-approximate solution for this problem; *i.e.* for a given accuracy $\varepsilon > 0$,

$$(P_\varepsilon) \qquad \text{compute } x \in B \text{ s.t. } f(x) \leq (1+\varepsilon)\lambda^* e.$$

The approach is based on the Lagrangian duality relation:

$$\lambda^* = \min_{x \in B} \max_{p \in P} p^T f(x) = \max_{p \in P} \min_{x \in B} p^T f(x),$$

where $P = \{p \in \mathbb{R}^M \mid \sum_{m=1}^M p_m = 1, p_m \geq 0\}$. Denoting $\Lambda(p) = \min_{x \in B} p^T f(x)$, a pair $x \in B$ and $p \in P$ is optimal, if and only if $\lambda(x) = \Lambda(p)$. The corresponding $\varepsilon$-approximate dual problem has the form:

$$(D_\varepsilon) \qquad \text{compute } p \in P \text{ s.t. } \Lambda(p) \geq (1-\varepsilon)\lambda^*.$$

The Lagrangian or price-directive decomposition method is an iterative strategy that solves the primal problem $(P_\varepsilon)$ and its dual problem $(D_\varepsilon)$ by computing a sequence of pairs $x$, $p$ to approximate the exact solution from above and below, respectively.

Grigoriadis and Khachiyan [5] proved that the primal problem $(P_\varepsilon)$ and the dual problem $(D_\varepsilon)$ can be solved with a $t$-approximate block solver that solves the block problem for a given tolerance $t$:

$$ABS(p,t) \qquad \text{compute } \hat{x} = \hat{x}(p) \in B \text{ s.t. } p^T f(\hat{x}) \leq (1+t) \min_{y \in B} p^T f(y).$$

Usually the number of the calls to the approximate block solver to obtain the solution required is applied as a criterion of the complexity of the approximation algorithm. The reason is that we consider the general case of these problems where the approximate block solver is unknown to us (used as an oracle in the approximation algorithm). Therefore, the running time of the approximation algorithm is just the product of the number of calls and the running time of the block solver. The number of iterations (or calls to the block solver) in [5] is $O(M(\varepsilon^{-2} \ln \varepsilon^{-1} + \ln M))$.

The method uses either the exponential or standard logarithmic potential function [4, 5] and is based on a ternary search procedure that repeatedly scales the functions $f$. Villavicencio and Grigoriadis [11] proposed a modified logarithmic potential function to avoid the scaling phases and to simplify the analysis. The number of iterations used in [11] is also $O(M(\varepsilon^{-2} \ln \varepsilon^{-1} + \ln M))$. Furthermore, Grigoriadis et al [6] studied a general covering or concave max-min resource sharing problem (that is orthogonal to the packing problem studied

here). They showed that the number of iterations or block optimization steps is only $O(M(\varepsilon^{-2}+\ln M))$ for the general covering problem. Therefore it is natural to ask whether one can improve the number of iterations for the packing problem. In fact we reduce the number of iterations for the general packing problem $(P_\varepsilon)$ and dual problem $(D_\varepsilon)$ to $O(M(\varepsilon^{-2}+\ln M))$.

On the other hand, in general the block problem is hard to approximate [1, 2]. This means that the assumption to have a block solver with accuracy $t = O(\varepsilon)$ is too strict. Therefore we consider in this paper the case where we have only a weak approximate block solver. A $(t,c)$-approximate block solver is defined as follows:

$$ABS(p,t,c) \qquad \text{compute } \hat{x} = \hat{x}(p) \in B \text{ s.t. } p^T f(\hat{x}) \le c(1+t)\min_{y\in B} p^T f(y),$$

where $c \ge 1$ is the approximation ratio. The main goal is now to solve the following primal problem (using the weak block solver)

$$(P_{\varepsilon,c}) \qquad \text{compute } x \in B \text{ s.t. } f(x) \le c(1+\varepsilon)\lambda^* e.$$

The corresponding dual problem has the form:

$$(D_{\varepsilon,c}) \qquad \text{compute } p \in P \text{ s.t. } \Lambda(p) \ge \frac{1}{c}(1-\varepsilon)\lambda^*.$$

**New results.** Our main result is an approximation algorithm that for any accuracy $\varepsilon \in (0,1)$ solves the problem $(P_{\varepsilon,c})$ in

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

iterations or coordination steps. Each step requires a call to the weak block solver $ABS(p, O(\varepsilon), c)$ and an overhead of $O(M \ln\ln(M/\varepsilon))$ arithmetic operations. Furthermore for small ratios $c$ with $\ln c = O(\varepsilon)$ we improve the number of iterations to $O(M(\varepsilon^{-2}+\ln M))$.

**Related results.** Plotkin et al [10] considered the linear feasibility variants of both problems: either to find a point $x \in B$ such that $f(x) = Ax \ge (1-\varepsilon)b$ or to find a point $x \in B$ such that $f(x) = Ax \le (1+\varepsilon)b$ where $A$ is the coefficient matrix with $M$ rows and $b$ is an $M$-dimensional vector. The problems are solved in [10] by Lagrangian decomposition using exponential potential reductions. The number of iterations (calls to the corresponding block solver) in these algorithms are $O(\varepsilon^{-2}\rho\ln(M\varepsilon^{-1}))$ and $O(M+\rho\log^2 M+\varepsilon^{-2}\rho\ln(M\varepsilon^{-1}))$, where $\rho = \max_{1\le m\le M}\max_{x\in B} a_m^T x/b_m$ is the width of $B$ relative to $Ax \ge b$. Garg and Könemann [3] proposed a $(1+\varepsilon)$ approximation algorithm to solve the linear packing problem within $O(M\varepsilon^{-2}\ln M)$ iterations which is independent of the width. Recently Young [13] has proposed an approximation algorithm for a mixed linear packing and covering problem (with convex set $B = \mathbb{R}_+^N$) with running time $O(Md\ln M/\varepsilon^2)$ where $d$ is the maximum number of constraints any variable appears in. Young [12] studied also the linear

case of the packing problem but with weak block solvers. He proposed an algorithm that uses $O(\rho' \ln M/(\lambda^* \varepsilon^2))$ calls to the block solver, where $\rho' = \max_{1\le m\le M} \max_{x\in B} a_m^T x/b_m - \min_{1\le m\le M} \min_{x\in B} a_m^T x/b_m$ and $\lambda^*$ is the optimal value of the packing problem. Furthermore, Charikar et al [1] noticed that the result in [10] for the packing problem can be extended also to the case with weak block solvers and the same number $O(\varepsilon^{-2}\rho \ln(M\varepsilon^{-1}))$ of iterations. Recently Jansen and Porkolab [8] studied the general covering problem and showed that at most $O(M(\ln M + \varepsilon^{-2} + \varepsilon^{-3} \ln c)$ coordination steps are necessary.

**Main ideas.** Our paper is strongly based on ideas in [4, 5, 6, 11]. We use the modified logarithmic potential function proposed in [11]. Our algorithm is based on the scaling phase strategy such that the relative error tolerances $\sigma$ in all phases achieve the given relative tolerance $\varepsilon$ gradually. The oracle $ABS(p, t, c)$ is called once in each iteration. We found that the stopping rules proposed in [4, 5, 11] are too strict, and that the number of iterations would be at least $O(Mc^2(\ln Mc + \varepsilon^{-3} \ln c))$. Therefore we analyzed a combination of two stopping rules in order to obtain a running time independent of $c$. In fact our result is the first one independent of the width $\rho$, the optimal value $\lambda^*$ and the approximation ratio $c$. For certain $c$ small enough, we use an upper bound for the difference of potential function values $\phi_t(x) - \phi_t(x)$ for two points $x, x' \in B$ similar to [6]. This enables us to show that the original method in [11] (with a slightly modified stopping rule) uses only $O(M(\varepsilon^{-2} + \ln M))$ coordination steps for $c$ with $\ln c = O(\varepsilon)$.

The paper is organized as follows. In Section 2 the modified logarithmic potential function and price vector are introduced and their properties are presented. The algorithm is described in details in Section 3. Finally, in Section 4 the correctness and the complexity of the algorithm are analyzed.

## 2. Modified logarithmic potential function.

In order to solve the min-max resource sharing problem $(P)$, we use the Lagrangian decomposition method that is based on a special potential function. Villavicencio and Grigoriadis [11] proposed the following modification of Karmarkar's potential function to relax the coupling constraints of $(P)$:

$$\Phi_t(\theta, x) = \ln \theta - \frac{t}{M} \sum_{m=1}^{M} \ln(\theta - f_m(x)), \tag{1}$$

where $\theta \in \mathbb{R}_+$ and $x \in B$ are variables and $t \in \mathbb{R}_+$ is a fixed tolerance parameter (that is used in the approximate block solver $ABS(p, t, c)$). In our algorithm, we shall set values of $t$ from $1/6$ initially down to $O(\varepsilon)$, where $\varepsilon$ is the desired relative accuracy for the solution. The function $\Phi_t$ is well-defined for $\lambda(f(x)) < \theta < \infty$ where $\lambda(x) = \max\{f_1(x), \ldots, f_M(x)\}$ and has the *barrier property*: $\Phi_t(\theta, x) \to \infty$ for $\theta \to \lambda(x)$ and $\theta \to \infty$.

We define the reduced potential function as the minimal value $\Phi_t(\theta, x)$ over $\theta \in (\lambda(x), \infty)$ for a given fixed $x \in B$, *i.e.*

$$\phi_t(x) = \min_{\lambda(x)<\theta<\infty} \Phi_t(\theta, x). \tag{2}$$

The minimum $\theta(x)$ of $\Phi_t(\theta, x)$ (using the first-order optimality condition) is the solution of the equation:

$$\frac{t}{M} \sum_{m=1}^{M} \frac{\theta}{\theta - f_m(x)} = 1. \tag{3}$$

The function $g(\theta) = \frac{t}{M} \sum_{m=1}^{M} \frac{\theta}{\theta - f_m}$ is a strictly decreasing function of $\theta$ in $(\lambda(x), \infty)$. Therefore the implicit function $\theta(x)$ is the unique root of (3) in the interval $(\lambda(x), \infty)$. The following two lemmas by the definition of $\theta(x)$ show the bounds of $\theta(x)$ and $\phi_t(x)$, similar to those in [11] and [6].

**Lemma 1** *$\lambda(x)/(1 - t/M) \leq \theta(x) \leq \lambda(x)/(1 - t)$ for any $x \in B$.*

**Lemma 2** *$(1 - t)\ln\lambda(x) \leq \phi_t(x) \leq (1 - t)\ln\lambda(x) + t\ln(e/t)$ for any $x \in B$.*

**Remark.** Lemmas 1 and 2 show (for certain sufficiently small values of $t$) that the minimum value $\theta(x)$ approximates $\lambda(x)$ and that the potential function $\phi_t(x)$ approximates $\ln\lambda(x)$ closely. This gives us the possibility to solve the approximation problem $(P_{\varepsilon,c})$ by minimizing the smooth function $\phi_t(x)$ over $x \in B$ based on these two Lemmas.

## 2.1. Price vector function.

The price vector $p(x) \in \mathbb{R}^M$ is defined as follows [11]:

$$p_m(x) = \frac{t}{M} \frac{\theta(x)}{\theta(x) - f_m(x)}, \qquad m = 1, \ldots, M. \tag{4}$$

According to equation (3), each price value $p_m(x)$ is nonnegative and for any $x \in B$, $\sum_{m=1}^{M} p_m(x) = 1$, which are the properties desired. Thus, we can just simply compute the price vector $p$ from (4), which is easier to calculate compared with other methods (e.g. using the exponential potential function). In view of the definition above, the following lemma holds.

**Lemma 3** *$p(x)^T f(x) = \theta(x)(1 - t)$ for any $x \in B$.*

**Remark.** By Lemma 3, for $t$ small enough, the dual value $p^T f(x)$ is only slightly less than $\theta(x)$, the minimum of the potential function. This shows that the dual value $p^T f(x)$ is also an approximation of $\lambda(x)$. Therefore the primal problem can also be solved by obtaining the solution of the dual value and in fact that is what we need to construct our algorithm.

## 3. The approximation algorithm.

The exact dual value $\Lambda(p)$ can be approximated by the dual value $p^T f(\hat{x})$, where $\hat{x}$ is the block solution computed by the $(t,c)$-approximate block solver for the current price vector $p$. Furthermore, to establish the stopping rules of the scaling phases in the approximation algorithm, the value of duality gap should be estimated in each iteration. For our first stopping rule we use the following parameter $\nu$:

$$\nu = \nu(x,\hat{x}) = \frac{p^T f(x) - p^T f(\hat{x})}{p^T f(x) + p^T f(\hat{x})}. \tag{5}$$

If $\nu$ is $O(\varepsilon)$, then the duality gap is also quite small. But for larger $\nu$ close to 1, the gap can be extremely large (see also Section 4). Therefore, we have to define a second parameter. Let $\sigma$ be the relative error of a scaling phase. Then the parameter $w$ is given by:

$$w = \begin{cases} \frac{1+\sigma}{(1+\sigma/6)M} & \text{for the first } \sigma\text{-scaling phase,} \\ \frac{1+\sigma}{1+2\sigma} & \text{otherwise.} \end{cases} \tag{6}$$

Let $x_s$ be the solution of $s$th scaling phase. Then the two stopping rules used in the $s$th scaling phase are:

$$\begin{array}{ll} \mathit{Rule}\ 1: & \nu \le \sigma/6, \\ \mathit{Rule}\ 2: & \lambda(x) \le w\,\lambda(x_{s-1}). \end{array}$$

**Remark.** Grigoriadis et al [6, 11] used either only the first stopping rule or the rule $p^T f(x) - p^T f(\hat{x}) \le \sigma\theta(x)/2$ that is similar to the former. In the case of a weak block solver, such stopping rules are not sufficient to obtain a running time independent of the ratio $c$. It may happen that the block solver is called more times than what necessary. Therefore we have introduced the second stopping rule to make sure that the scaling phase stops as soon as the solution meets the requirement of the phase. On the other hand the first stopping rule is needed to have always a constant decrement in the potential function (see Section 4).

The algorithm works now as follows. First we apply the scaling phase strategy. In each scaling phase the relative error tolerance $\sigma$ is set. Then based on the known pair of $x$ and $p$, a solution $\hat{x}$ is generated by the approximate block solver. Afterwards an appropriate linear combination of the old solution $x$ and $\hat{x}$ is computed as the new solution. The iteration stops when the solution satisfies one of the stopping rules. After one scaling phase, the error tolerance $\sigma$ is halved and the next scaling phase is started until the error tolerance $\sigma \le \varepsilon$. The solution generated in the last scaling phase solves the primal problem $(P_{\varepsilon,c})$ (see also Section 4).

In our algorithm we set $t = \sigma/6$ for the error tolerance in the block solver $ABS(p,t,c)$. To run the algorithm, we need an initial solution $x_0 \in B$ in advance. Here we use as $x_0$ the solution of the block solver $ABS(e/M, \sigma_0/6, c)$

where the price vector $e/M$ is the vector of all $1/M$'s and the initial error tolerance $\sigma_0 = 1$.

**Algorithm** $\mathcal{L}(f, B, \varepsilon, c)$:
initialize: $s := 0$, $\sigma := \sigma_0 := 1$, $t := \sigma/6$, $p := e/M$, $x := ABS(p, t, c)$
and $x_s := x$,
**while** $\sigma > \varepsilon/2$ **do** /* $\sigma$-*scaling phase* */
$s := s + 1$, $x := x_{s-1}$, $finished := false$,
**while** $not(finished)$ **do** /* *coordination step* */
computer $\theta(x)$ from (3) and $p = p(x) \in P$ from (4),
$\hat{x} := ABS(p, t, c)$,
compute $\nu = \nu(x, \hat{x})$ from (5) and $w$ from (6),
**if** either *Stopping Rule 1* or *2* is satisfied then
$x_s := x$, $finished := true$,
**else**
$x := (1 - \tau)x + \tau\hat{x}$ for an appropriate step length $\tau \in (0, 1]$,
**end**
**end**
$\sigma := \sigma/2$, $t := \sigma/6$,
**end**

We set the step length $\tau$ as:

$$\tau = \frac{t\theta\nu}{2M(p^T f + p^T \hat{f})}. \tag{7}$$

We note that the step length $\tau$ can be computed also by a line search to minimize the potential value $\phi_t$. The algorithm and the analysis remains valid if we use such a line search to compute $\tau$ (see also Section 4).

## 4. Analysis of the algorithm $\mathcal{L}$.

In this section we verify the convergence of algorithm $\mathcal{L}$ by proving that the algorithm stops in each scaling phase after a finite number of iterations. Furthermore we show that the vector $x$ computed in the final phase solves the primal problem $(P_{\varepsilon,c})$. From now on we denote $\theta = \theta(x)$, $\theta' = \theta(x')$, $f = f(x)$, $f' = f(x')$ and $\hat{f} = f(\hat{x})$. Before proving the correctness of the approximation algorithm we have the following bound for the initial solution $x_0$.

**Lemma 4** *If $x_0$ is the solution of $ABS(e/M, t, c)$ with $t = 1/6$, then $\lambda(x_0) \leq (7/6)cM\lambda^*$.*

**Lemma 5** *If algorithm $\mathcal{L}$ stops, then the computed $x \in B$ solves $(P_{\varepsilon,c})$.*

**Proof:** We shall consider both stopping rules. If the first stopping rule is satisfied, then using the definition of $\nu$ and $\nu \leq t$ we have

$$(1 - t)p^T f \leq (1 + t)p^T \hat{f}.$$

According to Lemma 3, the inequality above, and the value $p^T f(\hat{x})$ of the solution $\hat{x}$ computed by the block solver $ABS(p,t,c)$, we have

$$\theta = \frac{p^T f}{1-t} \leq \frac{1+t}{(1-t)^2} p^T \hat{f} \leq c \left(\frac{1+t}{1-t}\right)^2 \Lambda(p).$$

Denote by $\sigma_s$ the value of $\sigma$ in the $s$th scaling phase, using the fact that $\Lambda(p) \leq \lambda^*$ and $t = \sigma_s/6$, we obtain $\theta \leq c(1+\sigma_s)\lambda^*$. Then by Lemma 1, $f_m(x_s) \leq \lambda(x_s) \leq \theta(x_s) \leq c(1+\sigma_s)\lambda^*$.

If the second stopping rule is satisfied, then we consider two further cases. If the rule is satisfied in the first phase $s = 1$, then according to the value of $w$ in formula (6) and $\sigma = \sigma_0$,

$$\lambda(x_1) \leq \frac{1+\sigma_0}{(1+\sigma_0/6)M} \lambda(x_0).$$

Since $x_0$ is the solution of $ABS(e/M,t,c)$ we have $\lambda(x_0) \leq c(1+\sigma_0/6)M\lambda^*$. This implies that $\lambda(x_1) \leq c(1+\sigma_0)\lambda^*$.

Now let us assume (by induction) that $\lambda(x_{s-1}) \leq c(1+\sigma_{s-1})\lambda^*$ after scaling phase $s-1$. Thus if the second stopping rule is satisfied in phase $s$, then according to the definition of $w$ and $\sigma_s = \sigma_{s-1}/2$, we have

$$\lambda(x_s) \leq \frac{(1+\sigma_s)}{(1+2\sigma_s)} \lambda(x_{s-1}) \leq (1+\sigma_s)c\lambda^*.$$

Therefore in both cases we have $\lambda(x_s) \leq (1+\sigma_s)c\lambda^*$ for any $s > 0$. Since $\sigma \leq \varepsilon$ when the algorithm $\mathcal{L}$ halts, the solution $x = x_s$ computed in the last phase solves $(P_{\varepsilon,c})$. □

In the next Lemma we show that the potential function $\phi_t$ decreases by a constant factor in each coordination step. This enables us later to prove an upper bound for the number of iterations.

**Lemma 6** *For any two consecutive iterates $x$, $x' \in B$ within a scaling phase of Algorithm $\mathcal{L}$,*

$$\phi_t(x') \leq \phi_t(x) - \frac{t\nu^2}{4M}.$$

The proof is similar to that in [6] and we do not give the details in this paper. Now we are ready to estimate the complexity of Algorithm $\mathcal{L}$.

**Theorem 1** *For a given relative accuracy $\varepsilon \in (0,1]$, Algorithm $\mathcal{L}$ finishes with a solution $x$ that satisfies $\lambda(x) \leq c(1+\varepsilon)\lambda^*$ and performs a total of*

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

*coordination steps.*

**Proof:** If algorithm $\mathcal{L}$ finishes after a finite number of iterations with a solution $x$, then using Lemma 5 $\lambda(x) \leq c(1+\varepsilon)\lambda^*$. First we calculate a bound on the

number $N_\sigma$ of coordination steps performed in a single scaling phase. Afterwards we obtain a bound for all scaling phases. This shows also that algorithm $\mathcal{L}$ halts and proves the theorem.

Let $x$, $x'$ denote the initial and final iterates of a $\sigma$-scaling phase. In addition let $\bar{x}$ be the solution after $\overline{N}_\sigma = N_\sigma - 1$ iterations in the same scaling phase. During the phase from $x$ to $x'$ we have $\nu > t$; otherwise the phase would finish earlier. Furthermore (by the same reason) the objective value $\lambda(\bar{x}) > w\lambda(x)$. Then Lemma 6 provides

$$\phi_t(x) - \phi_t(\bar{x}) \geq \overline{N}_\sigma \frac{t\nu^2}{4M} \geq \overline{N}_\sigma \frac{t^3}{4M}.$$

By the left and right inequality of Lemma 2 we have $(1-t)\ln\lambda(\bar{x}) \leq \phi_t(\bar{x})$ and $\phi_t(x) \leq (1-t)\ln\lambda(x) + t\ln(e/t)$, respectively. Hence,

$$\overline{N}_\sigma \frac{t^3}{4M} \leq (1-t)\ln\frac{\lambda(x)}{\lambda(\bar{x})} + t\ln\frac{e}{t}.$$

This gives directly a bound for $N_\sigma$,

$$N_\sigma = \overline{N}_\sigma + 1 \leq 4Mt^{-3}\left((1-t)\ln\frac{\lambda(x)}{\lambda(\bar{x})} + t\ln\frac{e}{t}\right) + 1.$$

Next we shall bound the term $\lambda(x)/\lambda(\bar{x})$. For the first scaling phase we have $\lambda(\bar{x}) > \frac{1+\sigma}{(1+\sigma/6)M}\lambda(x)$. In this case

$$\frac{\lambda(x)}{\lambda(\bar{x})} \leq \frac{(1+\sigma/6)M}{1+\sigma} < \frac{2M}{1+\sigma}.$$

Since $t = 1/6$ and $\sigma = 1$ during the first scaling phase, there are only $O(M\ln M)$ coordination steps in the first phase. For the other scaling phases, we have $\lambda(\bar{x}) > \frac{1+\sigma}{1+2\sigma}\lambda(x)$. Therefore,

$$\frac{\lambda(x)}{\lambda(\bar{x})} \leq \frac{1+2\sigma}{1+\sigma} = 1 + \frac{\sigma}{1+\sigma}.$$

Then according to the elementary inequality $\ln(1+u) \leq u$ for $u \geq 0$,

$$\ln\frac{\lambda(x)}{\lambda(\bar{x})} \leq \ln(1+\frac{\sigma}{1+\sigma}) \leq \frac{\sigma}{1+\sigma} < \sigma.$$

Substituting this in the bound for $N_\sigma$ above and using $t = \sigma/6$, we have the expression

$$N_\sigma = O(M\sigma^{-2}\ln\sigma^{-1}).$$

Finally, the total number of coordination steps $N$ is obtained by summing the $N_\sigma$ over all scaling phases:

$$N = O(M(\ln M + \sum_{\sigma} \sigma^{-2} \ln \sigma^{-1})).$$

Since $\sigma$ is halved in each scaling phase, the sum above is further bounded by

$$\begin{aligned} \textstyle\sum_{\sigma} \sigma^{-2} \ln \sigma^{-1} &= O(\textstyle\sum_{q=0}^{\lceil \log \varepsilon^{-1} \rceil} 2^{2q} \ln(2^q)) \\ &= O(\log \varepsilon^{-1} \textstyle\sum_{q=0}^{\lceil \log \varepsilon^{-1} \rceil} 2^{2q}) \\ &= O(\varepsilon^{-2} \log \varepsilon^{-1}), \end{aligned}$$

which provides the claimed bound. □

## 4.1. Analysis for small approximation ratio $c$.

In this subsection we analyze the case that the approximation ratio $c$ of the block solver is small enough, *i.e.*, $\ln c = O(\varepsilon)$. In this case we modify the algorithm $\mathcal{L}$ as follows: we use only the first stopping rule $\nu \leq t$. This rule is similar to the one in [11]. Let $\mathcal{L}'$ be the modified algorithm. First it can be proved that the pair $x$ and $p$ computed in the last scaling phase is a solution of both primal problem $(P_{\varepsilon,c})$ and dual problem $(D_{\varepsilon,c})$, respectively. Furthermore we prove an upper bound on the difference of the potential function for any two arbitrary points $x$ and $x'$ in $B$. The proof is similar to one in [6] for the general covering problem.

**Lemma 7** *For any two iterates $x$, $x' \in B$ within a scaling phase with $\lambda(x) > 0$ and $\lambda(x') > 0$,*

$$\phi_t(x) - \phi_t(x') \leq (1-t) \ln \frac{p^T f}{p^T f'} \leq (1-t) \ln \frac{p^T f}{\Lambda(p)},$$

*where $p = p(x)$ is defined by* (4).

For the complexity of the algorithm $\mathcal{L}'$ we have the following theorem.

**Theorem 2** *In the case of $\ln c = O(\varepsilon)$, algorithm $\mathcal{L}'$ performs a total of*

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

*coordination steps.*

The proof will be given in the full version of the paper.

## 4.2. Analysis of accuracy.

In the analysis above we assumed that the price vector $p \in P$ can be computed exactly from (4). However, in practice this is not true since $\theta(x)$ is needed in (4). $\theta(x)$ is the root of (3) that in general can only be computed approximately. We can prove that if $p$ has a relative accuracy bounded by $O(\varepsilon)$, the

algorithm can still generate the desired solution. To obtain that requirement, the absolute error of $\theta$ must be bounded by $O(\varepsilon^2/M)$. Then $O(M \ln(M/\varepsilon))$ arithmetic operations are sufficient to compute the price vector $p$ with binary search. If the Newton's method is applied, only $O(M \ln\ln(M/\varepsilon))$ operations are enough. The detailed analysis will be presented in the full version of this paper. A similar argument how to resolve this problem can be found in [4, 5, 6, 11].

## 5. Conclusion.

In this paper we have proposed an approximation algorithm for the general packing problem that needs only $O(M(\ln M + \varepsilon^{-2} \ln \varepsilon^{-1}))$ coordination steps or calls to a weak block solver with approximation ratio $c$. This bound is independent of the approximation ratio $c$. However, if the original methods in [4, 5, 11] are used, then the number of iterations can be bounded only by $O(Mc^2(\ln Mc + \varepsilon^{-3} \ln c))$, which is larger than our bound here significantly. The reason is that the stopping rules in [4, 5, 11] are too strict and that the duality gap is too large.

But we note that the original method could be faster, since we don't have a lower bound on the number of iterations. Another interesting point is that our algorithm does not automatically generate a dual solution for general $c$. We can prove that only $O(M\varepsilon^{-3} \ln(M/\varepsilon))$ coordination steps are necessary to get a dual solution of $(D_{\varepsilon,c})$. It would be interesting whether this bound can be improved. We will focus on these questions in our future research.

## References

[1] M. Charikar, C. Chekuri, A. Goel, S. Guha and S. Plotkin, Approximating a finite metric by a small number of tree metrics, *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, FOCS 1998, 379-388.

[2] G. Even, J. S. Naor, S. Rao and B. Schieber, Fast approximate graph partitioning algorithms, *SIAM Journal on Computing*, 6 (1999), 2187-2214.

[3] N. Garg and J. Könemann, Fast and simpler algorithms for multicommodity flow and other fractional packing problems, *Proceedings of the 39th IEEE Annual Symposium on Foundations of Computer Science*, FOCS 1998, 300-309.

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

[5] M. D. Grigoriadis and L. G. Khachiyan, Coordination complexity of parallel price-directive decomposition, *Mathematics of Operations Research*, 2 (1996), 321-340.

[6] M. D. Grigoriadis, L. G. Khachiyan, L. Porkolab and J. Villavicencio, Approximate max-min resource sharing for structured concave optimization, *SIAM Journal on Optimization*, 11 (2001), 1081-1091.

[7] K. Jansen, Approximation algorithms for fractional covering and packing problems, and applications, Manuscript, (2001).

[8] K. Jansen and L. Porkolab, On preemptive resource constrained scheduling: polynomial-time approximation schemes, *Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization*, IPCO 2002.

[9] J. K. Lenstra, D. B. Shmoys and E. Tardos, Approximation algorithms for scheduling unrelated parallel machines, *Mathematical Programming*, 24 (1990), 259-272.

[10] S. A. Plotkin, D. B. Shmoys and E. Tardos, Fast approximation algorithms for fractional packing and covering problems, *Mathematics of Operations Research*, 2 (1995), 257-301.

[11] J. Villavicencio and M. D. Grigoriadis, Approximate Lagrangian decomposition with a modified Karmarkar logarithmic potential, *Network Optimization, P. Pardalos, D. W. Hearn and W. W. Hager, Eds, Lecture Notes in Economics and Mathematical Systems 450, SpringerVerlag, Berlin*, (1997), 471-485.

[12] N. E. Young, Randomized rounding without solving the linear program, *Proceedings of the 6th ACM-SIAM Symposium on Discrete Algorithms*, SODA 1995, 170–178.

[13] N. E. Young, Sequential and parallel algorithms for mixed packing and covering, *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, FOCS 2001, 538-546.